Project and Developer Details

Project Name: FlyAway

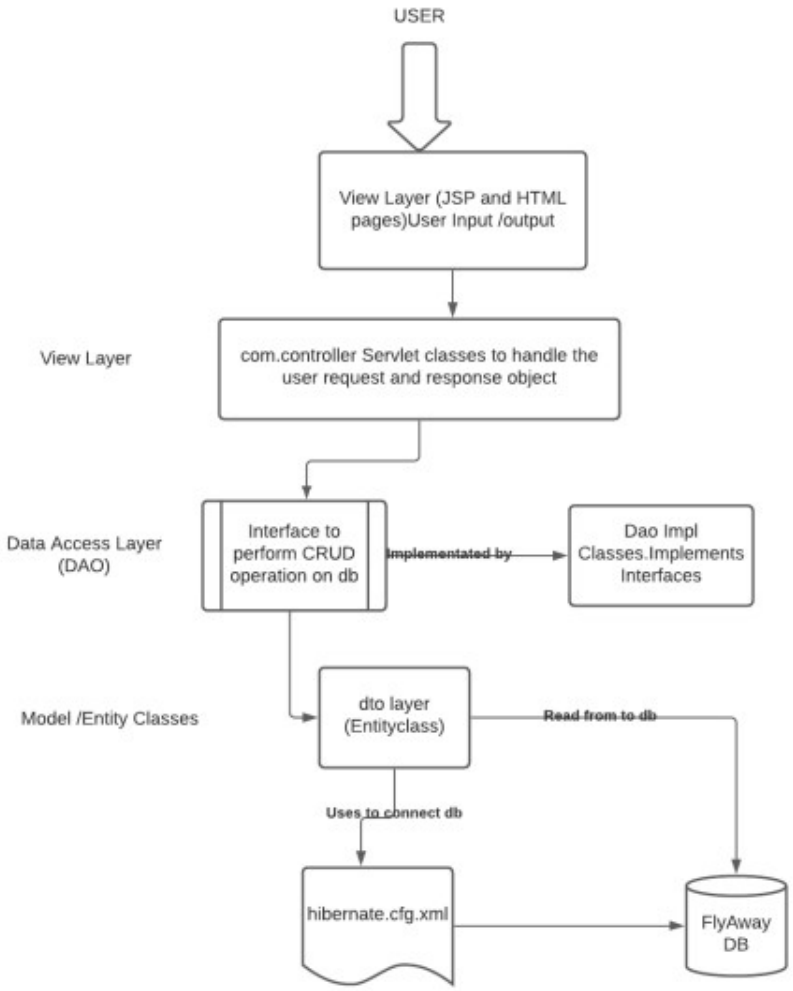
Developer Name: Amal Aldawas

Project Details:

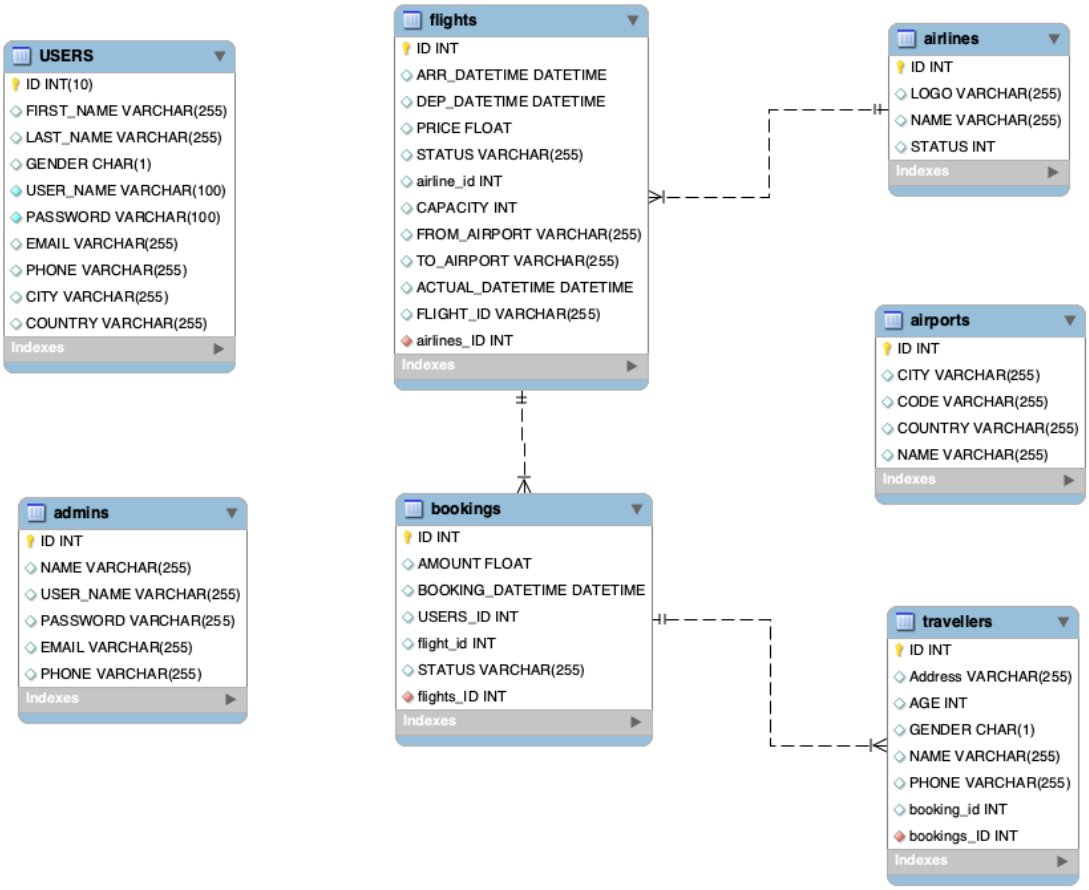
Steps to use:

1. Open in IDE with Tomcat and SQL
2. Create new database called "flyawaydb"
3. Load in tables/data with information from db with setUpDB.sql file
4. Right Click Project Name and Run on Serve

