FlyAway (An Airline Booking Portal)

-By Robin Singh Kuntal

This document contains sections for:

- Login Form for the user in HTML form
- Separate Login Form for the Admin in HTML form
- Java Servlet for admin validation
- User Dashboard for searching flights using inputs like Source, Destination and date using HTML
- Java Servlet for retrieving the required information after the user enter the details
- Admin Dashboard for Edit, Update and Delete operation for flight schedule using HTML and Java Servlet
- Java Servlet for performing CRUD (Create, Retrieve, Update and Delete) operation
- HTML page for entering details of flights which are needed to be updated
- Separate HTML page for dummy payment page
- MySQL Database creation named Flyaway1. In that creating different tables for Flight Schedule, User Details and Admin Details
- Core Concepts used in development of the project
- Unique Selling Points of the Application
- Conclusions

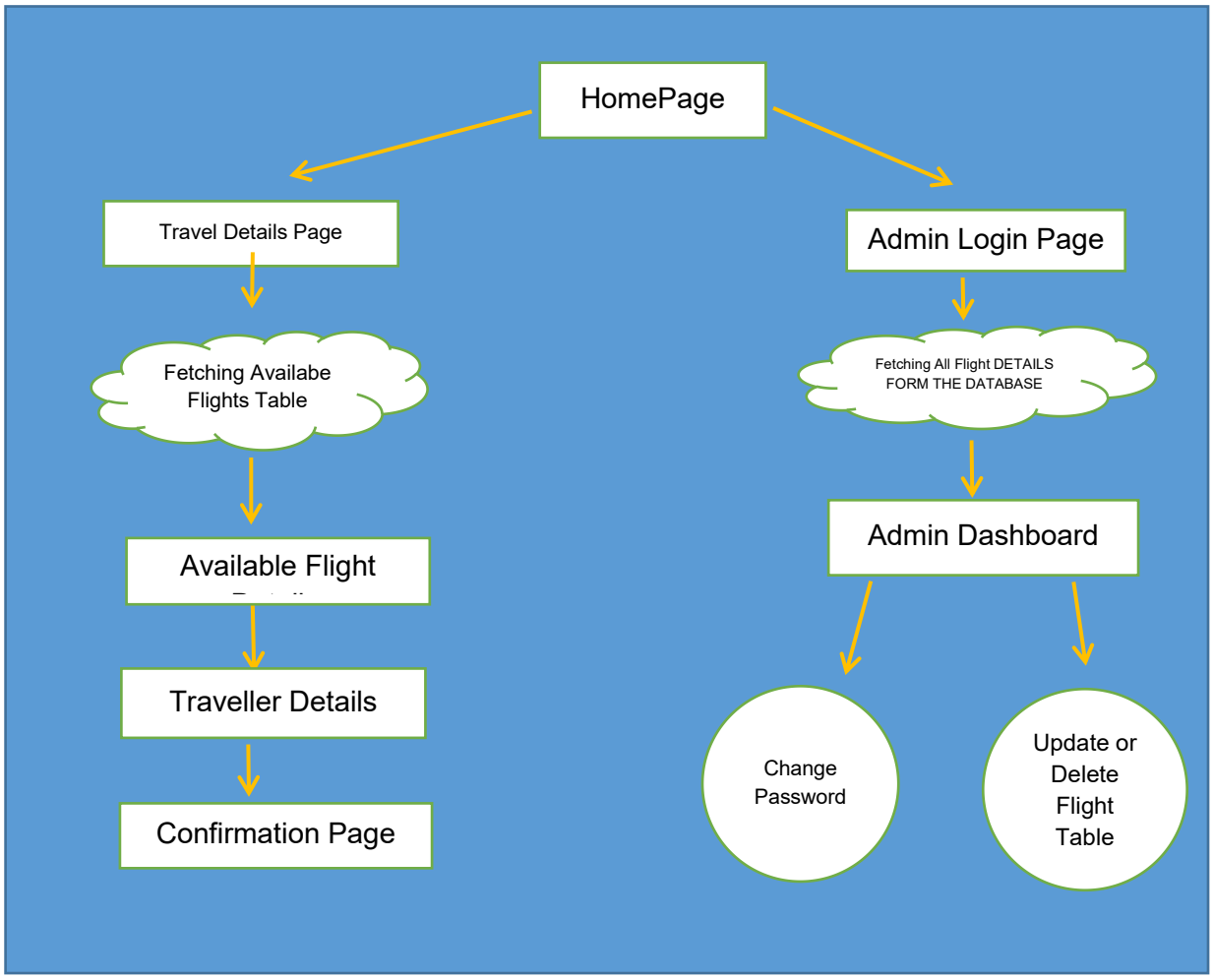
Tools and Technology used in this Project

