

Help Document PA6+Final Project

This document will go over the functionality that should have been achieved in PA6 as well as provide hints for the final project.

Catching up with PA6 Requirements:

1.Addcustomer

2.AddBooking

3.StatusWaitlist

4.StatusFlightDay

IMPORTANT:

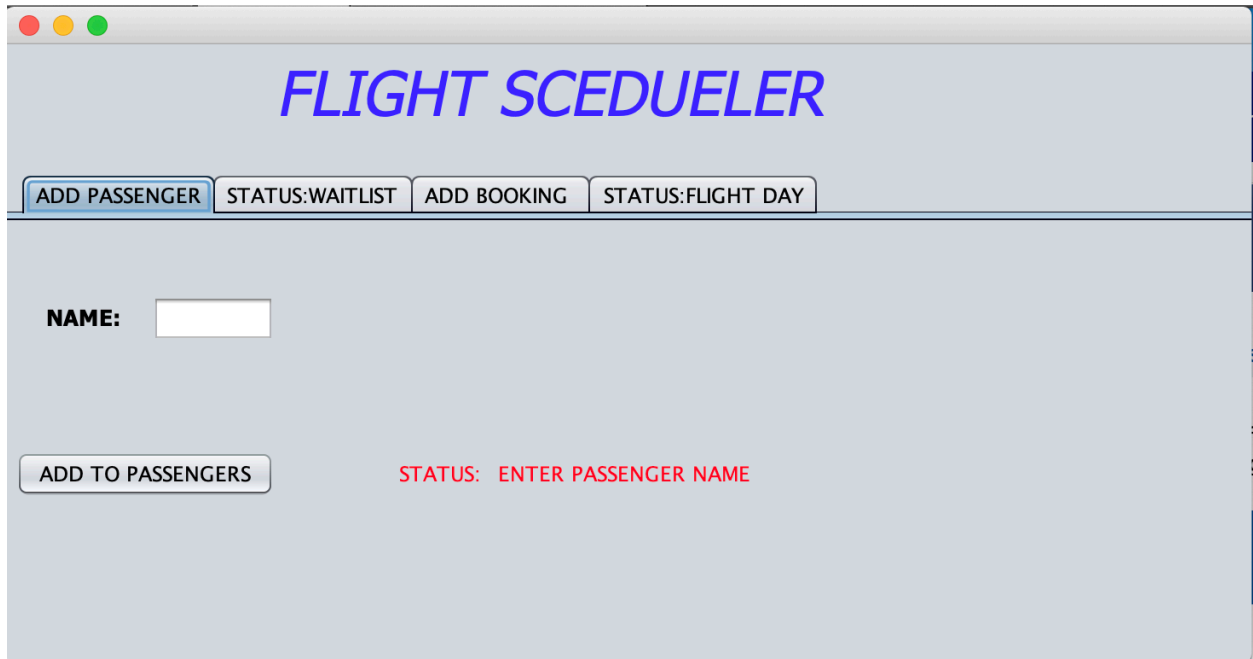
Before we begin please be advised it is best practice to write your code logic under the action listeners for the **BUTTONS** only on each page this will cause far less problems and help you complete your work smoothly, example below writes the **COMPLETE** code for adding a passenger under action listener for add passenger **BUTTON**:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    Passenger passenger = new Passenger();  
    //Create Passenger object  
    // string name = getTextfromTextfield()  
    //finalresult = addPassenger(name)  
    //print a label to let user know  
    //Update combo box for other page  
    // TODO add your handling code here:  
}
```

Same should be followed for all other functionalities.

1.ADD-CUSTOMER:

- Required: “name”



The screenshot shows a Java Swing window titled "FLIGHT SCEDUELER" in blue italicized font. The window has a light blue background and a standard macOS-style title bar with red, yellow, and green buttons. Below the title bar, there are four buttons: "ADD PASSENGER" (highlighted with a blue border), "STATUS:WAITLIST", "ADD BOOKING", and "STATUS:FLIGHT DAY". Below these buttons, there is a label "NAME:" followed by a text input field. At the bottom left, there is a button labeled "ADD TO PASSENGERS". At the bottom right, there is a red status message that reads "STATUS: ENTER PASSENGER NAME".

Code under “add button action listener”:

```
private void add-ToPassenger-Button(java.awt.event.ActionEvent evt) {  
    Passenger passenger = new Passenger();  
    //Create Passenger object  
    // string name = getTextfromTextfield()  
    //finalresult = addPassenger(name)  
    //print a label to let user know  
    //Update combo box for other page  
}
```

2.ADD-PASSENGER:

- Required:
 - Functional Add passenger tab
 - Name from combo box1
 - Flight number (preloaded to database)
 - Date of flight (preloaded to database)

The screenshot shows a window titled "FLIGHT SCEDUELER" with a light blue background. At the top, there are four buttons: "ADD PASSENGER", "STATUS:WAITLIST", "ADD BOOKING" (which is highlighted with a blue border), and "STATUS:FLIGHT DAY". Below these buttons, there are three input fields, each with a dropdown arrow: "NAME:" with a blank dropdown, "FLIGHT NUMBER:" with "F101" selected, and "DATE:" with "2018-04-10" selected. At the bottom, there is another "ADD BOOKING" button and a status message in red text that says "STATUS: Choose Flight to book".

Code under “add booking button action listener”:

```
private void Add-Booking-ButtonActionPerformed(java.awt.event.ActionEvent
evt) {
    //name = getnamefromcombobox()
    //flight = getflightnamefromcombobox()
    //date = getDatefromCombo()
    //Timestamp timestamp = new
java.sql.Timestamp(Calendar.getInstance().getTime().getTime());
    //create waitlist item
    //create bookingitem
    //createflightitem
    //seats taken = booking.getSeatsBooked(flightName, day);
    //seats left = flight.getLeftSeats(flightName);
    //if (taken = left){waitlist else book}
    //print label for the action
}
```

3.FLIGHT-DAY:

- Required:
 - Working addbooking tab
 - Flightname
 - Flightdate

FLIGHT SCEDUELER

ADD PASSENGER STATUS:WAITLIST ADD BOOKING STATUS:FLIGHT DAY

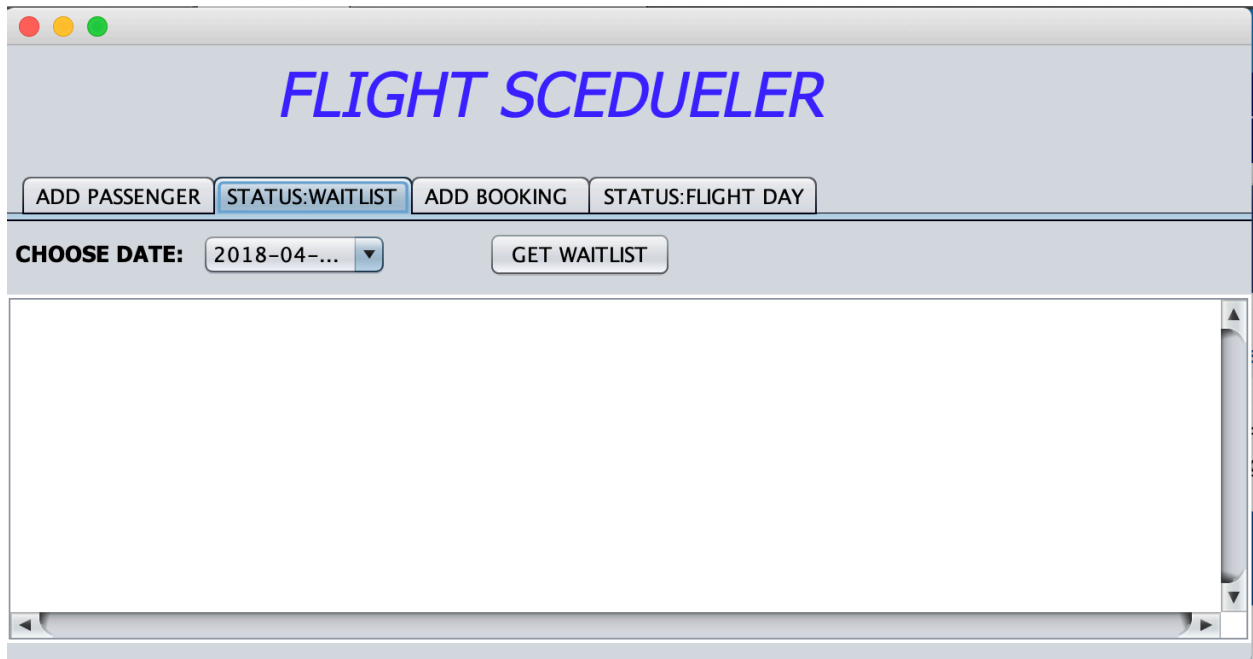
FLIGHTNAME: F101 DATE: 2018-04-10 CHECK STATUS

Code under “check status button action listener”:

```
private void Flight-DayStatus-ButtonActionPerformed(java.awt.event.ActionEvent  
evt) {  
    //create booking object  
    //date = getdatefromcombo  
    //name = getflightnamefromcombo  
    //create list  
    //result = book.sql-query-to-get-all-booked  
    //all results to be looped into the list  
    //textbox.settext(results)  
}
```

4.WAITLIST

- Required:
 - Date
 - Working booking tab



Code under “get waitlist button action listener”:

```
private void get-waitlist-button(java.awt.event.ActionEvent evt) {  
    //create wait list item  
    //date = getdatefromCombo  
    //create List  
    //result = sql-query-to-get-customers-on-waitlist-table  
    //loop all result to list  
    //textarea.setText(result)  
}
```

Hints for Final Project

Final Project Requirements:

1.AddFlight

2.AddDate

3.StatusCustomer

4.Cancel

5.Drop

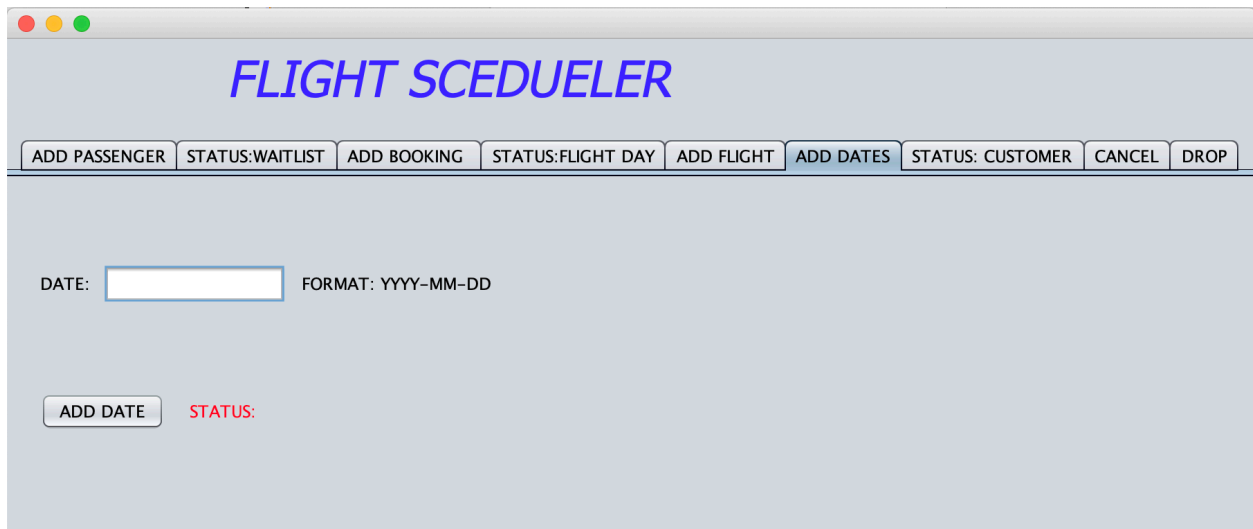
IMPORTANT:

1.All pictures below are suggested templates

2.Truncate your database of any preloaded data Final requires NO preloaded data.

1.ADD-DATE:

- Required:
 - Date text field
 - Add button
 - Status label



The screenshot shows a Java Swing window titled "FLIGHT SCEDUELER" with a light blue background. At the top, there is a menu bar with the following items: "ADD PASSENGER", "STATUS:WAITLIST", "ADD BOOKING", "STATUS:FLIGHT DAY", "ADD FLIGHT", "ADD DATES" (highlighted in blue), "STATUS: CUSTOMER", "CANCEL", and "DROP". Below the menu bar, the main area contains a "DATE:" label followed by a text input field. To the right of the input field is the text "FORMAT: YYYY-MM-DD". At the bottom left, there is a button labeled "ADD DATE". To the right of the button is a red label "STATUS:".

Hints for Code under “add Date button action listener”:

- 1.my sql table requires only one column
- 2.Apply same idea as adding passenger name
- 3.error detection
- 4.I want the status label to inform my user of the action (fail/success)

2.ADD-FLIGHT:

- Required:
 - Flight name text field
 - Seats on flight
 - Add button
 - Status label

The screenshot shows a web application window titled "FLIGHT SCEDUELER" in blue text. Below the title is a horizontal menu bar with several buttons: "ADD PASSENGER", "STATUS:WAITLIST", "ADD BOOKING", "STATUS:FLIGHT DAY", "ADD FLIGHT", "ADD DATES", "STATUS: CUSTOMER", "CANCEL", and "DROP". The "ADD FLIGHT" button is highlighted. Below the menu bar, there is a form with two input fields: "FLIGHT NAME:" followed by a text box, and "SEATS:" followed by a small square input field. At the bottom left of the form is a button labeled "ADD FLIGHT". To the right of this button is a red text label "STATUS:".

Hints for Code under “add flight button action listener”:

- 1.my sql table requires only two columns
- 2.Apply same idea as adding passenger name
- 3.error detection