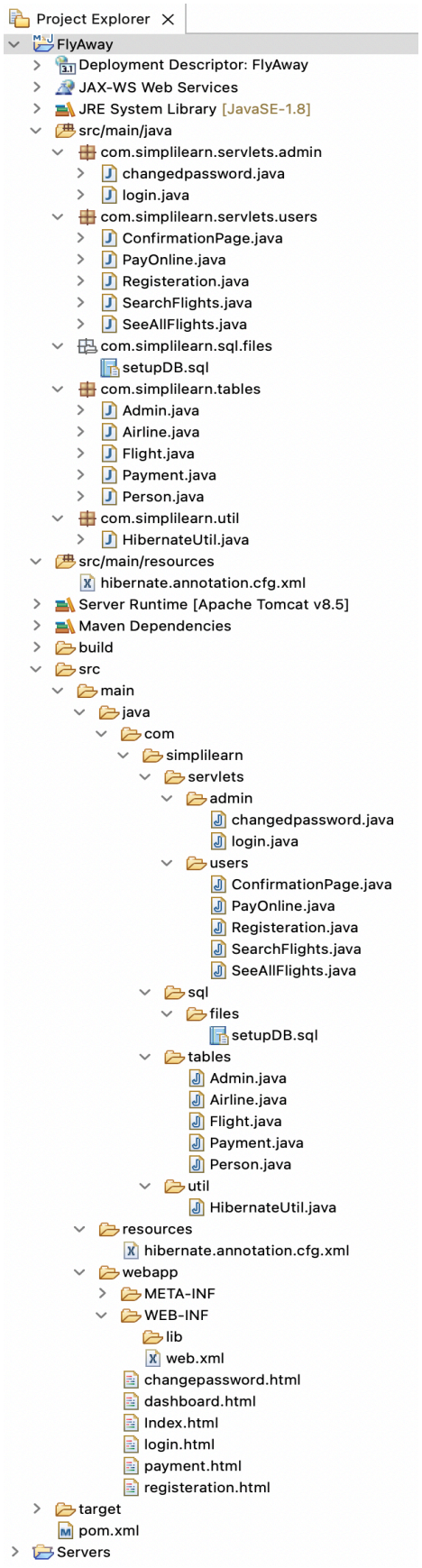
Flow of the Application



Database table design and relationship:



Folder Structure:



Project code:

pom.xml

FlyAway/pom.xml

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.simplilearn</groupId>
  <artifactId>FlyAway</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
     <!-- hibernate-core -->
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>5.4.1.Final</version>
    </dependency>

    <!-- mysql-connector-java -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.21</version>
    </dependency>

    <dependency>
     <groupId>javax.servlet</groupId>
     <artifactId>servlet-api</artifactId>
     <version>2.5</version>
    </dependency>

  </dependencies>
  <build>
    <finalName>flyaway</finalName>
     <sourceDirectory>src/main/java</sourceDirectory>
        <plugins>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.5.1</version>                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
  </build>


</project>
```

hibernate.annotation.cfg.xml

hibernate.annotation.cfg.xml

```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="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/flyawaydb</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>
        <property name="hibernate.connection.pool_size">1</property>
        <property name="hibernate.current_session_context_class">thread</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</property>
        <property name="hibernate.id.new_generator_mappings">false</property>

        <mapping class="com.simplilearn.tables.Admin" />
        <mapping class="com.simplilearn.tables.Airline" />
        <mapping class="com.simplilearn.tables.Flight" />
        <mapping class="com.simplilearn.tables.Payment" />
        <mapping class="com.simplilearn.tables.Person" />
    </session-factory>
</hibernate-configuration>
```

Index.html:

Index.html

FlyAway/src/main/webapp/Index.html

```html
<html>
<head>
 <meta charset="UTF-8">
 <title>Welcome To Fly Away Ticket Booking Portal</title>
 </head>
<body>
     <a href="login.html">Admin Login</a>
     <hr>
     <h1>Search For Flights</h1>
     <form name="searchForFlights" method="GET" action="search-flights">
         Flying From : <input type="text" name="source"> <br/>
         Flying To : <input type="text" name="destination"> <br/>
         Your Max Price : <input type="number" name="price"> <br/>
         Flights On or After Date : <input type="date" name="date"> <br/>
         <br/>
         <input type="submit" value="Search for flights!">
     </form>
     <form name ="seeAllFlights" method="GET" action="see-all-flights">
         <input type="submit" value="See All Flights">
     </form>
 </body>
 </html>
```

changepassword.html:

changepassword.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Admin Login Page</title>
</head>
<body>
    <a href="dashboard.html">Dashboard</a>

    <h1>Change Password</h1>
    <form name="change password" method="POST" action="change-password">
        New Password : <input type="password" name="password"> <br/>
        Confirm New Password: <input type="password" name="confirmpassword"> <br/>
        <br/>
        <input type="submit" value="Change password">
    </form>

</body>
</html>
```

dashboard.html:

dashboard.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Admin Dashboard</title>
</head>
<body>
    <a href="index.html">Home</a> |
    <a href="index.html">Logout</a> |
    <a href="changepassword.html">Change Password</a> |
    <a href="list-airlines">Airline Master View </a> |
    <a href="list-source-destination">Source/Destination Master View</a> |
    <a href="list-flights">Flight Master View</a><br/>
    <hr>
</body>
</html>
```

login.html:

login.html

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Admin Login Page</title>
</head>
<body>
    <a href="index.html">Home</a>

    <h1>Admin Login Page</h1>
    <form name="loginForm" method="POST" action="login">
        Username: <input type="text" name="username"> <br/>
        Password : <input type="password" name="password"> <br/>
        <br/>
        <input type="submit" value="Login">
    </form>

</body>
</html>
```

payment.html:

payment.html

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Payment</title>
</head>
<body>
    <h1>Enter Payment Details</h1>
    <form name="paymentDetails" method="POST" action="pay-online">
        First Name On Card : <input type="text" name="firstname" placeholder="John" required> <br/>
        Last Name On Card : <input type="text" name="lastname" placeholder="Smith" required> <br/>
        Card Number: <input type="text" name="cardnumber" placeholder="1234 1234 4321 9123" required> <br/>
        Expiration (yyyy-MM-dd) : <input type="date" name="expiration" placeholder="2027-10-31"  required> <br/>
        Security Code : <input type="number" name="securitycode" placeholder="987" required> <br/>
        <br/>
        <input type="submit" value="Pay Ticket">
    </form>

