Java FSD with Angular: Course End Project 1

Source Code:

Configuration File:

Hibernate.config.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
 "-//HIBERNATE/HIBERNATE Configuration DTD 3.0//EN"
 "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
	<session-factory>
		<property
name="connection.driver_class">com.mysql.jdbc.Driver</property>
		<property
name="connection.url">jdbc:mysql://localhost:3306/airline</property>
		<property name="connection.username">root</property>
		<property name="connection.password">root</property>
		<property
name="dialect">org.hibernate.dialect.MySQL57Dialect</property>
		<property name="hibernate.format_sql">true</property>
		<property name="hibernate.show_sql">flase</property>
		<property name="hbm2ddl.auto">update</property>

		<mapping class="com.AirlineBookingPortal.model.Admin" />
		<mapping class="com.AirlineBookingPortal.model.Place" />
		<mapping class="com.AirlineBookingPortal.model.Flight" />
		<mapping class="com.AirlineBookingPortal.model.OpenFlights" />
		<mapping class="com.AirlineBookingPortal.model.Passenger" />

	</session-factory>

</hibernate-configuration>
```

Frontend Code:

Index.html:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Airline Booking Portal</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
	<div class="wrapper">
		<h2 class="title">Welcome to Airline Booking Portal</h2>
		<div class="links">
			<a href="admin_login.html">Admin Login</a>
			<a href="Searchflights">Find Flights</a>
		</div>
	</div>
</body>
</html>
```

Admin_login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Admin Login</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
	<main id="adminLogin" class="wrapper">
		<h2 class="title">Admin Login</h2>
		<form action="AdminVerification" method="post">
			<label for="username">Username: </label>
			<input type="email" placeholder="Username" id="username"
name="email" /> <br>
			<label for="password">Password: </label>
			<input type="password" placeholder="Password" id="password"
name="password" /><br>
			<input type="submit" value="Login"/>
		</form>
		<p>Or</p>
		<a href="index.html">Go to Home</a>
	</main>
</body>
</html>
```

dashboard.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<link rel="stylesheet" type="text/css" href="style.css">
<title>Dashboard</title>
</head>
<body>
	<div id="admin-info">
		<h2 class="title">Admin Details</h2>
		<c:set var="admin" value="${admin}" />
		<p>Name: ${admin.name }</p>
		<p>Email: ${admin.username }</p>
	</div>
	<div class="dashboard-wrapper">
		<div>
			<h2 class="title">Add Places</h2>
			<form action="AddPlace" method="post">
				<label for="place">Enter the Place Name: </label>
<input type="text"
					required="required" placeholder="Place"
id="place" name="place" /><br>
				<input type="submit" value="Add Place" />
			</form>
		</div>
		<div>
			<h2 class="title">Places in database</h2>
			<table border=2>
```

```
                                <tr>
                                        <th>Place Names</th>
                                </tr>

                                <c:forEach var="place" items="${places}">
                                        <tr>
                                                <td>${place.name}</td>
                                        </tr>
                                </c:forEach>


                        </table>
                </div>
        </div>
        <div class="dashboard-wrapper">
                <div>
                        <h2 class="title">Add flight</h2>
                        <form action="AddFlight" method="post">
                                <label for="flight_name">Enter the Flight Name:
</label> <input
                                        type="text" required="required"
placeholder="Flight Name"
                                        id="flight_name" name="flightName" /><br>
<label
                                        for="no_of_passengers">Total Number of
Passengers: </label> <input
                                        type="number" required="required"
placeholder="No of Passengers"
                                        id="no_of_passengers" name="noOfPassengers"
/><br> <label
                                        for="source">Select Source: </label> <select
id="source"
                                        required="required" name="source">
                                        <option value="">Select the source</option>
                                        <c:forEach var="place" items="${places}">
                                                <option
value="${place.id}">${place.name}</option>
                                        </c:forEach>
                                </select><br> <label for="destination">Select
Destination: </label> <select
                                        id="destination" required="required"
name="destination">
                                        <option value="">Select the
destination</option>
                                        <c:forEach var="place" items="${places}">
                                                <option
value="${place.id}">${place.name}</option>
                                        </c:forEach>
                                </select><br> <input type="submit" value="Add Flight"
/>
                        </form>
                </div>
                <div>
                        <h2 class="title">Flights in database</h2>
                        <table border=2>
                                <tr>
                                        <th>Flight Names</th>
                                        <th>Source</th>
                                        <th>Destination</th>
                                </tr>

                                <c:forEach var="flight" items="${flights}">
```

```
                                        <tr>
                                                <td>${flight.flightname}</td>
                                                <td>${flight.source.name}</td>
                                                <td>${flight.destination.name}</td>
                                        </tr>
                                </c:forEach>


                        </table>
                </div>
        </div>
        <div class="dashboard-wrapper">
                <div>
                        <h2 class="title">Allot Flight To Date</h2>
                        <form action="AddDateOfFlight" method="post">
                                <label for="flight">Select Flight: </label> <select
id="flight"
                                        required="required" name="flight">
                                        <option value="">Select the flight</option>
                                        <c:forEach var="flight" items="${flights}">
                                                <option
value="${flight.getFlightid()}">${flight.flightname}</option>
                                        </c:forEach>
                                </select><br> <label for="data">Enter date: </label>
<input type="date"
                                        required="required" placeholder="Date"
id="date" name="flightDate" /><br>
                                <input type="submit" value="Allot Flight" />
                        </form>
                </div>
                <div>
                        <h2 class="title">Flights Allocated</h2>
                        <table border=2>
                                <tr>
                                        <th>Flight Names</th>
                                        <th>Source</th>
                                        <th>Destination</th>
                                        <th>Date</th>
                                </tr>

                                <c:forEach var="of" items="${ofs}">
                                        <tr>
                                                <td>${of.flight.flightname}</td>
                                                <td>${of.flight.source.name}</td>
                                                <td>${of.flight.destination.name}</td>
                                                <td>${of.date}</td>
                                        </tr>
                                </c:forEach>


                        </table>
                </div>
        </div>
        <div class="dashboard-wrapper">
        <div>
                        <h2 class="title">Get Passenger List</h2>
                        <table border="2">
                                <tr>
                                        <th>Passenger Name</th>
                                        <th>Passenger Age</th>
                                        <th>Flight Name</th>
                                        <th>Date of Booking</th>
```

```
                                </tr>
                                <c:forEach var="passenger" items="${passengers}">
                                        <tr>
                                                <td>${passenger.name}</td>
                                                <td>${passenger.age}</td>
                                                <td>${passenger.flight.flightname}</td>
                                                <td>${passenger.date}</td>
                                        </tr>
                                </c:forEach>
                        </table>
                </div>
        </div>
</body>
</html>
```

