

Project and Developer Details

Project Name: FlyAway

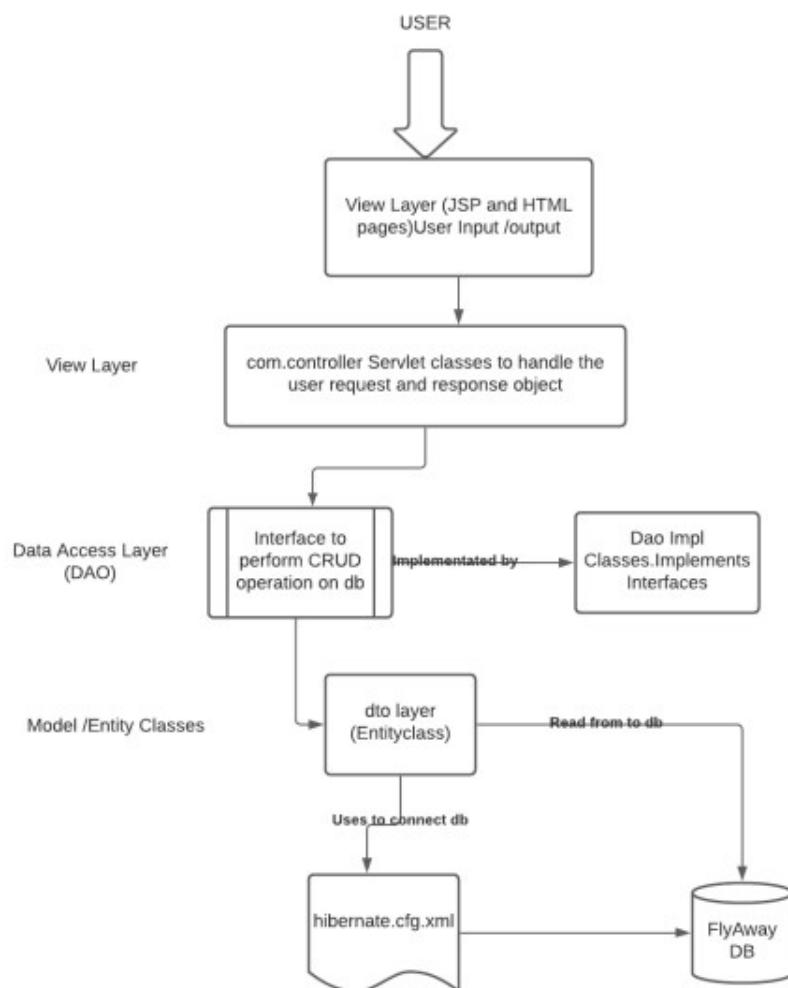
Developer Name: Amal Aldawas

Project Details:

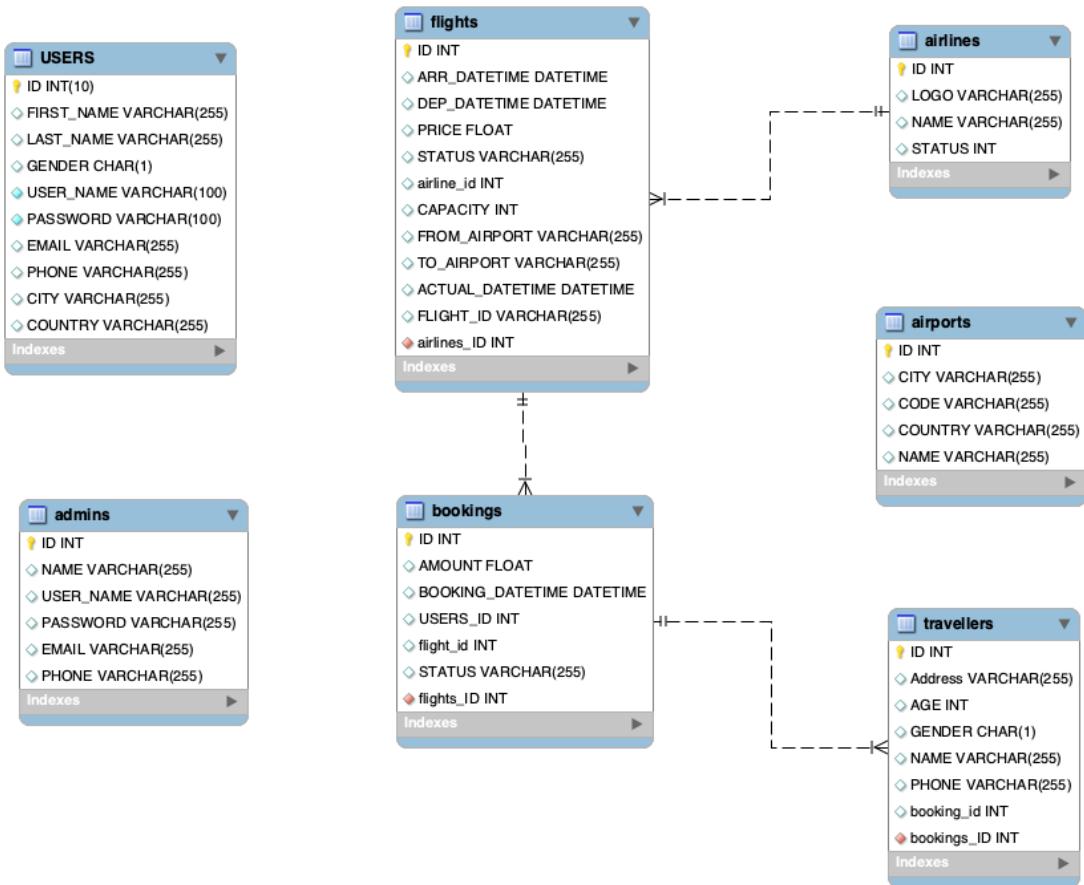
Steps to use:

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

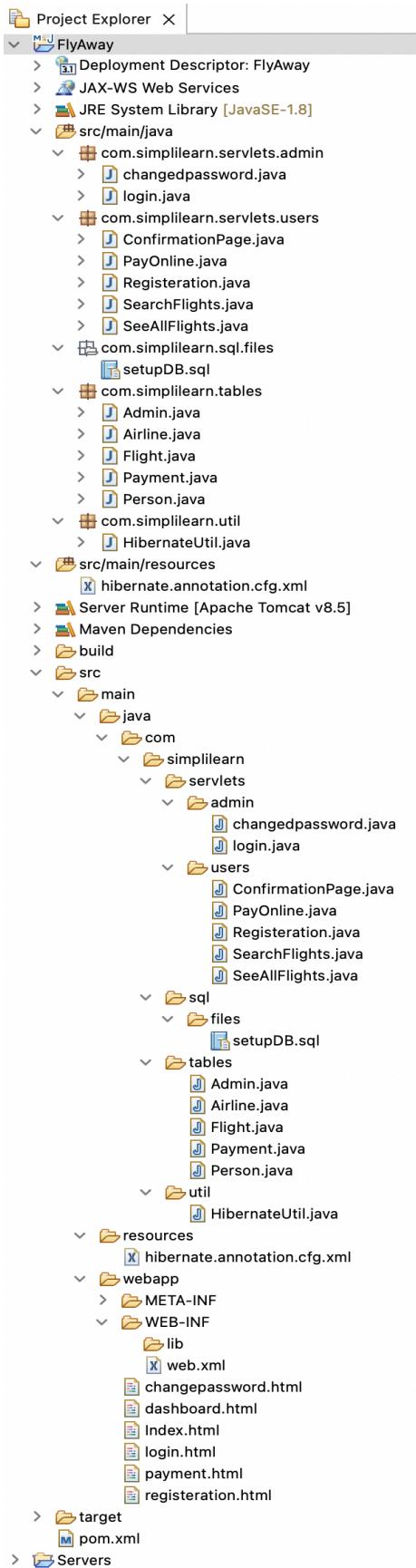
Flow of the Application



Database table design and relationship:



Folder Structure:



Project code:

pom.xml

```
1  <project xmlns="http://maven.apache.org/POM/4.0.0"
2   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5     <groupId>com.simplilearn</groupId>
6     <artifactId>FlyAway</artifactId>
7     <version>0.0.1-SNAPSHOT</version>
8     <packaging>war</packaging>
9
10    <url>http://maven.apache.org</url>
11  <dependencies>
12    <dependency>
13      <groupId>junit</groupId>
14      <artifactId>junit</artifactId>
15      <version>3.8.1</version>
16      <scope>test</scope>
17    </dependency>
18    <!-- hibernate-core -->
19    <dependency>
20      <groupId>org.hibernate</groupId>
21      <artifactId>hibernate-core</artifactId>
22      <version>5.4.1.Final</version>
23    </dependency>
24
25    <!-- mysql-connector-java -->
26    <dependency>
27      <groupId>mysql</groupId>
28      <artifactId>mysql-connector-java</artifactId>
29      <version>8.0.21</version>
30    </dependency>
31
32    <dependency>
33      <groupId>javax.servlet</groupId>
34      <artifactId> servlet-api</artifactId>
35      <version>2.5</version>
36    </dependency>
37
38  </dependencies>
39  <build>
40    <finalName>flyaway</finalName>
41    <sourceDirectory>src/main/java</sourceDirectory>
42    <plugins>
43      <plugin>
44        <artifactId>maven-compiler-plugin</artifactId>
45        <version>3.5.1</version>           <configuration>
46          <source>1.8</source>
47          <target>1.8</target>
48        </configuration>
49      </plugin>
50    </plugins>
51  </build>
52
53
54 </project>
```

hibernate.annotation.cfg.xml

```
hibernate.annotation.cfg.xml X
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-configuration PUBLIC
3   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5
6<@.hibernate-configuration>
7<@ session-factory>
8  <property name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
9  <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/flyawaydb</property>
10 <property name="hibernate.connection.username">root</property>
11 <property name="hibernate.connection.password">root</property>
12 <property name="hibernate.connection.pool_size">1</property>
13 <property name="hibernate.current_session_context_class">thread</property>
14 <property name="hibernate.show_sql">true</property>
15 <property name="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</property>
16 <property name="hibernate.id.new_generator_mappings">false</property>
17
18 <mapping class="com.simplilearn.tables.Admin" />
19 <mapping class="com.simplilearn.tables.Airline" />
20 <mapping class="com.simplilearn.tables.Flight" />
21 <mapping class="com.simplilearn.tables.Payment" />
22 <mapping class="com.simplilearn.tables.Person" />
23 </session-factory>
24 </hibernate-configuration>
```