</body>
</html>
```

registeration.html:

registeration.html

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Flight Registration Page</title>
</head>
<body>
    <h1>Flight Registration Page</h1>
    <form name="flightRegistraton" method="POST" action="registeration">
        Flight ID : <input type="number" name="flightid" placeholder="12" required> <br/>
        First Name : <input type="text" name="firstname" placeholder="John" required> <br/>
        Last Name : <input type="text" name="lastname" placeholder="Doe" required> <br/>
        Email : <input type="email" name="email" placeholder="johndoe@gmail.com" required> <br/>
        Birthday (yyyy-MM-dd) : <input type="date" name="birthday" placeholder="1990-04-02" required> <br/>
        <br/>
        <input type="submit" value="Submit Flight Registration">
    </form>

</body>
</html>
```

com.simplilearn.servlets.admin

changepassword.java

*changedpassword.java

```java
package com.simplilearn.servlets.admin;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *   Servlet implementation class changedpassword
 */
@WebServlet("/change-password")
public class changedpassword extends HttpServlet {      private static final long serialVersionUID = 1L;
    /**
 *   @see HttpServlet#HttpServlet()
     */
    public changedpassword() {         super();
        // TODO Auto-generated constructor stub
    }


    /**
 *   @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
 *   @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        // TODO Auto-generated method stub
        PrintWriter out = resp.getWriter();
        String pass = req.getParameter("password");         if (!login.isLoggedIn){
            out.println("You must login first");
        }
        else if (pass.equals("")){
            out.println("Password can't be empty");
        }
        else if (login.isLoggedIn && !pass.equals("")){
            login.password = pass;
            out.println("Password changed. New Password is "+login.password);
            try {
                Thread.sleep(1000);
            }
            catch (InterruptedException e) {
                // TODO Auto-generated catch block e.printStackTrace();
            }
        }
        else {
            out.println("something went wrong");
        }
        try {
            req.getRequestDispatcher("passwordverified.html").forward(req, resp);
        }
        catch (ServletException e) {
            // TODO Auto-generated catch block e.printStackTrace();
        }
        catch (IOException e) {
            // TODO Auto-generated catch block e.printStackTrace();
        }
        out.close();
    }
}
```

login.java

*login.java

```java
package com.simplilearn.servlets.admin;

import java.io.IOException; import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *  Servlet implementation class login
 */

@WebServlet("/login")
public class login extends HttpServlet {
    /**
 *      */
    private static final long serialVersionUID = 1L;    public static boolean isLoggedIn = false;     public static String password = "Abhi18@ayaan19";     public static String email = "kumar.abhishek7885@gmail.com";

    @Override
    public void doPost (HttpServletRequest req, HttpServletResponse resp) throws IOException {        PrintWriter out = resp.getWriter();

        String email = req.getParameter("username");        String pass = req.getParameter("password");

        if (email.equals(login.email) && pass.equals(login.password)){            isLoggedIn = true;
            out.println("You have LoggedIn");

            try {
                req.getRequestDispatcher("dashboard.html").forward(req, resp);
            } catch (ServletException e) {
                // TODO Auto-generated catch block                e.printStackTrace();
            } catch (IOException e) {
                // TODO Auto-generated catch block                e.printStackTrace();
            }
                    }           else {
            isLoggedIn = false;            out.println("Login Failed : Incorrect email or Password");
                    }
        out.close();
    }
}
```

com.simplilearn.servlets.users

ConfirmationPage.java:

ConfirmationPage.java

```java
package com.simplilearn.servlets.users;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.hibernate.Session;
import org.hibernate.SessionFactory;

import com.simplilearn.tables.Flight;
import com.simplilearn.util.HibernateUtil;

/**
 * Servlet implementation class ConfirmationPage
 */
@WebServlet("/confirmation-page")
public class ConfirmationPage extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ConfirmationPage() {
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // print writer
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");

        HttpSession httpsession = request.getSession();
        String flightid = (String) httpsession.getAttribute("flightid");
        String firstName = (String) httpsession.getAttribute("firstname");
        int flightIdConverted = Integer.parseInt(flightid);
        //open a connection
        try {
            SessionFactory sFactory = HibernateUtil.buildSessionFactory();
            Session session = sFactory.openSession();
            session.beginTransaction();

            String query = "select f from Flight f" +
                    " inner join Airline a" +
                    " on f.airline=a.id" +
                    " and f.numberOfSeats > 0" +
                    " and f.id = " + flightIdConverted;
            @SuppressWarnings("unchecked")
            List<Flight> flights = session.createQuery(query).list();

            if (flights.size() > 0) {
            out.print("<a href=\"index.html\">Home</a><br/>");
            out.println("<h1>Congratulations! You're all set, "  + firstName + "!</h1><br/><br/>");
            out.println("<h1>List of Confirmed Flights for "  + firstName + "</h1><hr>");
            out.println("<style> table,th,td { border : 1px solid black ; padding :15px;} </style>");
            out.println("<table>");
            out.println("<tr>");
                out.println("<th>"); out.println("Flight ID"); out.println("</th>");
                out.println("<th>"); out.println("Airline"); out.println("</th>");
                out.println("<th>"); out.println("Number Of Seats"); out.println("</th>");
                out.println("<th>"); out.println("Source"); out.println("</th>");
                out.println("<th>"); out.println("Desintation"); out.println("</th>");
                out.println("<th>"); out.println("Departure Date"); out.println("</th>");
                out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
                out.println("<th>"); out.println("Price"); out.println("</th>");
            out.println("</tr>");
            for(Flight i:flights) {
                if (i.getNumberOfSeats() > 0) {
                    out.println("<tr>");
                        out.println("<td>"); out.println(i.getId()); out.println("</td>");
                        out.println("<td>"); out.println(i.getAirline().getAirline().toString()); out.println("</td>");
                        out.println("<td>"); out.println(i.getNumberOfSeats()); out.println("</td>");
                        out.println("<td>"); out.println(i.getSource()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDestination()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDateOfDeparture()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDateOfArrival()); out.println("</td>");
                        out.println("<td>"); out.println(i.getPrice()); out.println("</td>");
                    out.println("</tr>");
                }
            }
            out.println("</table>");
            } else {
                out.print("<a href=\"index.html\">Home</a>");
                out.print("<h1>Sorry You don't have any are no flights currently!<h1>");
            }

            session.close();

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            out.close();
        }
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    }

}
```