- JSP, HTML, CSS, Bootstrap for View.
- JAVA Servlets as Controller
- MySQL database using Hibernate for Model to create tables for admin and flight details. Hosted on a remote server at AWS RDS.
- Tomcat 8.5 as an Application Server.
- Eclipse: As an IDE to code for the application.
- Java: A programming language to develop the web pages, databases.
- Maven: To create a web-enabled Maven project and build deployable war file.
- 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

Sprint Planning (Planned for 4 sprints)

Sprint number	Modules
1	Design homepage and travel details page.
2	Fetch available flights from database and redirect to user details page.
3	Checkout page with details of flight and form for payment details.
4	Admin Login page and Admin Dashboard page with all the flights data fetched from the database along with change password and add flight button. Testing

FlowChart



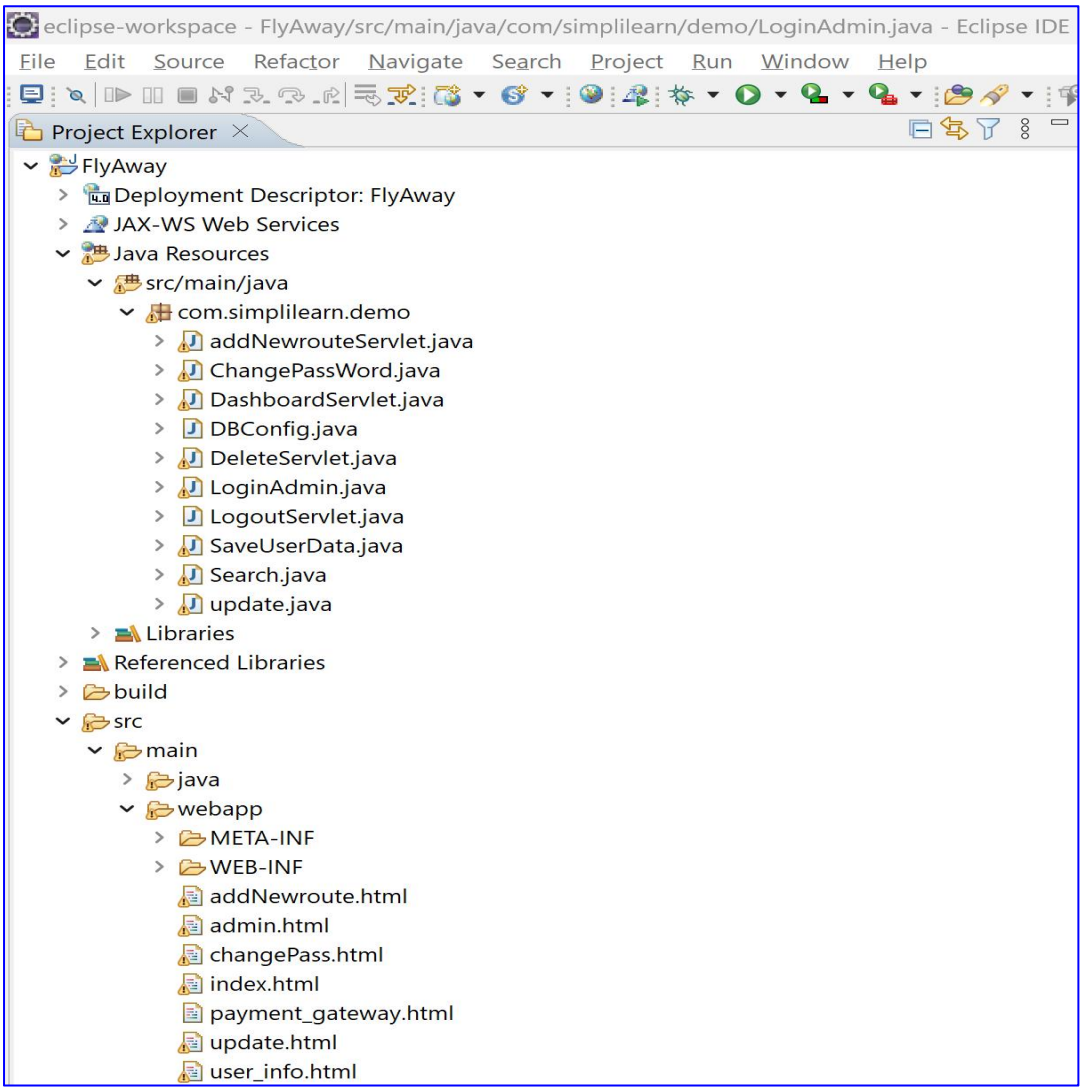
To demonstrate the product capabilities, below are the sub-sections configured to highlight appearance and user interactions for the project:

- Creating the project in Eclipse
- Creating a Login Page in HTML for a login point for the user/client (**index.html**)
- Creating a Login Servlet in Java used to validate user credentials. As per the project requirement the user credentials are hardcoded **(LoginServlet.java)**
- Creating a Dashboard Servlet in Java to Welcome the user to Dashboard after the login validation and providing a Logout option(**DashboardServlet.java**)

- Creating a Logout Servlet in Java to handle logout request and redirect user to the Login Page (**LogoutServlet.java**)
- Pushing the code to GitHub repository

Step 1: Creating a new Dynamic Web Project in Eclipse

- Open Eclipse
- Go to File -> New -> Dynamic Web Project.
- Type in any project name and click on “Finish.”
- Select your project and go to src -> main -> webapp->Right click and create new HTML file



Step 2: Creating a HomePage in HTML for a login point for the user/client **(index.html)**

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

<style type="text/css">


 .mainBox{
  display: flex;
  justify-content: center;


 }

 .inBox{
            margin:25px;
            padding:25px;
 border-width: 25px;
 border-color:yellow;

 }

 h1{
 color: red;
 }

 </style>

<div class=mainBox>
<div class = inBox>

<h1>Welcome to FlyAway Flight Booking Website</h1>

<form action="search" method="post">

From:<input type="text" name="from">
To:      <input type="text" name="to">
Date:<input type="date" name="date"><br><br>
<input type="submit" value="Search">


<a href="admin.html">Admin Dashboard</a>


</form>

</div>
</div>




</body>
</html>
```

Step 3: Creating a Login Servlet in Java used to validate Admin credentials. As per the project requirement the user credentials are hardcoded **(AdminLogin.java)**

```
package com.simplilearn.demo;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
@WebServlet("/admin")
public class LoginAdmin extends HttpServlet {
public static boolean isLoggedIn = false;
public static String password = "Admin";
public static String email = "admin@flyaway.com";
@Override
public void doPost (HttpServletRequest req,
HttpServletResponse resp) throws IOException {
PrintWriter out = resp.getWriter();
String email = req.getParameter("email");
String pass = req.getParameter("pass");
if (email.equals(LoginAdmin.email) &&
pass.equals(LoginAdmin.password)){
isLoggedIn = true;
out.println("You have LoggedIn");
resp.sendRedirect("dashboard");
}
else {
isLoggedIn = false;
out.println("Login Failed : Incorrect email or
Password");
}
out.close();
```

Step 4: Creating a Dashboard Servlet in Java to Welcome the Admin to Dashboard **(DashboardServlet.java)**