Index.html:

```
Index.html X
FlyAway/src/main/webapp/Index.html
2<@html>
3<@head>
4 <meta charset="UTF-8">
5 <title>Welcome To Fly Away Ticket Booking Portal</title>
6 </head>
7<@body>
8   <a href="login.html">Admin Login</a>
9   <hr>
10  <h1>Search For Flights</h1>
11<@ form name="searchForFlights" method="GET" action="search-flights">
12    Flying From : <input type="text" name="source"> <br/>
13    Flying To : <input type="text" name="destination"> <br/>
14    Your Max Price : <input type="number" name="price"> <br/>
15    Flights On or After Date : <input type="date" name="date"> <br/>
16    <br/>
17    <input type="submit" value="Search for flights!">
18 </form>
19<@ form name="seeAllFlights" method="GET" action="see-all-flights">
20    <input type="submit" value="See All Flights">
21 </form>
22 </body>
23 </html>
```

change password.html:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Admin Login Page</title>
6 </head>
7 <body>
8   <a href="dashboard.html">Dashboard</a>
9
10  <h1>Change Password</h1>
11 <form name="change password" method="POST" action="change-password">
12   New Password : <input type="password" name="password"> <br/>
13   Confirm New Password: <input type="password" name="confirmpassword"> <br/>
14   <br/>
15   <input type="submit" value="Change password">
16 </form>
17
18 </body>
19 </html>
```

dashboard.html:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="ISO-8859-1">
5   <title>Admin Dashboard</title>
6 </head>
7 <body>
8   <a href="index.html">Home</a> |
9   <a href="index.html">Logout</a> |
10  <a href="changepassword.html">Change Password</a> |
11  <a href="list-airlines">Airline Master View </a> |
12  <a href="list-source-destination">Source/Destination Master View</a> |
13  <a href="list-flights">Flight Master View</a><br/>
14  <hr>
15 </body>
16 </html>
```

login.html:

```
login.html X
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Admin Login Page</title>
6  </head>
7  <body>
8      <a href="index.html">Home</a>
9
10     <h1>Admin Login Page</h1>
11    <form name="loginForm" method="POST" action="login">
12        Username: <input type="text" name="username"> <br/>
13        Password : <input type="password" name="password"> <br/>
14        <br/>
15        <input type="submit" value="Login">
16    </form>
17
18 </body>
19 </html>
```

payment.html:

```
payment.html X
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Payment</title>
6  </head>
7  <body>
8      <h1>Enter Payment Details</h1>
9      <form name="paymentDetails" method="POST" action="pay-online">
10         First Name On Card : <input type="text" name="firstname" placeholder="John" required> <br/>
11         Last Name On Card : <input type="text" name="lastname" placeholder="Smith" required> <br/>
12         Card Number: <input type="text" name="cardnumber" placeholder="1234 1234 4321 9123" required> <br/>
13         Expiration (yyyy-MM-dd) : <input type="date" name="expiration" placeholder="2027-10-31" required> <br/>
14         Security Code : <input type="number" name="securitycode" placeholder="987" required> <br/>
15         <br/>
16         <input type="submit" value="Pay Ticket">
17     </form>
18
19 </body>
20 </html>
```

registration.html:

```
registration.html X
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Flight Registration Page</title>
6  </head>
7  <body>
8      <h1>Flight Registration Page</h1>
9      <form name="flightRegistration" method="POST" action="registration">
10         Flight ID : <input type="number" name="flightid" placeholder="12" required> <br/>
11         First Name : <input type="text" name="firstname" placeholder="John" required> <br/>
12         Last Name : <input type="text" name="lastname" placeholder="Doe" required> <br/>
13         Email : <input type="email" name="email" placeholder="johndoe@gmail.com" required> <br/>
14         Birthday (yyyy-MM-dd) : <input type="date" name="birthday" placeholder="1990-04-02" required> <br/>
15         <br/>
16         <input type="submit" value="Submit Flight Registration">
17     </form>
18
19 </body>
20 </html>
```

com.simplilearn.servlets.admin

changepassword.java

```
*changedpassword.java X
1 package com.simplilearn.servlets.admin;
2 import java.io.IOException;
3 import java.io.PrintWriter;
4 import javax.servlet.ServletException;
5 import javax.servlet.annotation.WebServlet;
6 import javax.servlet.http.HttpServlet;
7 import javax.servlet.http.HttpServletRequest;
8 import javax.servlet.http.HttpServletResponse;
9
10 /**
11 * Servlet implementation class changedpassword
12 */
13 @WebServlet("/change-password")
14 public class changedpassword extends HttpServlet {
15     /**
16      * @see HttpServlet#HttpServlet()
17      */
18     public changedpassword() { super();
19         // TODO Auto-generated constructor stub
20     }
21
22
23 /**
24 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
25 */
26 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
27     // TODO Auto-generated method stub response.getWriter().append("Served at: ").append(request.getContextPath());
28 }
29
30 /**
31 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
32 */
33 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
34     // TODO Auto-generated method stub
35     PrintWriter out = resp.getWriter();
36     String pass = req.getParameter("password");
37     if (!login.isLoggedIn){
38         out.println("You must login first");
39     } else if (pass.equals("")){
40         out.println("Password can't be empty");
41     } else if (login.isLoggedIn && !pass.equals("")){
42         login.password = pass;
43         out.println("Password changed. New Password is "+login.password);
44         try {
45             Thread.sleep(1000);
46         } catch (InterruptedException e) {
47             // TODO Auto-generated catch block e.printStackTrace();
48         }
49     } else {
50         out.println("something went wrong");
51     }
52     try {
53         req.getRequestDispatcher("passwordverified.html").forward(req, resp);
54     } catch (ServletException e) {
55         // TODO Auto-generated catch block e.printStackTrace();
56     } catch (IOException e) {
57         // TODO Auto-generated catch block e.printStackTrace();
58     }
59     out.close();
60 }
61 }
62 }
63 }
64 }
65 }
66 }
```