Search_flight.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Search Flights</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
        <div class="wrapper">
                <h2 class="title">Welcome to Airline Booking Portal</h2>
                <h5 class="sub-title">Please Enter below details to find flights</h5>
                <form action="BookFlight">
                        <label for="data">Enter date: </label> <input type="date"
                                required="required" placeholder="Date" id="date"
name="flightDate" /><br>
                        <label for="source">Select Source: </label> <select
id="source"
                                required="required" name="source">
                                <option value="">Select the source</option>
                                <c:forEach var="place" items="${places}">
                                        <option
value="${place.id}">${place.name}</option>
                                </c:forEach>
                        </select><br> <label for="destination">Select Destination:
</label> <select
                                id="destination" required="required"
name="destination">
                                <option value="">Select the destination</option>
                                <c:forEach var="place" items="${places}">
                                        <option
value="${place.id}">${place.name}</option>
                                </c:forEach>
                        </select><br> <label for="no_of_passengers">Total Number of
                                Passengers: </label> <input type="number"
required="required"
                                placeholder="No of Passengers" id="no_of_passengers"
max="5"
                                name="noOfPassengers" /><br> <input type="submit"
                                value="Find Flight" />
                </form>
```

```
	</div>
</body>
</html>
```

book_flight.jsp:

```
<%@page import="com.AirlineBookingPortal.Utils.FlightUtil"%>
<%@page import="com.AirlineBookingPortal.model.Flight"%>
<%@page import="java.util.List"%>
<%@page import="com.AirlineBookingPortal.Utils.PlaceUtil"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Book Flights</title>
</head>
<body>

	<%
	request.getSession();
	String source =
PlaceUtil.getPlace(Integer.parseInt(request.getParameter("source"))).getName();
	String destination =
PlaceUtil.getPlace(Integer.parseInt(request.getParameter("destination"))).getName();
	String flightdate = request.getParameter("flightDate");
	int nop = Integer.parseInt(request.getParameter("noOfPassengers"));
	pageContext.setAttribute("nop", nop);
	pageContext.setAttribute("source", source);
	pageContext.setAttribute("destination", destination);
	pageContext.setAttribute("flightdate", flightdate);