```
@WebServlet("/dashboard")
public class DashboardServlet extends HttpServlet {
private static final long serialVersionUID = 1L;
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
// TODO Auto-generated method stub
PrintWriter out=resp.getWriter();
resp.setContentType("text/html");
Properties props=new Properties();
InputStream in=getServletContext().getResourceAsStream("/WEB-
INF/application.properties");
props.load(in);
Connection conn=DBConfig.getConnection(props);
if(conn!=null) {
out.print("Connection Established");
Statement stmt;
try {
stmt=conn.createStatement();
out.print("<table width=75% border=1>");
out.print("<caption>Airline Table Schedule : </caption>");
ResultSet rs=stmt.executeQuery("select * from airline_table");
ResultSetMetaData rsmd = rs.getMetaData();
int totalColumn = rsmd.getColumnCount();


out.print("<tr>");

for(int i=1; i<=totalColumn; i++) {

out.print("<th>" + rsmd.getColumnName(i)+"</th>");

}

out.print("<tr>");

while(rs.next()) {

out.print("<tr><td>"+ rs.getInt(1)+"</td><td>"+
rs.getString(2)+"</td><td>"+

rs.getString(3)+"</td><td>"+ rs.getString(4)+"</td><td>"+
rs.getString(5)+"</td><td>"+

rs.getString(6)+"</td><td>"+ rs.getInt(7)+"</td><td>");

}

out.print("</table>");

out.print("<br><br>");

out.print("<a href=update.html>Update or Delete a Route</a>");
```

```
out.print("<br><br>");

out.print("<a href=addNewroute.html>Add a New Route</a>");

out.print("<br><br>");

out.print("<a href=changePass.html>Change Password?</a>");

out.print("<br><br>");

out.print("<a href='logout'> Logout </a>");

} catch (Exception e) {

// TODO: handle exception

}}

else {

out.println("Error While Connecting");

}}

@Override

protected void doPost(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {

// TODO Auto-generated method stub

doGet(req, resp);

}}
```

Step 3: Creating a DataBase Flyaway1 in MySQL and adding tables named airline_table, user_table and admin_table

```
Enter password: *********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 56
Server version: 8.0.31 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use flyaway1;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_flyaway1 |
+--------------------+
| admin_table        |
| airline_table      |
| user_table         |
+--------------------+
3 rows in set (0.01 sec)

mysql>
```

Adding Data dummy to all these tables

```
mysql> describe admin_table;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | bigint       | NO   | PRI | NULL    | auto_increment |
| name     | varchar(100) | YES  |     | NULL    |                |
| email    | varchar(100) | YES  |     | NULL    |                |
| password | varchar(100) | YES  |     | NULL    |                |
| mob_no   | bigint       | YES  |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

```
mysql> describe user_table;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | bigint       | NO   | PRI | NULL    | auto_increment |
| name      | varchar(100) | YES  |     | NULL    |                |
| email     | varchar(100) | YES  |     | NULL    |                |
| mob_no    | bigint       | YES  |     | NULL    |                |
| gender    | varchar(30)  | YES  |     | NULL    |                |
| mobile_no | decimal(19,2)| YES  |     | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> describe airline_table;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| id           | bigint       | NO   | PRI | NULL    | auto_increment |
| airline_name | varchar(100) | YES  |     | NULL    |                |
| capacity     | varchar(100) | YES  |     | NULL    |                |
| source       | varchar(100) | YES  |     | NULL    |                |
| destination  | varchar(100) | YES  |     | NULL    |                |
| time_taken   | varchar(100) | YES  |     | NULL    |                |
| price        | bigint       | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)
```