login.java

```
*login.java X
1 package com.simplilearn.servlets.admin;
2 import java.io.IOException; import java.io.PrintWriter;
3 import javax.servlet.ServletException;
4 import javax.servlet.annotation.WebServlet;
5 import javax.servlet.http.HttpServlet;
6 import javax.servlet.http.HttpServletRequest;
7 import javax.servlet.http.HttpServletResponse;
8 import javax.servlet.http.HttpSession;
9 import javax.servlet.http.HttpSessionEvent;
10
11 /**
12 * Servlet implementation class login
13 */
14
15 @WebServlet("/login")
16 public class login extends HttpServlet {
17     /**
18     * @see HttpServlet#HttpServlet()
19     */
20     private static final long serialVersionUID = 1L;    public static boolean isLoggedIn = false;    public static String password = "Abhi18uyaani9";    public static String email = "kumar.abhishek7885@gmail.com";
21     @Override
22     public void doPost (HttpServletRequest req, HttpServletResponse resp) throws IOException {
23         PrintWriter out = resp.getWriter();
24         String email = req.getParameter("username");    String pass = req.getParameter("password");
25         if (email.equals(login.email) && pass.equals(login.password)){    isLoggedIn = true;
26             out.println("You have LoggedIn");
27         }
28         try {
29             req.getRequestDispatcher("dashboard.html").forward(req, resp);
30         } catch (ServletException e) {
31             // TODO Auto-generated catch block e.printStackTrace();
32         } catch (IOException e) {
33             // TODO Auto-generated catch block e.printStackTrace();
34         }
35         if (isLoggedIn){    out.println("Login Failed : Incorrect email or Password");
36     }
37     }
38     isLoggedIn = false;    out.println("Login Failed : Incorrect email or Password");
39
40     out.close();
41 }
42 }
```

com.simplilearn.servlets.users

ConfirmationPage.java:

```
ConfirmationPage.java X
1 package com.simplilearn.servlets.users;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.util.List;
6
7 import javax.servlet.ServletException;
8 import javax.servlet.annotation.WebServlet;
9 import javax.servlet.http.HttpServlet;
10 import javax.servlet.http.HttpServletRequest;
11 import javax.servlet.http.HttpServletResponse;
12 import javax.servlet.http.HttpSession;
13
14 import org.hibernate.Session;
15 import org.hibernate.SessionFactory;
16
17 import com.simplilearn.tables.Flight;
18 import com.simplilearn.util.HibernateUtil;
19
20 /**
21 * Servlet implementation class ConfirmationPage
22 */
23 @WebServlet("/confirmation-page")
24 public class ConfirmationPage extends HttpServlet {
25     private static final long serialVersionUID = 1L;
26
27 /**
28 * @see HttpServlet#HttpServlet()
29 */
30 public ConfirmationPage() {
31 }
32
33 /**
34 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
35 */
36 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
37     // print writer
38     PrintWriter out = response.getWriter();
39     response.setContentType("text/html");
40
41     HttpSession httpsession = request.getSession();
42     String flightid = (String) httpsession.getAttribute("flightid");
43     String firstName = (String) httpsession.getAttribute("firstname");
44     int flightIdConverted = Integer.parseInt(flightid);
45     //open a connection
46     try {
47         SessionFactory sFactory = HibernateUtil.buildSessionFactory();
48         Session session = sFactory.openSession();
49         session.beginTransaction();
50
51         String query = "select f from Flight f"
52             " inner join Airline a"
53             " on f.airline=a.id"
54             " and f.numberofSeats > 0"
55             " and f.id = " + flightIdConverted;
56         @SuppressWarnings("unchecked")
57         List<Flight> flights = session.createQuery(query).list();
58
59         if (flights.size() > 0) {
60             out.print("<a href=\"index.html\">Home</a><br/>");
61             out.print("<h1>Congratulations! You're all set, " + firstName + "</h1><br/><br/>");
62             out.print("<h2>List of Confirmed Flights for " + firstName + "</h2><br/>");
63             out.print("<style> table,th,td { border : 1px solid black ; padding :15px; } </style>");
64             out.print("<table>");
65             out.print("<tr>");
66                 out.println("<th>"); out.println("Flight ID"); out.println("</th>");
67                 out.println("<th>"); out.println("Airline"); out.println("</th>");
68                 out.println("<th>"); out.println("Number Of Seats"); out.println("</th>");
69                 out.println("<th>"); out.println("Source"); out.println("</th>");
70                 out.println("<th>"); out.println("Desination"); out.println("</th>");
71                 out.println("<th>"); out.println("Departure Date"); out.println("</th>");
72                 out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
73                 out.println("<th>"); out.println("Price"); out.println("</th>");
74             out.println("</tr>");
75             /*Flight i:flights */
76
77                 out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
78                 out.println("<th>"); out.println("Price"); out.println("</th>");
79             out.println("</tr>");
80             for(Flight i:flights) {
81                 if (i.getNumberOfSeats() > 0) {
82                     out.print("<tr>");
83                         out.println("<td>"); out.println(i.getId()); out.println("</td>");
84                         out.println("<td>"); out.println(i.getAirline().getAirline().toString()); out.println("</td>");
85                         out.println("<td>"); out.println(i.getNumberofSeats()); out.println("</td>");
86                         out.println("<td>"); out.println(i.getSource()); out.println("</td>");
87                         out.println("<td>"); out.println(i.getDestination()); out.println("</td>");
88                         out.println("<td>"); out.println(i.getDateOfDeparture()); out.println("</td>");
89                         out.println("<td>"); out.println(i.getDateOfArrival()); out.println("</td>");
90                         out.println("<td>"); out.println(i.getPrice()); out.println("</td>");
91                     out.println("</tr>");
92                 }
93             }
94             session.close();
95
96         } catch (Exception e) {
97             e.printStackTrace();
98         } finally {
99             out.close();
100         }
101     }
102 }
103
104 /**
105 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
106 */
107 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
108
109 }
110 }
```