PayOnline.java:

PayOnline.java

```java
package com.simplilearn.servlets.users;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import java.util.regex.Pattern;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.query.Query;

import com.simplilearn.tables.Flight;
import com.simplilearn.tables.Payment;
import com.simplilearn.tables.Person;
import com.simplilearn.util.HibernateUtil;

/**
 * Servlet implementation class PayOnline
 */
@WebServlet("/pay-online")
public class PayOnline extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public PayOnline() {
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.sendRedirect("payment.html");
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    @SuppressWarnings({ "unchecked", "rawtypes" })
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");

        HttpSession httpsession = request.getSession(false);

        String personflightid = (String) httpsession.getAttribute("flightid");
        String personfirstname = (String) httpsession.getAttribute("firstname");
        String personlastname = (String) httpsession.getAttribute("lastname");
        String personemail = (String) httpsession.getAttribute("email");
        String personbirthday = (String) httpsession.getAttribute("birthday");
        long millis = System.currentTimeMillis();
        Date timeBooked = new Date(millis);
        Date dateSqlPerson = null;
        try {
            if ((dateSqlPerson=Date.valueOf(personbirthday)) == null) {
                out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
                out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
            };

        } catch (Exception e1) {
            out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
            out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
        }
```

PayOnline.java

```
        String fName = request.getParameter("firstname");
        String lName = request.getParameter("lastname");
        String cardNumber = request.getParameter("cardnumber").replaceAll("\\s", "");
        System.out.println(cardNumber);
        String expiration = request.getParameter("expiration");
        int securitycode = Integer.parseInt(request.getParameter("securitycode"));
        Date dateSqlPayment = null;
        try {
            dateSqlPayment=Date.valueOf(expiration);

        } catch (Exception e1) {
            out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
            out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
        }


        if (!Pattern.matches("\\d{15,16}", cardNumber)) {
            out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
            out.print("Make sure you are entering your card number in correctly");

        } else if (!Pattern.matches("\\d{3,4}", Integer.toString(securitycode))) {
            out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
            out.print("Make sure you are entering your security code in correctly");
        } else if (dateSqlPayment.before(timeBooked)) {
            out.println("<h1>You must ensure your card is not expired<h1>");
            out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
        }
        else {


        //step 1. Confirm Fields Match Criteria
            //Else redirect to payment
        //step 2. insert into person table
        //open a connection
        try {

            SessionFactory sFactory = HibernateUtil.buildSessionFactory();
            Session session = sFactory.openSession();
            session.beginTransaction();

            int personflightidConverted = Integer.parseInt(personflightid);

            String query1 = " from Flight where id = " + personflightidConverted;
            List<Flight> list = (List<Flight>) session.createQuery(query1).list();
            Set<Flight> flights = new HashSet<Flight>(list);



            Person person = new Person();
            person.setFirstName(personfirstname);
            person.setLastName(personlastname);
            person.setEmail(personemail);
            person.setBirthday(dateSqlPerson);
            person.setTimeBooked(timeBooked);
            person.setFlights(flights);

            Payment payment = new Payment();
            payment.setFirstNameOnCard(fName);
            payment.setLastNameOnCard(lName);
            payment.setCardNumber(cardNumber);
            payment.setExpiration(dateSqlPayment);
            payment.setSecurityCode(securitycode);
            payment.setPerson(person);


            session.save(person);
            session.save(payment);

            String query = "select numberOfSeats from Flight f" +
                    " where id = " +  personflightidConverted;
            int resultFromQ1 = (int) session.createQuery(query).getSingleResult();
            Query q3=session.createQuery("update Flight set numberOfSeats=:n where id=:i");
            q3.setParameter("n",(resultFromQ1 - 1));
            q3.setParameter("i",personflightidConverted);
```

```
            int status=q3.executeUpdate();
            System.out.println(status);

            session.getTransaction().commit();
            session.close();
            response.sendRedirect("confirmation-page");
            out.close();
          } catch (Exception e) {
              e.printStackTrace();
          }
          }
          //step 3. insert into payment table
          //step 4. reduce person count
          //step 5. Redirect to confirmation page

      }

}
```

Registeration.java:

Registeration.java

```java
package com.simplilearn.servlets.users;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;
import java.util.List;
import java.util.regex.Pattern;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.hibernate.Session;
import org.hibernate.SessionFactory;

import com.simplilearn.tables.Flight;
import com.simplilearn.util.HibernateUtil;

/**
 * Servlet implementation class Registration
 */
@WebServlet("/registeration")
public class Registeration extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public Registeration() {
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    @SuppressWarnings("unchecked")
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // print writer
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");

        HttpSession httpsession = request.getSession();
        String flightid = request.getParameter("flightid");
        String firstname = request.getParameter("firstname");
        String lastname = request.getParameter("lastname");
        String email = request.getParameter("email");
        String birthday = request.getParameter("birthday");

        request.setAttribute("firstname", firstname);
        Date dateSqlBirthday = null;
        try {
            dateSqlBirthday = Date.valueOf(birthday);
        } catch (Exception e1) {
            out.print("<a href = \"registeration.html\"> Re-Enter Information </a><br/>");
            out.print("Make sure you are entering your date code in correctly (yyyy-MM-dd)");
        }

        //email
        if (!(Pattern.matches("[A-Za-z0-9]+(.+)@(.+)", email))) {
            out.print("<a href = \"registeration.html\"> Re-enter Information </a><br/>");
            out.print("Make sure you are entering in a valid email address");
        } else if (dateSqlBirthday==null){
            out.print("<a href = \"registeration.html\"> Re-enter Information </a><br/>");
            out.print("Make sure you are entering in a valid date format (yyyy-MM-dd)");
        } else {
        httpsession.setAttribute("flightid", flightid);
```

```java
            out.print("Make sure you are entering in a valid email address");
        } else if (dateSqlBirthday==null){
            out.print("<a href = \"registeration.html\"> Re-enter Information </a><br/>");
            out.print("Make sure you are entering in a valid date format (yyyy-MM-dd)");
        } else {
        httpsession.setAttribute("flightid", flightid);
        httpsession.setAttribute("firstname", firstname);
        httpsession.setAttribute("lastname", lastname);
        httpsession.setAttribute("email", email);
        httpsession.setAttribute("birthday", birthday);


        //open a connection
        try {
            SessionFactory sFactory = HibernateUtil.buildSessionFactory();
            Session session = sFactory.openSession();
            session.beginTransaction();

            String query = "select f from Flight f" +
                    " inner join Airline a" +
                    " on f.airline=a.id" +
                    " and f.numberOfSeats > 0";
                    if (flightid !=null && flightid.trim().length() > 0) {
                        try {
                            int flightIdParsed = Integer.parseInt(flightid);
                            String flightIdQuery = " and f.id =" + flightIdParsed;
                            query = query.concat(flightIdQuery);
                        } catch (Exception e){
                            out.println("<h1>You must enter a valid flight id<h1>");
                        }
                    }

            List<Flight> flights = session.createQuery(query).list();

            if (flights.size() > 0) {
            out.print("<a href=\"index.html\">Home</a>");
            out.println("<h1>Confirm Flight Details: </h1>");
            for(Flight i:flights) {
                if (i.getNumberOfSeats() > 0) {
                out.println("Flight ID: " + i.getId() + "<br/>");
                out.println("Airline: " + i.getAirline().getAirline().toString() + "<br/>");
                out.println("Number Of Seats: " + i.getNumberOfSeats() + "<br/>");
                out.println("Source " + i.getSource() + "<br/>");
                out.println("Desintation:" + i.getDestination() + "<br/>");
                out.println("Departure Date: "+i.getDateOfDeparture() + "<br/>");
                out.println("Arrival Date: " + i.getDateOfArrival() + "<br/>");
                out.println("<h1>Your Total is: $" + i.getPrice() + "<h1><br/>");
                out.println("<hr>");
                }
            }
            request.getRequestDispatcher("payment.html").include(request, response);
            } else {
                out.print("<a href=\"index.html\">Home</a>");
                out.print("<h1>You must enter a valid flight id!<h1>");
            }
            session.close();
            out.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        }
        }

}
```

SearchFlights.java:

SearchFlights.java

```java
package com.simplilearn.servlets.users;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Date;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.hibernate.Session;
import org.hibernate.SessionFactory;

import com.simplilearn.tables.Flight;
import com.simplilearn.util.HibernateUtil;

/**
 * Servlet implementation class SearchFlights
 */
@WebServlet("/search-flights")
public class SearchFlights extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public SearchFlights() {
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    @SuppressWarnings({ "unchecked" })
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // print writer
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");

        String source = request.getParameter("source");
        String destination = request.getParameter("destination");
        String price = request.getParameter("price");
        String date = request.getParameter("date");

        //open a connection
        try {
            SessionFactory sFactory = HibernateUtil.buildSessionFactory();
            Session session = sFactory.openSession();
            session.beginTransaction();

            String query = "select f from Flight f" +
                    " inner join Airline a" +
                    " on f.airline=a.id" +
                    " and f.numberOfSeats > 0";
                    if (source !=null && source.trim().length() > 0) {
                        String sourceQuery = " and f.source = '" + source + "'";
                        query = query.concat(sourceQuery);
                    }
                    if (destination != null && destination.trim().length() > 0) {
                        String destinationQuery = " and f.destination = '" + destination + "'";
                        query = query.concat(destinationQuery);
                    }
                    if (price !=null && price.trim().length() > 0) {
                        try {
                            Double priceParsed = Double.parseDouble(price);
                            String priceQuery = " and f.price <= " + priceParsed;
                            query = query.concat(priceQuery);
                        } catch (Exception e){
                            out.println("<h1>You must enter a valid number for price<h1>");
                        }
                    }
                    if (date !=null && date.trim().length()> 0) {
                        try {
                            Date dateSql=Date.valueOf(date);
                            String dateQuery = " and f.dateOfDeparture >= '" + dateSql + "'";
                            query = query.concat(dateQuery);
                        } catch (Exception e1) {
                            out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
                        }
                    }
                    System.out.println(query);
            List<Flight> flights = session.createQuery(query).list();

            if (flights.size() > 0) {
            out.print("<a href=\"index.html\">Home</a>");
            out.println("<h1>List of all Flights: </h1>");
            out.println("<style> table,th,td { border : 1px solid black ; padding :15px;} </style>");
            out.println("<table>");
            out.println("<tr>");
                out.println("<th>"); out.println("Flight ID"); out.println("</th>");
                out.println("<th>"); out.println("Airline"); out.println("</th>");
                out.println("<th>"); out.println("Number Of Seats"); out.println("</th>");
                out.println("<th>"); out.println("Source"); out.println("</th>");
                out.println("<th>"); out.println("Desintation"); out.println("</th>");
                out.println("<th>"); out.println("Departure Date"); out.println("</th>");
                out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
                out.println("<th>"); out.println("Price"); out.println("</th>");
            out.println("</tr>");
            for(Flight i:flights) {
                if (i.getNumberOfSeats() > 0) {
                    out.println("<tr>");
                        out.println("<td>"); out.println(i.getId()); out.println("</td>");
                        out.println("<td>"); out.println(i.getAirline().getAirline().toString()); out.println("</td>");
                        out.println("<td>"); out.println(i.getNumberOfSeats()); out.println("</td>");
                        out.println("<td>"); out.println(i.getSource()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDestination()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDateOfDeparture()); out.println("</td>");
                        out.println("<td>"); out.println(i.getDateOfArrival()); out.println("</td>");
                        out.println("<td>"); out.println(i.getPrice()); out.println("</td>");
                    out.println("</tr>");
                }
            }
            out.println("</table>");
            request.getRequestDispatcher("registeration.html").include(request, response);
            } else {
                out.print("<a href=\"index.html\">Home</a>");
                out.print("<h1>Sorry there are no flights currently!<h1>");
            }

            session.close();
            out.close();

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }

}
```