Step 4: Creating Search Servlet in Java for Searching the required flight and present it to the user in a Tabular Form and the user can Select the flight by clicking **BOOK** button which will redirect the user to Personal Information page

```
@WebServlet("/search")

public class Search extends HttpServlet {

private static final long serialVersionUID = 1L;

public Search() {

super();

// TODO Auto-generated constructor stub

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {

PrintWriter out=resp.getWriter();

resp.setContentType("text/html");

// get data from index page

String source = req.getParameter("from");

String destination = req.getParameter("to");

Properties props=new Properties();

InputStream in=getServletContext().getResourceAsStream("/WEB-
INF/application.properties");
```

```
props.load(in);

Connection conn=DBConfig.getConnection(props);

if(conn!=null) {

try {

PreparedStatement ps = conn.prepareStatement("SELECT * FROM
airline_table WHERE source=? AND destination=?");

ps.setString(2, destination);

ps.setString(1, source);

out.print("<table width=75% border=1>");

out.print("<caption>Search Results : </caption>");

ResultSet rs = ps.executeQuery();

ResultSetMetaData rsmd = rs.getMetaData();

int totalColumn = rsmd.getColumnCount();

out.print("Search Results for Flights from "+source+"to
"+destination);

out.print("<tr>");

for(int i=1; i<=totalColumn; i++) {

out.print("<th>" + rsmd.getColumnName(i)+"</th>");

}

out.print("<tr>");

while(rs.next()) {

out.print("<tr><td>"+ rs.getInt(1)+"</td><td>"+
rs.getString(2)+"</td><td>"+ rs.getString(3)+"</td><td>"+
rs.getString(4)+"</td><td>"+ rs.getString(5)+"</td><td>"+

rs.getString(6)+"</td><td>"+ rs.getInt(7)+"</td><td>"+"<a
href=user_info.html>BOOK</a> "+"</td>");

}

out.print("</table>");

} catch (Exception e) {

// TODO: handle exception
```

```
}}

else {

out.println("Error While Connecting");

}}

protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

// TODO Auto-generated method stub

doGet(request, response);

}}
```

Step 5: Creating HTML page **user_info.html** . Here user will has to provide his/her details which will be saved in Flyaway1 database>user_table

```
<form action="save" method="post">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required><br>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required><br>

<label for="mobile_no">Mobile Number:</label>

<input type="text" id="mobile_no" name="mobile_no" required><br>

<label for="gender">Gender:</label>

<input type="text" id="gender" name="gender" required><br>

<input type="submit" value="Book">

</form>
```

Step 6: Creating SaveUserData Servlet which get the data from **user_info.html** and then dump it in MySQL inside user_table

```
@WebServlet("/save")

public class SaveUserData extends HttpServlet {

@Override

protected void doGet(HttpServletRequest req,
HttpServletResponse resp) throws ServletException,
IOException {

// TODO Auto-generated method stub

//1. check the connectivity

Properties props=new Properties();

InputStream
in=getServletContext().getResourceAsStream("/WEB-
INF/application.properties");

props.load(in);

//2.get connection object

Connection conn=DBConfig.getConnection(props);

//3.get parameters from html

String name=req.getParameter("name");

String email=req.getParameter("email");

int mobile_no=
Integer.parseInt(req.getParameter("mobile_no"));

String gender=req.getParameter("gender");

//4.write query to insert data

PrintWriter out=resp.getWriter();
```

```
if(conn!=null) {

out.print("connection Established");

//query to insert data using prepared statements

try {

PreparedStatement stmt=conn.prepareStatement("insert into
user_table (name,email,mob_no,gender) values (?,?,?,?)");

stmt.setString(1, name);

stmt.setString(2, email);

stmt.setInt(3, mobile_no);

stmt.setString(4, gender);

int x=stmt.executeUpdate();

if(x>0) {

System.out.println("Data inserted successfully");

out.print("Data inserted Successfully");

//action

resp.sendRedirect("payment_gateway.html");

}

else {

System.out.println("Error While Inserting a Data");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}}

@Override
```

```
protected void doPost(HttpServletRequest req,
HttpServletResponse resp) throws ServletException,
IOException {

// TODO Auto-generated method stub

doGet(req, resp);

}}
```

Step 7: Creating dummy payment page in HTML format

```
 <h1>Payment Gateway</h1>

<form action="fetch" method="post">

<label for="name">Name on Card:</label>

<input type="text" id="name" name="name"
required><br>

<label for="card-number">Card Number:</label>

<input type="text" id="card-number" name="card-
number" required><br>

<label for="expiry-date">Expiry Date:</label>

<input type="text" id="expiry-date" name="expiry-
date" required><br>

<label for="cvv">CVV:</label>

<input type="text" id="cvv" name="cvv"
required><br>

<button type="submit">Submit Payment</button>
```

Step 6: Pushing the code to GitHub repository

●Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

●Initialize repository using the following command:

git init

●Add all the files to your git repository using the following command:

git add .

●Commit the changes using the following command:

git commit . -m <commit message>

●Push the files to the folder you initially created using the following command:

git push -u origin master