PayOnline.java:

```
1 package com.simplilearn.servlets.users;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Date;
6 import java.util.HashSet;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.regex.Pattern;
10
11 import javax.servlet.ServletException;
12 import javax.servlet.annotation.WebServlet;
13 import javax.servlet.http.HttpServlet;
14 import javax.servlet.http.HttpServletRequest;
15 import javax.servlet.http.HttpServletResponse;
16 import javax.servlet.http.HttpSession;
17
18 import org.hibernate.Session;
19 import org.hibernate.SessionFactory;
20 import org.hibernate.query.Query;
21
22 import com.simplilearn.tables.Flight;
23 import com.simplilearn.tables.Payment;
24 import com.simplilearn.tables.Person;
25 import com.simplilearn.util.HibernateUtil;
26
27 /**
28  * Servlet implementation class PayOnline
29 */
30 @WebServlet("/pay-online")
31 public class PayOnline extends HttpServlet {
32     private static final long serialVersionUID = 1L;
33
34     /**
35      * @see HttpServlet#HttpServlet()
36     */
37     public PayOnline() {
38     }
39
40     /**
41      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
42     */
43     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
44         response.sendRedirect("payment.html");
45     }
46
47     /**
48      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
49     */
50     @SuppressWarnings({"unchecked", "rawtypes"})
51     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
52         PrintWriter out = response.getWriter();
53         response.setContentType("text/html");
54
55         HttpSession httpsession = request.getSession(false);
56
57         String personflightid = (String) httpsession.getAttribute("flightid");
58         String personfirstname = (String) httpsession.getAttribute("firstname");
59         String personlastname = (String) httpsession.getAttribute("lastname");
60         String personemail = (String) httpsession.getAttribute("email");
61         String personbirthday = (String) httpsession.getAttribute("birthday");
62         long millis = System.currentTimeMillis();
63         Date timeBooked = new Date(millis);
64         Date dateSqlPerson = null;
65         try {
66             if ((dateSqlPerson.getDate().valueOf(personbirthday)) == null) {
67                 out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
68                 out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
69             }
70         } catch (Exception e1) {
71             out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
72             out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
73         }
74     }
75 }
```

```

PayOnline.java X
1    String fName = request.getParameter("firstname");
2    String lName = request.getParameter("lastname");
3    String cardNumber = request.getParameter("cardnumber").replaceAll("\\s", "");
4    System.out.println(cardNumber);
5    String expiration = request.getParameter("expiration");
6    int securitycode = Integer.parseInt(request.getParameter("securitycode"));
7    Date dateSqlPayment = null;
8    try {
9        dateSqlPayment=Date.valueOf(expiration);
10   }
11   catch (Exception e1) {
12       out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
13       out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
14   }
15
16   if (!Pattern.matches("\\d{15,16}", cardNumber)) {
17       out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
18       out.print("Make sure you are entering your card number in correctly");
19   }
20   else if (!Pattern.matches("\\d{3,4}", Integer.toString(securitycode))) {
21       out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
22       out.print("Make sure you are entering your security code in correctly");
23   } else if (dateSqlPayment.before(timeBooked)) {
24       out.println("<h1>You must ensure your card is not expired<h1>");
25       out.print("<a href = \"payment.html\"> Re-enter Information </a><br/>");
26   }
27   else {
28
29       //step 1. Confirm Fields Match Criteria
30       //Else redirect to payment
31       //step 2. insert into person table
32       //open a connection
33       try {
34
35           SessionFactory sFactory = HibernateUtil.buildSessionFactory();
36           Session session = sFactory.openSession();
37           session.beginTransaction();
38
39           int personflightidConverted = Integer.parseInt(personflightid);
40
41           String query1 = " from Flight where id = " + personflightidConverted;
42           List<Flight> list = (List<Flight>) session.createQuery(query1).list();
43           Set<Flight> flights = new HashSet<Flight>(list);
44
45
46           Person person = new Person();
47           person.setFirstName(personfirstname);
48           person.setLastName(personlastname);
49           person.setEmail(personemail);
50           person.setBirthday(dateSqlPerson);
51           person.setTimeBooked(timeBooked);
52           person.setFlights(flights);
53
54           Payment payment = new Payment();
55           payment.setFirstNameOnCard(fName);
56           payment.setLastNameOnCard(lName);
57           payment.setCardNumber(cardNumber);
58           payment.setExpiration(dateSqlPayment);
59           payment.setSecurityCode(securitycode);
60           payment.setPerson(person);
61
62           session.save(person);
63           session.save(payment);
64
65           String query = "select numberOfSeats from Flight f" +
66               " where id = " + personflightidConverted;
67           int resultFromQ1 = (int) session.createQuery(query).getSingleResult();
68           Query q3=session.createQuery("update Flight set numberOfSeats=:n where id=:i");
69           q3.setParameter("n",resultFromQ1 - 1);
70           q3.setParameter("i",personflightidConverted);
71
72
73           int status=q3.executeUpdate();
74           System.out.println(status);
75
76           session.getTransaction().commit();
77           session.close();
78           response.sendRedirect("confirmation-page");
79           out.close();
80       } catch (Exception e) {
81           e.printStackTrace();
82       }
83   }
84
85   //step 3. insert into payment table
86   //step 4. reduce person count
87   //step 5. Redirect to confirmation page
88
89 }
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170