	%>
	<h3>There are ${NoOfFlights} flights from ${source} to
		${destination} are:</h3>
	<c:forEach var="flight" items="${flights}" varStatus="loop">
		<div>
			<p>${loop.index+1}.
				${ flight.flightname} <a

	href="BookingDeatils.jsp?flightdate=${flightdate}&flightId=${flight.getFlighti
d()}&nop=${nop}">Book</a>
			</p>
		</div>
	</c:forEach>
</body>
</html
```

Booking_deatils.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="ISO-8859-1">
<title>Booking deatils</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
	<%
	String flightId = request.getParameter("flightId");
	int nop = Integer.parseInt(request.getParameter("nop"));
	String date = request.getParameter("flightdate");
	pageContext.setAttribute("flightdate", date);
	pageContext.setAttribute("nop", nop);
	pageContext.setAttribute("flightId", flightId);
	%>
	<div class="wrapper">
		<h2 class="title">Enter Personal details</h2>
		<form
action="MakePayment?flightdate=${flightdate}&flightId=${flightId}" method="post">
			<%
			int i = 0;
			pageContext.setAttribute("i", i);
			%>
			<%
			while (i++ < nop) {
			%>
			<label for="passenger<%=i%>">Passenger <%=i%> Name:
			</label> <input id="passenger<%=i%>" name="names"
placeholder="Name"
				required="required" /><br> <label
for="passenger<%=i%>age">Passenger
				<%=i%> Age:
			</label> <input id="passenger<%=i%>age" name="ages"
placeholder="Age"
				required="required" /><br>
			<%
			}
			%>
			<input type="submit" value="Submit and go to payment" />
		</form>
	</div>
</body>
</html>
```

Payment_gateway.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
	pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
	<h2 class="title">Payment Successful</h2>
	<a href="index.html">Go to Home</a>
</body>
</html>
```

Style.css:

```css
@charset "ISO-8859-1";

* {
        box-sizing: border-box;
        margin: 0;
        padding: 0;
}

#heading{
        text-align:center;
}

a{
        text-decoration: none;
        padding: .5rem;
        margin:1rem;
        cursor: pointer;
        background-color: yellow;
        color:inherit;
        font-weight: bold;
}

.wrapper {
        display: flex;
        flex-direction: column;
        justify-content: center;
        align-items: center;
        gap: 1em;
        width: 100vw;
        height: 100vh;
}

.dashboard-wrapper{
        display:flex;
        justify-content: space-around;
        align-items: center;
        border:2px block thin;
        margin: 2em;
}

div>table{
        border-collapse: collapse;
        margin:10px;
}
tr > td,tr > th{
        padding:5px;
}
tr:hover{
        background-color: yellow;
}

.title {
        font-size: 2em;
}

form>label {
        font-size: 1.2rem;
        margin-top: 5px;
        margin-bottom: 5px;
}

form>input:not(:last-child), form>select {
```

```
        font-size: 1.1rem;
        border-bottom: 2px thin;
        border-right: none;
        border-left: none;
        border-top: none;
        padding: 5px;
        outline: none;
        margin-top: 5px;
        margin-bottom: 5px;
}

input[type="submit" i] {
        width: 100%;
        padding: 5px;
        margin-top: 5px;
        font-size: 1rem;
}

#admin-info{
        margin: 10px;
}
#admin-info>p{
        font-size: 1.2rem;
        margin-left: 2rem;
}

.wrapper>.sub-title {
        color: red;
}
```

Models or Entities:

Admin.java:

```
@Entity
public class Admin {

        @Id
        private String Email;
        private String Name;
        private String password;

        public Admin() {
        }

        public Admin(String email, String name, String password) {
                Email = email;
                Name = name;
                this.password = password;
        }


        public String getUsername() {
                return Email;
        }

        public void setUsername(String username) {
                Email = username;
        }

        public String getName() {
```

```
                return Name;
        }

        public void setName(String name) {
                Name = name;
        }

        public String getPassword() {
                return password;
        }

        public void setPassword(String password) {
                this.password = password;
        }

        @Override
        public String toString() {
                return "Admin [Email=" + Email + ", Name=" + Name + ", password=" +
password + "]";
        }
}
```

Flight.java:

```
@Entity
public class Place {
        @Id
        @GeneratedValue(strategy = GenerationType.AUTO)
        private int id;
        private String name;

        public Place() {
        }

        public Place(String name) {
                this.name = name;
        }

        public int getId() {
                return id;
        }

        public void setId(int id) {
                this.id = id;
        }

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }

        @Override
        public String toString() {
                return "Place [id=" + id + ", name=" + name + "]";
        }

}
```

Flight.java:

```
@Entity
public class Flight {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private int flight_id;
	private String flightname;
	private int numofpassengers;

	@ManyToOne
	private Place source;
	@ManyToOne
	private Place destination;

	public Flight() {
	};

	public Flight(String flightname, int numofpassengers, Place source, Place
destination) {
		this.flightname = flightname;
		this.numofpassengers = numofpassengers;
		this.source = source;
		this.destination = destination;
	}

	public int getFlightid() {
		return flight_id;
	}

	public void setFlightid(int flightid) {
		this.flight_id = flightid;
	}

	public String getFlightname() {
		return flightname;
	}

	public void setFlightname(String flightname) {
		this.flightname = flightname;
	}

	public int getNumofpassengers() {
		return numofpassengers;
	}

	public void setNumofpassengers(int numofpassengers) {
		this.numofpassengers = numofpassengers;
	}

	public Place getSource() {
		return source;
	}

	public void setSource(Place source) {
		this.source = source;
	}

	public Place getDestination() {
		return destination;
	}
```

```
	public void setDestination(Place destination) {
		this.destination = destination;
	}

	@Override
	public String toString() {
		return "Flight [flight_id=" + flight_id + ", flightname=" + flightname
+ ", numofpassengers=" + numofpassengers
				+ ", source=" + source + ", destination=" + destination
+ "]";
	}

}
```

OpenFlights.java:

```
@Entity
@IdClass(OpenFlightsPK.class)
public class OpenFlights {

	@Id
	@ManyToOne
	@JoinColumn(name = "flight_id")
	private Flight flight;

	@Id
	private Date date;

	public OpenFlights() {
	}

	public OpenFlights(Flight flight, Date date) {
		this.flight = flight;
		this.date = date;
	}

	public Flight getFlight() {
		return flight;
	}

	public void setFlight(Flight flight) {
		this.flight = flight;
	}

	public Date getDate() {
		return date;
	}

	public void setDate(Date date) {
		this.date = date;
	}

}

class OpenFlightsPK implements Serializable {

	private static final long serialVersionUID = 1L;

	private Flight flight;
	private Date date;
```

```
	public OpenFlightsPK() {
	}

	public OpenFlightsPK(Flight flight, Date date) {
		this.flight = flight;
		this.date = date;
	}

	public Flight getFlight() {
		return flight;
	}

	public void setFlight(Flight flight) {
		this.flight = flight;
	}

	public Date getDate() {
		return date;
	}

	public void setDate(Date date) {
		this.date = date;
	}

	@Override
	public String toString() {
		return "OpenFlightsPK [flight=" + flight + ", date=" + date + "]";
	}

}
```

Passenger.java:

```
@Entity
public class Passenger {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	private int Id;

	private String name;
	private int age;

	@ManyToOne
	@JoinColumn(name = "flight_id", referencedColumnName = "flight_id")
	private Flight flight;
	private Date date;

	public Passenger() {
	}

	public Passenger(String name, int age, Flight flight, Date date) {
		this.name = name;
		this.age = age;
		this.flight = flight;
		this.date = date;
	}

	public int getId() {
		return Id;
```

```
	}

	public void setId(int id) {
		Id = id;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public int getAge() {
		return age;
	}

	public void setAge(int age) {
		this.age = age;
	}

	public Flight getFlight() {
		return flight;
	}

	public void setFlight(Flight flight) {
		this.flight = flight;
	}

	public Date getDate() {
		return date;
	}

	public void setDate(Date date) {
		this.date = date;
	}
}
```

Utils or DAO's:

HibernateUtil.java:

```
public class HibernateUtil {
	private static final SessionFactory SESSION_FACTORY = buildSessionFactory();

	private static SessionFactory buildSessionFactory() {
		StandardServiceRegistry registry = new
StandardServiceRegistryBuilder().configure("hibernate.config.xml").build();
		@SuppressWarnings("deprecation")
		Metadata metadata = new
MetadataSources(registry).getMetadataBuilder(registry).build();
		return metadata.buildSessionFactory();
	}

	public static SessionFactory getSessionFactory() {
		return SESSION_FACTORY;
	}
}
```

AdminUtil.java:

```
public class AdminUtil {

	public static void addAdmin(String username, String name, String password) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		session.beginTransaction();

		Admin admin = new Admin(username, name, password);

		session.save(admin);
		session.getTransaction().commit();
		session.close();
	}

	public static Admin getAdmin(String email,String password) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		String hql = "FROM Admin where email = :email and password =
:password";
		Query query = session.createQuery(hql,Admin.class);
		query.setParameter("email", email);
		query.setParameter("password", password);
		List<Admin> admins = query.getResultList();
		if(admins.size()<1) {
			return null;
		}
		Admin admin = admins.get(0);
		return admin;
	}

	public static Admin getAdmin(String email) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		String hql = "FROM Admin where email = :email";
		Query query = session.createQuery(hql,Admin.class);
		query.setParameter("email", email);
		List<Admin> admins = query.getResultList();
		if(admins.size()<1) {
			return null;
		}
		Admin admin = admins.get(0);
		return admin;
	}

}
```

PlaceUtil.java:

```
public class PlaceUtil {
	public static void addplace(String name) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		session.beginTransaction();

		Place place = new Place(name);

		session.save(place);
		session.getTransaction().commit();
		session.close();
	}

	public static List<Place> getPlace() {
		Session session = HibernateUtil.getSessionFactory().openSession();
```

```
                String hql = "FROM Place" ;
                List<Place> places= session.createQuery(hql).getResultList();
                session.close();
                return places;
        }

        public static Place getPlace(int id) {
                Session session = HibernateUtil.getSessionFactory().openSession();
                Place place = null;
                place = session.get(Place.class, id);
                return place;
        }
}
```

FlightUtil.java:

```
public class FlightUtil {
        public static void addFlight(String flightname, int numofpassengers, Place
source, Place destination) {
                Session session = HibernateUtil.getSessionFactory().openSession();
                session.beginTransaction();

                Flight flight = new Flight(flightname, numofpassengers, source,
destination);

                session.save(flight);
                session.getTransaction().commit();
                session.close();
        }

        public static List<Flight> getFlight() {
                Session session = HibernateUtil.getSessionFactory().openSession();
                List<Flight> flights = session.createQuery("from
Flight").getResultList();
                session.close();
                return flights;
        }

        public static Flight getFlight(int id) {
                Session session = HibernateUtil.getSessionFactory().openSession();
                Flight flight = null;
                flight = session.get(Flight.class, id);
                session.close();
                return flight;
        }
}

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

public class HibernateUtil {
        private static final SessionFactory SESSION_FACTORY = buildSessionFactory();

        private static SessionFactory buildSessionFactory() {
                StandardServiceRegistry registry = new
StandardServiceRegistryBuilder().configure("hibernate.config.xml").build();
                @SuppressWarnings("deprecation")
                Metadata metadata = new
MetadataSources(registry).getMetadataBuilder(registry).build();
                return metadata.buildSessionFactory();
        }
```

```
	public static SessionFactory getSessionFactory() {
		return SESSION_FACTORY;
	}
}
```

AllotedFlightUtil.java:

```
public class AllotedFlightUtil {

	public static void addDateToFlight(Flight flight, Date date) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		session.beginTransaction();

		OpenFlights of = new OpenFlights(flight, date);

		session.save(of);
		session.getTransaction().commit();
		session.close();
	}

	public static List<OpenFlights> getAllocations() {
		Session session = HibernateUtil.getSessionFactory().openSession();
		List<OpenFlights> openFlights = session.createQuery("from
OpenFlights").getResultList();
		session.close();
		return openFlights;
	}

	public static List<Flight> getFlights(Date date,Integer sourceId,Integer
destinationId) {
		Session session = HibernateUtil.getSessionFactory().openSession();
		String hql = "select f from Flight f join OpenFlights o on f.flight_id
= o.flight.flight_id where o.date = :date and f.source.id=:sourceId and
f.destination.id=:destinationId";
		Query query = session.createQuery(hql);
		query.setParameter("date", date);
		query.setParameter("sourceId", sourceId);
		query.setParameter("destinationId", destinationId);
		List<Flight> of = query.getResultList();
		session.close();
		return of;
	}
}
```

PassesngerUtil.java:

```
public class PassengetUtil {

	public static void addPassenger(String name, int age, Flight flight,Date date)
{
		Session session = HibernateUtil.getSessionFactory().openSession();
		session.beginTransaction();

		Passenger passenger = new Passenger(name, age, flight,date);

		session.save(passenger);
		session.getTransaction().commit();
		session.close();
	}
```

```
	public static List<Passenger> getPassengers() {
		Session session = HibernateUtil.getSessionFactory().openSession();
		List<Passenger> passengers = session.createQuery("from
Passenger").getResultList();
		session.close();
		return passengers;
	}
}
```

Servlets:

AdminVerification.java:

```
@WebServlet("/AdminVerification")
public class AdminVerification extends HttpServlet {
	private static final long serialVersionUID = 1L;

	public AdminVerification() {
		super();
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		PrintWriter out = response.getWriter();
		response.setContentType("text/html");
		out.println("<h2>URL Not Found</h2>");
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		String email = request.getParameter("email");
		String password = request.getParameter("password");
		Admin admin = AdminUtil.getAdmin(email, password);
		RequestDispatcher dispatcher = null;
		PrintWriter out = response.getWriter();
		response.setContentType("text/html");
		if (admin == null) {
			dispatcher = request.getRequestDispatcher("admin_login.html");
			out.println("<p>Invalid Credentials</p>");
			dispatcher.include(request, response);
		} else {
			HttpSession session = request.getSession();
			session.setAttribute("email", admin.getUsername());
			response.sendRedirect("dashboard");
		}
	}
}
```

AdminDashboard.java:

```
@WebServlet("/dashboard")
public class AdminDashboard extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		HttpSession session = request.getSession(false);
		RequestDispatcher dispatcher = null;
		if (session != null) {
```

```
                        String email = (String) session.getAttribute("email");
                        Admin admin = AdminUtil.getAdmin(email);
                        List<Place> places = PlaceUtil.getPlace();
                        List<Flight> flights = FlightUtil.getFlight();
                        List<OpenFlights> ofs = AllotedFlightUtil.getAllocations();
                        List<Passenger> passengers = PassengetUtil.getPassengers();
                        for(Passenger pas:passengers) {
                                System.out.println(pas.getDate());
                        }
                        request.setAttribute("passengers", passengers);
                        request.setAttribute("places", places);
                        request.setAttribute("admin", admin);
                        request.setAttribute("flights", flights);
                        request.setAttribute("ofs", ofs);
                        dispatcher = request.getRequestDispatcher("dashboard.jsp");
                        dispatcher.forward(request, response);
                } else {
                        PrintWriter out = response.getWriter();
                        out.println("<p>Invalid Credentials</p>");
                        dispatcher = request.getRequestDispatcher("admin_login.html");
                        dispatcher.include(request, response);
                }

        }

        protected void doPost(HttpServletRequest request, HttpServletResponse
response)
                        throws ServletException, IOException {
                PrintWriter out = response.getWriter();
                response.setContentType("text/html");
                out.println("<h2>URL Not Found</h2>");
        }

}
```

AddPlace.java:

```
@WebServlet("/AddPlace")
public class AddPlace extends HttpServlet {
        private static final long serialVersionUID = 1L;


    public AddPlace() {
        super();
    }


        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                PrintWriter out = response.getWriter();
                response.setContentType("text/html");
                out.println("<h2>URL Not Found</h2>");
        }

        protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                String place = request.getParameter("place");
                PlaceUtil.addplace(place);
                response.sendRedirect("dashboard");
        }
}
```

AddFlight.java

```
@WebServlet("/AddFlight")
public class AddFlight extends HttpServlet {
	private static final long serialVersionUID = 1L;

	public AddFlight() {
		super();
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		PrintWriter out = response.getWriter();
		response.setContentType("text/html");
		out.println("<h2>URL Not Found</h2>");
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		String flight_name = request.getParameter("flightName");
		int numOfPassenger =
Integer.parseInt(request.getParameter("noOfPassengers"));
		int source_id = Integer.parseInt(request.getParameter("source"));
		int destination_id =
Integer.parseInt(request.getParameter("destination"));