SeeAllFlights.java:

SeeAllFlights.java

```java
package com.simplilearn.servlets.users;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.hibernate.Session;
import org.hibernate.SessionFactory;

import com.simplilearn.tables.Flight;
import com.simplilearn.util.HibernateUtil;

/**
 * Servlet implementation class SeeAllFlights
 */
@WebServlet("/see-all-flights")
public class SeeAllFlights extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public SeeAllFlights() {
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    @SuppressWarnings({ "unused", "unchecked" })
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // print writer
                PrintWriter out = response.getWriter();
                response.setContentType("text/html");

                String source = request.getParameter("source");
                String destination = request.getParameter("destination");
                String price = request.getParameter("price");
                String date = request.getParameter("date");

                //open a connection
                try {
                    SessionFactory sFactory = HibernateUtil.buildSessionFactory();
                    Session session = sFactory.openSession();
                    session.beginTransaction();

                    String query = "select f from Flight f" +
                            " inner join Airline a" +
                            " on f.airline=a.id" +
                            " and f.numberOfSeats > 0";

                    List<Flight> flights = session.createQuery(query).list();

                    if (flights.size() > 0) {
                    out.print("<a href=\"index.html\">Home</a>");
                    out.println("<h1>List of all Flights: </h1>");
                    out.println("<style> table,th,td { border : 1px solid black ; padding :15px;} </style>");
                    out.println("<table>");
                    out.println("<tr>");
                        out.println("<th>"); out.println("Flight ID"); out.println("</th>");
                        out.println("<th>"); out.println("Airline"); out.println("</th>");
                        out.println("<th>"); out.println("Number Of Seats"); out.println("</th>");
                        out.println("<th>"); out.println("Source"); out.println("</th>");
                        out.println("<th>"); out.println("Desintation"); out.println("</th>");
                        out.println("<th>"); out.println("Departure Date"); out.println("</th>");
                        out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
                        out.println("<th>"); out.println("Price"); out.println("</th>");
                    out.println("</tr>");
                    for(Flight i:flights) {
                        if (i.getNumberOfSeats() > 0) {
                            out.println("<tr>");
                                out.println("<td>"); out.println(i.getId()); out.println("</td>");
                                out.println("<td>"); out.println(i.getAirline().getAirline().toString()); out.println("</td>");
                                out.println("<td>"); out.println(i.getNumberOfSeats()); out.println("</td>");
                                out.println("<td>"); out.println(i.getSource()); out.println("</td>");
                                out.println("<td>"); out.println(i.getDestination()); out.println("</td>");
                                out.println("<td>"); out.println(i.getDateOfDeparture()); out.println("</td>");
                                out.println("<td>"); out.println(i.getDateOfArrival()); out.println("</td>");
                                out.println("<td>"); out.println(i.getPrice()); out.println("</td>");
                            out.println("</tr>");
                        }
                    }
                    out.println("</table>");
                    request.getRequestDispatcher("registeration.html").include(request, response);
                    } else {
                        out.print("<a href=\"index.html\">Home</a>");
                        out.print("<h1>Sorry there are no flights currently!<h1>");
                    }

                    session.close();
                    out.close();

                } catch (Exception e) {
                    e.printStackTrace();
                }
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }

}
```

com.simplilearn.tables

Admin.java:

Admin.java

```java
package com.simplilearn.tables;

import javax.persistence.Column;

@Entity
@Table(name="admin")
public class Admin {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="admin_id")
    private int id;

    @Column(name="first_name")
    private String firstName;

    @Column(name="last_name")
    private String lastName;

    @Column(name="email")
    private String email;

    @Column(name="password")
    private String password;

    public Admin() {}
    public Admin(String firstName, String lastName, String email, String password) {
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
        this.password = password;
    }

    public int getId() {
        return id;
    }

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

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPassword() {
        return password;
    }

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

Airline.java:

Airline.java

```java
package com.simplilearn.tables;

import java.util.Set;

@Entity
@Table(name="airline")
public class Airline {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="a_id")
    private int id;

    @Column(name="airline")
    private String airline;

    @OneToMany(mappedBy="airline")
    private Set <Flight> flights;

    public Airline() {}
    public Airline(String airline, Set<Flight> flights) {
        super();
        this.airline = airline;
        this.flights = flights;
    }



    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getAirline() {
        return airline;
    }
    public void setAirline(String airline) {
        this.airline = airline;
    }
    public Set<Flight> getFlights() {
        return flights;
    }
    public void setFlights(Set<Flight> flights) {
        this.flights = flights;
    }


}
```

Flight.java:

Flight.java

```java
package com.simplilearn.tables;

import java.sql.Date;

@Entity
@Table(name="flight")
public class Flight {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="f_id")
    private int id;

    @Column(name="number_of_seats")
    private int numberOfSeats;

    @Column(name="price")
    private double price;

    @Column(name="source")
    private String source;

    @Column(name="destination")
    private String destination;