```

Registration.java:

```
1 package com.simplilearn.servlets.users;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Date;
6 import java.util.List;
7 import java.util.regex.Pattern;
8
9 import javax.servlet.ServletException;
10 import javax.servlet.annotation.WebServlet;
11 import javax.servlet.http.HttpServlet;
12 import javax.servlet.http.HttpServletRequest;
13 import javax.servlet.http.HttpServletResponse;
14 import javax.servlet.http.HttpSession;
15
16 import org.hibernate.Session;
17 import org.hibernate.SessionFactory;
18
19 import com.simplilearn.tables.Flight;
20 import com.simplilearn.util.HibernateUtil;
21
22 /**
23  * Servlet implementation class Registration
24 */
25 @WebServlet("/registration")
26 public class Registration extends HttpServlet {
27     private static final long serialVersionUID = 1L;
28
29     /**
30      * @see HttpServlet#HttpServlet()
31     */
32     public Registration() {
33     }
34
35     /**
36      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
37     */
38     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
39
40     }
41
42     /**
43      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
44     */
45     @SuppressWarnings("unchecked")
46     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
47
48         // print writer
49         PrintWriter out = response.getWriter();
50         response.setContentType("text/html");
51
52         HttpSession httpSession = request.getSession();
53         String flightId = request.getParameter("flightid");
54         String firstname = request.getParameter("firstname");
55         String lastname = request.getParameter("lastname");
56         String email = request.getParameter("email");
57         String birthday = request.getParameter("birthday");
58
59         request.setAttribute("firstname", firstname);
60         Date dateSqlBirthday = null;
61         try {
62             dateSqlBirthday = Date.valueOf(birthday);
63         } catch (Exception e) {
64             out.print("<a href = \"registration.html\"> Re-Enter Information </a><br/>");
65             out.print("Make sure you are entering your date code in correctly (yyyy-MM-dd)");
66         }
67
68         //email
69         if (!Pattern.matches("[A-Za-z0-9]+(.+)@(.+)", email)) {
70             out.print("<a href = \"registration.html\"> Re-enter Information </a><br/>");
71             out.print("Make sure you are entering in a valid email address");
72         } else if (dateSqlBirthday==null) {
73             out.print("<a href = \"registration.html\"> Re-enter Information </a><br/>");
74             out.print("Make sure you are entering in a valid date format (yyyy-MM-dd)");
75         } else {
76             httpSession.setAttribute("flightid", flightId);
77             httpSession.setAttribute("firstname", firstname);
78             httpSession.setAttribute("lastname", lastname);
79             httpSession.setAttribute("email", email);
80             httpSession.setAttribute("birthday", birthday);
81
82             //open a connection
83             try {
84                 SessionFactory sfactory = HibernateUtil.buildSessionFactory();
85                 Session session = sfactory.openSession();
86                 session.beginTransaction();
87
88                 String query = "select f from Flight f" +
89                     " inner join Airline a" +
90                     " on f.airline.id" +
91                     " and f.numberOfSeats > 0";
92                 if (flightId !=null && flightId.trim().length() > 0) {
93                     try {
94                         int flightIdParsed = Integer.parseInt(flightId);
95                         String flightIdQuery = " and f.id =" + flightIdParsed;
96                         query.concat(flightIdQuery);
97                     } catch (Exception e) {
98                         out.println("<h1>You must enter a valid flight id<h1>");
99                     }
100                }
101
102                List<Flight> flights = session.createQuery(query).list();
103
104                if (flights.size() > 0) {
105                    out.print("<a href=\"index.html\">Home</a>");
106                    out.print("<h1>Confirm Flight Details: </h1>");
107                    for(Flight i:flights) {
108                        if (i.getNumberOfSeats() > 0) {
109                            out.println("Flight ID: " + i.getId() + "<br/>");
110                            out.println("Airline: " + i.getAirline().getAirline().toString() + "<br/>");
111                            out.println("Number Of Seats: " + i.getNumberOfSeats() + "<br/>");
112                            out.println("Source " + i.getSource() + "<br/>");
113                            out.println("Destination:" + i.getDestination() + "<br/>");
114                            out.println("Departure Date: " + i.getDateOfDeparture() + "<br/>");
115                            out.println("Arrival Date: " + i.getDateOfArrival() + "<br/>");
116                            out.println("<h1>Your Total is: $" + i.getPrice() + "<h1><br/>");
117                            out.println("<br>");
118                        }
119                    }
120                    request.getRequestDispatcher("payment.html").include(request, response);
121                } else {
122                    out.print("<a href=\"index.html\">Home</a>");
123                    out.print("<h1>You must enter a valid flight id!<h1>");
124                }
125                session.close();
126                out.close();
127            } catch (Exception e) {
128                e.printStackTrace();
129            }
130        }
131    }
132 }
133 }
```