		Place source = PlaceUtil.getPlace(source_id);
		Place destionation = PlaceUtil.getPlace(destination_id);
		FlightUtil.addFlight(flight_name, numOfPassenger, source,
destionation);
		response.sendRedirect("dashboard");
	}

}
```

AddDateofFlight.java:

```
@WebServlet("/AddDateOfFlight")
public class AddDateOfFlight extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		PrintWriter out = response.getWriter();
		response.setContentType("text/html");
		out.println("<h2>URL Not Found</h2>");
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		int flightId = Integer.parseInt(request.getParameter("flight"));
		String flightDate = request.getParameter("flightDate");
		Flight flight = FlightUtil.getFlight(flightId);
		SimpleDateFormat dateformat = new SimpleDateFormat("yyyy-MM-dd");
		Date date;
		try {
			date = dateformat.parse(flightDate);
			AllotedFlightUtil.addDateToFlight(flight, date);
			System.out.println(date);
			response.sendRedirect("dashboard");
```

```
                } catch (ParseException e) {
                        e.printStackTrace();
                }
        }
}
```

SearchFlight.java;

```
@WebServlet("/Searchflights")
public class Search_flights extends HttpServlet {
        private static final long serialVersionUID = 1L;



        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                List<Place> places = PlaceUtil.getPlace();
                request.setAttribute("places", places);
                request.getRequestDispatcher("search_flight.jsp").forward(request,
response);
        }

        protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                doGet(request, response);
        }
}
```

BookFlight.java:

```
@WebServlet("/BookFlight")
public class BookFlight extends HttpServlet {
        private static final long serialVersionUID = 1L;

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                String flightDate = request.getParameter("flightDate");
                int sourceId = Integer.parseInt(request.getParameter("source"));
                int destinationId =
Integer.parseInt(request.getParameter("destination"));
                SimpleDateFormat dateformat = new SimpleDateFormat("yyyy-MM-dd");
                Date date;
                try {
                        date = dateformat.parse(flightDate);
                        List<Flight> flights = AllotedFlightUtil.getFlights(date,
sourceId, destinationId);
                        request.setAttribute("flights", flights);
                        request.setAttribute("NoOfFlights", flights.size());
                        request.setAttribute("date", date);
                } catch (ParseException e) {
                        e.printStackTrace();
                }
                request.getRequestDispatcher("bookFlight.jsp").forward(request,
response);
        }

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

```
	}

}
```

MakePayment.java:

```
@WebServlet("/MakePayment")
public class MakePayment extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		response.getWriter().append("Served at:
").append(request.getContextPath());
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response)
			throws ServletException, IOException {
		int flightId = Integer.parseInt(request.getParameter("flightId"));
		String flightdate = request.getParameter("flightdate");
		String[] names = request.getParameterValues("names");
		String[] ages = request.getParameterValues("ages");
		SimpleDateFormat dateformat = new SimpleDateFormat("yyyy-MM-dd");
		try {
			Date date = dateformat.parse(flightdate);
			Flight flight = FlightUtil.getFlight(flightId);
			synchronized (names) {
				for (int i = 0; i < names.length; i++) {
					PassengetUtil.addPassenger(names[i],
Integer.parseInt(ages[i]), flight,date);
				}
			}
		} catch (ParseException e) {
			e.printStackTrace();
		}
		HttpSession httpSession = request.getSession(false);
		httpSession.invalidate();
		response.sendRedirect("paymentGateway.jsp");
	}
}
```

Filters:

SessionVerification.java

```
@WebFilter(urlPatterns = { "/AddPlace"
,"/AddFlight","","/AddDateOfFlight","/dashboard"})
public class SessionVerification extends HttpFilter implements Filter {

	private static final long serialVersionUID = 1L;

	public SessionVerification() {
		super();
	}

	public void doFilter(ServletRequest request, ServletResponse response,
FilterChain chain)
			throws IOException, ServletException {

		HttpServletRequest httpRequest = (HttpServletRequest) request;
```

```
                HttpSession session = httpRequest.getSession(false);
                RequestDispatcher dispatcher ;
                PrintWriter out = response.getWriter();
                if (session == null) {
                        dispatcher = request.getRequestDispatcher("admin_login.html");
                        out.println("<p>Please login again</p>");
                        dispatcher.include(request, response);
                }
                chain.doFilter(request, response);
        }

        public void init(FilterConfig fConfig) throws ServletException {
                System.out.println("Initialization");
        }

}
```