- 4.I want the status label to inform my user of the action (fail/success)

3.CUSTOMER-STATUS

- Required:
 - Name of passenger
 - Button
 - Two boxes maybe? (waitlist & confirmed flight)

FLIGHT SCEDUELER

ADD PASSENGER STATUS: WAITLIST ADD BOOKING STATUS: FLIGHT DAY ADD FLIGHT ADD DATES STATUS: CUSTOMER CANCEL DROP

NAME: osman GET STATUS

Hints for Code under “get status button action listener”:

1. I want to know if my customer is on a waitlist or/and flight
2. I should be simultaneously querying and displaying data from two sql tables
3. Two objects using two sql queries for the same name of passenger (waitlistquery/bookingquery)
3. I should print my data in a consistent manner

4.CANCEL

- Required:
 - Name of passenger
 - date
 - cancel button
 - one label each for waitlist/booking

The screenshot shows a web application titled "FLIGHT SCEDUELER" in blue italicized text. Below the title is a horizontal menu bar with buttons: "ADD PASSENGER", "STATUS:WAITLIST", "ADD BOOKING", "STATUS:FLIGHT DAY", "ADD FLIGHT", "ADD DATES", "STATUS: CUSTOMER", "CANCEL", and "DROP". The "CANCEL" button is highlighted in blue. Below the menu bar, there are two dropdown menus: "NAME:" with the value "osman" and "DATE:" with the value "2018-04-...". Below these, there is a "CANCEL" button and two red status messages: "STATUS BOOKING: CHOOSE PASSENGER AND DATE" and "STATUS WAITLIST: CHOOSE PASSENGER AND DATE".

Hints for Code under “cancel button action listener”:

1. When I hit cancel I must do two things:
 - remove passenger for this date from any booking
 - remove passenger for this date from any waitlist
2. Must once again run two parallel queries with arguments from the combo boxes
3. Upon cancel I must delete said query from both table if it exists in either.

5.DROP

- Required:
 - Name of flight

FLIGHT SCEDUELER

ADD PASSENGER STATUS:WAITLIST ADD BOOKING STATUS:FLIGHT DAY ADD FLIGHT ADD DATES STATUS: CUSTOMER CANCEL DROP

FLIGHT NUMBER: F101

STATUS: CHOOSE FLIGHT TO DROP
STATUS: CHOOSE FLIGHT TO DROP
STATUS: CHOOSE FLIGHT TO DROP
STATUS: CHOOSE FLIGHT TO DROP

DROP

Hints for Code under “cancel button action listener”:

1.What I intend to do:

- Delete selected flighth from databse
- Delete anyone on waitlist on anyday for said flight.
- Check for passengers booked on this flight:
 - If yes I must book them on the next free flight date
 - Show all labels for this

2.This seems a lot, but if one chronologically checks for these conditions they should be able to do fine

3.An idea to check the next available flight for rebooking is to have flight names in an array and loop and check for the first one that has seats left using a sql query function (getSeatsLeft) if yes break and book person on this, else go on till end.

4. a great way would be to draw a picture and solve the rebooking process on paper first before just writing code!