SearchFlights.java:

```
1 package com.simplilearn.servlets.users;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Date;
6 import java.util.List;
7
8 import javax.servlet.ServletException;
9 import javax.servlet.annotation.WebServlet;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13 import org.hibernate.Session;
14 import org.hibernate.SessionFactory;
15
16 import com.simplilearn.tables.Flight;
17 import com.simplilearn.util.HibernateUtil;
18
19 /**
20 * Servlet implementation class SearchFlights
21 */
22 @WebServlet("search-flights")
23 public class SearchFlights extends HttpServlet {
24     private static final long serialVersionUID = 1L;
25
26     /**
27      * @see HttpServlet#HttpServlet()
28     */
29     public SearchFlights() {
30     }
31
32     /**
33      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
34     */
35     @SuppressWarnings("unchecked")
36     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
37         PrintWriter out = response.getWriter();
38         response.setContentType("text/html");
39
40         String source = request.getParameter("source");
41         String destination = request.getParameter("destination");
42         String price = request.getParameter("price");
43         String date = request.getParameter("date");
44
45         //open a connection
46         try {
47             SessionFactory sFactory = HibernateUtil.buildSessionFactory();
48             Session session = sFactory.openSession();
49             session.beginTransaction();
50
51             String query = "select f from Flight f" +
52                 " inner join Airline aa" +
53                 " on f.airline=a.id" +
54                 " and f.numberofseats > 0";
55             if (source != null && source.trim().length() > 0) {
56                 String sourceQuery = " and f.source = '" + source + "'";
57                 query = query.concat(sourceQuery);
58             }
59             if (destination != null && destination.trim().length() > 0) {
60                 String destinationQuery = " and f.destination = '" + destination + "'";
61                 query = query.concat(destinationQuery);
62             }
63             if (price != null && price.trim().length() > 0) {
64                 try {
65                     Double priceParsed = Double.parseDouble(price);
66                     String priceQuery = " and f.price <=" + priceParsed;
67                     query = query.concat(priceQuery);
68                 } catch (Exception e) {
69                     out.println("<h1>You must enter a valid number for price<h1>");
70                 }
71             }
72             if (date != null && date.trim().length() > 0) {
73                 try {
74                     Date dateObj = Date.valueOf(date);
75                     String dateQuery = " and f.dateofDeparture >= '" + dateObj + "'";
76                     query = query.concat(dateQuery);
77                 } catch (Exception e1) {
78                     out.println("<h1>You must enter the date correctly into yyyy-MM-dd format<h1>");
79                 }
80             }
81             System.out.println(query);
82             List<Flight> flights = session.createQuery(query).list();
83
84             if (flights.size() > 0) {
85                 out.println("<html><head><title>All Flights</title></head><body>");
86                 out.println("<h1>List of All Flights:</h1>");
87                 out.println("<style> table,th,td { border : 1px solid black ; padding :15px; }</style>");
88                 out.println("<table>");
89                 out.println("<tr>"); out.println("<th>Flight ID"); out.println("</th>"); out.println("<th>Airline"); out.println("</th>"); out.println("<th>Number Of Seats"); out.println("</th>"); out.println("<th>Destination"); out.println("</th>"); out.println("<th>Departure Date"); out.println("</th>"); out.println("<th>Arrival Date"); out.println("</th>"); out.println("<th>Price"); out.println("</th>"); out.println("</tr>");
90                 for(Flight i:flights) {
91                     if (i.getNumberOfSeats() > 0) {
92                         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.getSource()); 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>"); }
93                 }
94                 out.println("</table>"); request.getRequestDispatcher("registration.html").include(request, response);
95             } else {
96                 out.print("<a href=\"index.html\">Home</a>"); out.print("<h1>Sorry there are no flights currently!<h1>"); }
97             session.close(); out.close();
98
99         } catch (Exception e) {
100             e.printStackTrace();
101         }
102     }
103
104     /**
105      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
106     */
107     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
108
109     }
110
111 }
```