    @Column(name="date_of_departure")
    private Date dateOfDeparture;

    @Column(name="date_of_arrival")
    private Date dateOfArrival;

    @ManyToOne()
    @JoinColumn(name="a_id", nullable=false)
    private Airline airline;

    @ManyToMany(mappedBy = "flights")
    Set<Person> persons;

    public Flight() {}

    public Flight(int numberOfSeats, double price, String source, String destination, Date dateOfDeparture,
            Date dateOfArrival, Airline airline, Set<Person> persons) {
        super();
        this.numberOfSeats = numberOfSeats;
        this.price = price;
        this.source = source;
        this.destination = destination;
        this.dateOfDeparture = dateOfDeparture;
        this.dateOfArrival = dateOfArrival;
        this.airline = airline;
        this.persons = persons;
    }

    public Set<Person> getPersons() {
        return persons;
    }

    public void setPersons(Set<Person> persons) {
        this.persons = persons;
    }


    public int getId() {
        return id;
    }


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


    public int getNumberOfSeats() {
        return numberOfSeats;
    }
```

```
67
68      public void setPersons(Set<Person> persons) {
69          this.persons = persons;
70      }
71
72
73      public int getId() {
74          return id;
75      }
76
77
78      public void setId(int id) {
79          this.id = id;
80      }
81
82
83      public int getNumberOfSeats() {
84          return numberOfSeats;
85      }
86
87      public void setNumberOfSeats(int numberOfSeats) {
88          this.numberOfSeats = numberOfSeats;
89      }
90
91      public double getPrice() {
92          return price;
93      }
94
95      public void setPrice(double price) {
96          this.price = price;
97      }
98
99      public String getSource() {
100         return source;
101     }
102
103     public void setSource(String source) {
104         this.source = source;
105     }
106
107     public String getDestination() {
108         return destination;
109     }
110
111     public void setDestination(String destination) {
112         this.destination = destination;
113     }
114
115     public Date getDateOfDeparture() {
116         return dateOfDeparture;
117     }
118
119     public void setDateOfDeparture(Date dateOfDeparture) {
120         this.dateOfDeparture = dateOfDeparture;
121     }
122
123     public Date getDateOfArrival() {
124         return dateOfArrival;
125     }
126
127     public void setDateOfArrival(Date dateOfArrival) {
128         this.dateOfArrival = dateOfArrival;
129     }
130
131     public Airline getAirline() {
132         return airline;
133     }
134
135     public void setAirline(Airline airline) {
136         this.airline = airline;
137     }
138
139
140 }
```

Payment.java:

Payment.java

```java
package com.simplilearn.tables;

import java.sql.Date;

@Entity
@Table(name="payment")
public class Payment {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="pay_id")
    private int id;

    @Column(name="card_number")
    private String cardNumber;

    @Column(name="expiration")
    private Date expiration;

    @Column(name="security_code")
    private int securityCode;

    @Column(name="card_fname")
    private String firstNameOnCard;

    @Column(name="card_lname")
    private String lastNameOnCard;

    @ManyToOne()
    @JoinColumn(name="per_id", nullable=false)
    private Person person;

    public Payment() {}

    public Payment(String cardNumber, Date expiration, int securityCode, String firstNameOnCard, String lastNameOnCard,
            Person person) {
        super();
        this.cardNumber = cardNumber;
        this.expiration = expiration;
        this.securityCode = securityCode;
        this.firstNameOnCard = firstNameOnCard;
        this.lastNameOnCard = lastNameOnCard;
        this.person = person;
    }

    public int getId() {
        return id;
    }

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

    public String getCardNumber() {
        return cardNumber;
    }

    public void setCardNumber(String cardNumber) {
        this.cardNumber = cardNumber;
    }

    public Date getExpiration() {
        return expiration;
    }

    public void setExpiration(Date expiration) {
        this.expiration = expiration;
    }

    public int getSecurityCode() {
        return securityCode;
    }

    public void setSecurityCode(int securityCode) {
        this.securityCode = securityCode;
    }
```

```java
    }

    public void setSecurityCode(int securityCode) {
        this.securityCode = securityCode;
    }

    public String getFirstNameOnCard() {
        return firstNameOnCard;
    }

    public void setFirstNameOnCard(String firstNameOnCard) {
        this.firstNameOnCard = firstNameOnCard;
    }

    public String getLastNameOnCard() {
        return lastNameOnCard;
    }

    public void setLastNameOnCard(String lastNameOnCard) {
        this.lastNameOnCard = lastNameOnCard;
    }

    public Person getPerson() {
        return person;
    }

    public void setPerson(Person person) {
        this.person = person;
    }
}
```

Person.java:

Person.java

```java
package com.simplilearn.tables;

import java.sql.Date;

@Entity
@Table(name="person")
public class Person {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="per_id")
    private int id;

    @Column(name="first_name")
    private String firstName;

    @Column(name="last_name")
    private String lastName;

    @Column(name="email")
    private String email;

    @Column(name="birthday")
    private Date birthday;

    @Column(name="time_booked")
    private Date timeBooked;

    @OneToMany(mappedBy="person")
    private Set <Payment> payments;

    @ManyToMany
    @JoinTable(
      name = "personflight",
      joinColumns = @JoinColumn(name = "per_id"),
      inverseJoinColumns = @JoinColumn(name = "f_id"))
    Set<Flight> flights;

    public Person () {}

    public Person(String firstName, String lastName, String email, Date birthday, Date timeBooked,
            Set<Payment> payments, Set<Flight> flights) {
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
        this.birthday = birthday;
        this.timeBooked = timeBooked;
        this.payments = payments;
        this.flights = flights;
    }

    public int getId() {
        return id;
    }

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

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public Date getBirthday() {
        return birthday;
    }

    public void setBirthday(Date birthday) {
        this.birthday = birthday;
    }

    public Date getTimeBooked() {
        return timeBooked;
    }

    public void setTimeBooked(Date timeBooked) {
        this.timeBooked = timeBooked;
    }

    public Set<Payment> getPayments() {
        return payments;
    }

    public void setPayments(Set<Payment> payments) {
        this.payments = payments;
    }

    public Set<Flight> getFlights() {
        return flights;
    }

    public void setFlights(Set<Flight> flights) {
        this.flights = flights;
    }



