**FlyAway (An Airline Booking Portal)**

Course-end Project 1

**From: Rahul Sharma**

Description

As a Full Stack Developer, I have designed and developed an airline booking portal named FlyAway. The project is completed in 33 Steps.

1. All related **MySql databse is created** for **Admin, Airlines, Places and Flights**.
2. One **Dynamic Web Project** Created
3. **pom.xml** file finalized
4. **FlyAway-servlet.xml** file finalized
5. **web.xml** file finalized
6. **Entity** Flights.java and related database created
7. **Entity** Airlines & Places and related databases created
8. **Entity** Admin and related database created
9. **Entity** Passengers created
10. **Data Access Object** ParametersDAO created
11. login.jsp and isValidUser **controller** created
12. searchPage.jsp and administration.jsp created for isValidUser **controller**
13. isFormValidated **controller** created
14. result.jsp created for isFormValidated **controller**
15. registerPassengerDetails **controller** created
16. register.jsp for registerPassengerDetails **controller** created
17. bookingDetailsCapture **controller** created
18. booking.jsp for bookingDetailsCapture **controller** created
19. confirmAndPay **controller** created
20. confirmAndPay.jsp for confirmAndPay **controller** created
21. continueWithPayment **controller** created
22. continueToPay.jsp for continueWithPayment **controller** created
23. cvvCodeValidation **controller** created
24. paymentValidation.jsp for cvvCodeValidation **controller** created
25. hashCal method created to generate the transaction ID
26. otpValidation **controller** created
27. paymentConfirmation.jsp for otpValidation **controller** created
28. reDirect **controller** created to redirect the user to the search page
29. adminChangePassword **controller** created
30. adminChangePassword.jsp for adminChangePassword **controller** created
31. adminPasswordValidation **controller** created
32. adminChangePassConfirmation.jsp for adminPasswordValidation **controller** created
33. redirectII **controller** created to redirect the admin to the Dashboard

**The FlyAway portal provides following features to the User:**

● A search form in the homepage to allow entry of travel details, like the date of travel, source, destination, and the number of persons.  
● Based on the travel details entered, it will show the available flights with their ticket prices.  
● Once a person selects a flight to book, they will be taken to a register page where they must fill in their personal details. In the next page, they are shown the flight details of the flight that they are booking, and the payment is done via a dummy payment gateway. On completion of the payment, they are shown a confirmation page with the details of the booking  
● Then the user can go back to the search page to book the another flight

**The FlyAway portal provides following features to the Admin:**

● An admin login page where the admin can change the password for any user after login, if the requirement comes in.

**I have used:**

● Eclipse IDE to code for the application   
● Java: A programming language to develop the web pages, databases, and others  
● MySQL: To create tables for admin, airlines, and other specifics  
● Maven: To create a web-enabled Maven project  
● **Hibernate** to connect with the database  
● **Spring Model View Controller** to setup and develop the project efficiently  
● Git: To connect and push files from the local system to GitHub   
● GitHub: To store the application code and track its versions   
● Scrum: An efficient agile framework to deliver the product incrementally   
● Search and Sort techniques: Data structures used for the project

**Conclusion:**

* All steps are passed to GitHub repository:

<https://github.com/hirahul0011/Assessment_FlyAway>

* + Project and developer details
    - Project: **FlyAway (An Airline Booking Portal)**
    - Developer: Rahul Sharma
  + Core concepts used in the project: **MySQL, Hibernate, Spring Model View Controller, Session management techniques**, Searching and filtering/sorting techniques, Dynamic data pulling from the session, **CURD operations** on the database through Hibernate.
  + Conclusions on enhancing the application and defining the USPs (Unique Selling Points)
* Application is not closing, exiting, or throwing an exception if the user specifies an invalid input.
* Application takes the inputs from the user and response accordingly, this way it runs dynamically.
* I have used the same login page for the user and Admin, If the username is specified as Admin then the portal would serve to the Administrator otherwise it would serve to the user to book the flights as per the requirements

**Algorithms and flowcharts of the application**

Logged In as Passenger

Logged In as Admin

Portal opens the search form to search for the flights

Portal opens the Dashboard where all the user and Admin details are mentioned along with the link to change the password

If form is filled correctly

No

Portal opens the form to change the password

No

Yes

No

Yes

No

Yes

No

Yes

Yes

Password is changed successfully and link provided to the Dashboard

If password is supplied correctly

Portal confirm that flight is booked and user can print the ticket and link provided to book another ticket

If transaction validated

Portal sends the user to the Bank’s payment gateway to validate the transaction

If CVV code provided correctly

Portal opens the page to provide the CVV code for the Card

Portal opens the page to confirm the payment and personal contact details

If form is filled correctly

Portal opens the registration form where user can put all the Identification details of the passengers

User select the flight as per his/her requirement

Portal shows all the flights available as per the search form