}
```

com.simplilearn.util

HibernateUtil.java:

HibernateUtil.java

```java
package com.simplilearn.util;

import org.hibernate.SessionFactory;
import org.hibernate.boot.Metadata;
import org.hibernate.boot.MetadataSources;
import org.hibernate.boot.registry.StandardServiceRegistry;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

public class HibernateUtil {

    private static SessionFactory factory;

    public static SessionFactory buildSessionFactory() {

        // 1. Load Configuration
        StandardServiceRegistry  sService = new StandardServiceRegistryBuilder()
                .configure("hibernate.annotation.cfg.xml").build();
        Metadata meta = new MetadataSources(sService).getMetadataBuilder().build();

        // 2. Create Session factory
        factory = meta.getSessionFactoryBuilder().build();

        return factory;
    }

}
```

com.simplilearn.sql.files

setupDB.sql

```sql
use flyawaydb;

create table airline(
    a_id int NOT NULL AUTO_INCREMENT,
    airline varchar(255) NOT NULL,
    PRIMARY KEY (a_id)
);

create table flight(
    f_id int(11) NOT NULL AUTO_INCREMENT,
    number_of_seats int(11) NOT NULL,
    price decimal(10,2) NOT NULL,
    source varchar(255) NOT NULL,
    destination varchar(255) NOT NULL,
    date_of_departure date NOT NULL,
    date_of_arrival date NOT NULL,
    a_id int(11) NOT NULL,
    PRIMARY KEY (f_id),
    CONSTRAINT airline_fk FOREIGN KEY (a_id) REFERENCES airline (a_id)
);

create table person(
    per_id int(11) NOT NULL AUTO_INCREMENT,
    first_name varchar(255) NOT NULL,
    last_name varchar(255) NOT NULL,
    email varchar(255) NOT NULL,
    birthday date NOT NULL,
    time_booked DATETIME default CURRENT_TIMESTAMP NOT NULL,
    PRIMARY KEY (per_id)
);


create table personflight(
    per_id int(11) NOT NULL,
    f_id int(11) NOT NULL,
    PRIMARY KEY (per_id, f_id),
    CONSTRAINT person_fk FOREIGN Key (per_id) REFERENCES person (per_id),
    CONSTRAINT flights_fk FOREIGN Key (f_id) REFERENCES flight (f_id)
);

create table payment(
    pay_id int(11) NOT NULL AUTO_INCREMENT,
    card_number varchar(255) NOT NULL,
    expiration DATE NOT NULL,
    security_code int(11) NOT NULL,
    card_fname varchar(255) NOT NULL,
    card_lname varchar(255) NOT NULL,
    per_id int(11) NOT NULL,
    PRIMARY KEY (pay_id),
    CONSTRAINT person_payment_fk FOREIGN KEY (per_id) REFERENCES person (per_id)
);

create table admin(
    admin_id int (11) NOT NULL AUTO_INCREMENT,
    first_name varchar(255) NOT NULL,
    last_name varchar(255) NOT NULL,
    email varchar(255) NOT NULL,
    password varchar(255) NOT NULL,
    PRIMARY KEY (admin_id)
);

INSERT INTO airline ( airline)
    values ("Emirates"), ("Delta"), ("United Airlines"), ("Turkish Airlines"), ("Uganda Airlines");

INSERT INTO flight (number_of_seats, price, source, destination, date_of_departure, date_of_arrival, a_id)
    values (180, 500, "ORD", "DBX", "2020-10-20", "2020-10-22",
        (select a_id from airline where airline = "Emirates")),
     (10, 1200, "EBB", "LAX", "2020-10-23", "2020-10-24",
        (select a_id from airline where airline = "Uganda Airlines")),
      (150, 200, "ORD", "LAX", "2020-10-21", "2020-10-21",
        (select a_id from airline where airline = "Delta")),
      (20, 150, "ORD", "LAX", "2020-10-20", "2020-10-21",
        (select a_id from airline where airline = "United Airlines")),
      (120, 700, "DBX", "IATA", "2020-10-24", "2020-10-25",
        (select a_id from airline where airline = "Turkish Airlines")),
      (123, 250, "DBX", "EBB", "2020-10-22", "2020-10-22",
        (select a_id from airline where airline = "Emirates"));

 INSERT INTO person (first_name, last_name, email, birthday)
     values ("Pablo" , "Smith" , "pablo@gmail.com", "1990-02-01"),
         ("Pablo" , "Tisasirana" , "pablo@gmail.com", "1990-02-01"),
         ("Veronica" , "Villagomez" , "veronica@gmail.com", "1990-02-01"),
         ("Shay" , "Finn" , "shay@gmail.com", "1990-02-01"),
         ("Nick" , "Lorance" , "nick@gmail.com", "1990-02-01"),
         ("John" , "Brook" , "john@gmail.com", "1990-02-01"),
         ("Albert" , "Gaca" , "albert@gmail.com", "1990-02-01"),
         ("Chris" , "Smith" , "chris@gmail.com", "1990-02-01");

 INSERT INTO personflight (per_id, f_id)
     values (1, 3),
         (2, 2),
         (3, 1),
         (4, 5),
         (5, 2),
         (6, 3),
         (7, 6),
         (8, 7);

 INSERT INTO payment (card_number, expiration, security_code, card_fname, card_lname, per_id)
     values ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 1),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 2),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 3),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 4),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 5),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 6),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 7),
         ( "1234 5678 1021 9182" , "2022-10-31", 123, "Pablo", "Smith", 8);

 INSERT INTO admin ( first_name, last_name, email, password)
     values ("Bryan", "Finn", "admin@gmail.com", "admin");

```

Pushing the code to GitHub repository