SeeAllFlights.java:

```
SeeAllFlights.java X
1 package com.simplilearn.servlets.users;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.util.List;
6
7 import javax.servlet.ServletException;
8 import javax.servlet.annotation.WebServlet;
9 import javax.servlet.http.HttpServlet;
10 import javax.servlet.http.HttpServletRequest;
11 import javax.servlet.http.HttpServletResponse;
12
13 import org.hibernate.Session;
14 import org.hibernate.SessionFactory;
15
16 import com.simplilearn.tables.Flight;
17 import com.simplilearn.util.HibernateUtil;
18
19 /**
20  * Servlet implementation class SeeAllFlights
21 */
22 @WebServlet("/see-all-flights")
23 public class SeeAllFlights extends HttpServlet {
24     private static final long serialVersionUID = 1L;
25
26     /**
27      * @see HttpServlet#HttpServlet()
28     */
29     public SeeAllFlights() {
30 }
31
32     /**
33      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
34     */
35     @SuppressWarnings({"unused", "unchecked"})
36     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
37         // print writer
38         PrintWriter out = response.getWriter();
39         response.setContentType("text/html");
40
41         String source = request.getParameter("source");
42         String destination = request.getParameter("destination");
43         String price = request.getParameter("price");
44         String date = request.getParameter("date");
45
46         //open a connection
47         try {
48             SessionFactory sFactory = HibernateUtil.buildSessionFactory();
49             Session session = sFactory.openSession();
50             session.beginTransaction();
51
52             String query = "select f from Flight f" +
53                 " inner join Airline a" +
54                 " on f.airline=a.id" +
55                 " and f.numberofSeats > 0";
56
57             List<Flight> flights = session.createQuery(query).list();
58
59             if (flights.size() > 0) {
60                 out.print("<a href=\"index.html\">Home</a\"");
61                 out.println("<h1>List of all Flights:</h1\"");
62                 out.println("<style> table,th,td { border : 1px solid black ; padding :15px;} </style>");
63                 out.println("<table>");
64                 out.println("<tr>");
65                 out.println("<th>"); out.println("Flight ID"); out.println("</th>");
66                 out.println("<th>"); out.println("Airline"); out.println("</th>");
67                 out.println("<th>"); out.println("Number Of Seats"); out.println("</th>");
68                 out.println("<th>"); out.println("Source"); out.println("</th>");
69                 out.println("<th>"); out.println("Desintation"); out.println("</th>");
70                 out.println("<th>"); out.println("Departure Date"); out.println("</th>");
71                 out.println("<th>"); out.println("Arrival Date"); out.println("</th>");
72                 out.println("<th>"); out.println("Price"); out.println("</th>");
73                 out.println("</tr>");
74                 for(Flight i:flights) {
75                     if (i.getNumberofSeats() > 0) {
76                         out.println("<tr>");
77                         out.println("<td>"); out.println(i.getId()); out.println("</td>");
78                         out.println("<td>"); out.println(i.getAirline().toString()); out.println("</td>");
79                         out.println("<td>"); out.println(i.getNumberofSeats()); out.println("</td>");
80                         out.println("<td>"); out.println(i.getSource()); out.println("</td>");
81                         out.println("<td>"); out.println(i.getDestination()); out.println("</td>");
82                         out.println("<td>"); out.println(i.getDateofDeparture()); out.println("</td>");
83                         out.println("<td>"); out.println(i.getDateofArrival()); out.println("</td>");
84                         out.println("<td>"); out.println(i.getPrice()); out.println("</td>");
85                         out.println("</tr>");
86                     }
87                 }
88                 out.println("</table>");
89                 request.getRequestDispatcher("registration.html").include(request, response);
90             } else {
91                 out.print("<a href=\"index.html\">Home</a\"");
92                 out.print("<h1>Sorry there are no flights currently!<h1\"");
93             }
94
95             session.close();
96             out.close();
97
98         } catch (Exception e) {
99             e.printStackTrace();
100        }
101    }
102
103    /**
104     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
105     */
106    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
107    }
108
109 }
```

com.simplilearn.tables

Admin.java:

```
J Admin.java X
1 package com.simplilearn.tables;
2
3+ import javax.persistence.Column;[]
4
5
6
7
8
9
10 @Entity
11 @Table(name="admin")
12 public class Admin {
13     @Id
14     @GeneratedValue(strategy=GenerationType.AUTO)
15     @Column(name="admin_id")
16     private int id;
17
18     @Column(name="first_name")
19     private String firstName;
20
21     @Column(name="last_name")
22     private String lastName;
23
24     @Column(name="email")
25     private String email;
26
27     @Column(name="password")
28     private String password;
29
30     public Admin() {}
31     public Admin(String firstName, String lastName, String email, String password) {
32         super();
33         this.firstName = firstName;
34         this.lastName = lastName;
35         this.email = email;
36         this.password = password;
37     }
38
39     public int getId() {
40         return id;
41     }
42
43     public void setId(int id) {
44         this.id = id;
45     }
46
47     public String getFirstName() {
48         return firstName;
49     }
50
51     public void setFirstName(String firstName) {
52         this.firstName = firstName;
53     }
54
55     public String getLastName() {
56         return lastName;
57     }
58
59     public void setLastName(String lastName) {
60         this.lastName = lastName;
61     }
62
63     public String getEmail() {
64         return email;
65     }
66
67     public void setEmail(String email) {
68         this.email = email;
69     }
70
71     public String getPassword() {
72         return password;
73     }
74
75     public void setPassword(String password) {
76         this.password = password;
77     }
78 }
```

Airline.java:

```
J Airline.java X
1 package com.simplilearn.tables;
2
3+import java.util.Set;
12
13 @Entity
14 @Table(name="airline")
15 public class Airline {
16@  @Id
17     @GeneratedValue(strategy=GenerationType.AUTO)
18     @Column(name="a_id")
19     private int id;
20
21@  @Column(name="airline")
22     private String airline;
23
24@  @OneToMany(mappedBy="airline")
25     private Set <Flight> flights;
26
27     public Airline() {}
28@  public Airline(String airline, Set<Flight> flights) {
29         super();
30         this.airline = airline;
31         this.flights = flights;
32     }
33
34
35
36@  public int getId() {
37         return id;
38     }
39@  public void setId(int id) {
40         this.id = id;
41     }
42@  public String getAirline() {
43         return airline;
44     }
45@  public void setAirline(String airline) {
46         this.airline = airline;
47     }
48@  public Set<Flight> getFlights() {
49         return flights;
50     }
51@  public void setFlights(Set<Flight> flights) {
52         this.flights = flights;
53     }
54
55
56 }
```

Flight.java:

```
Flight.java X
1 package com.simplilearn.tables;
2
3 import java.sql.Date;
4
5 @Entity
6 @Table(name="flight")
7 public class Flight {
8     @Id
9     @GeneratedValue(strategy=GenerationType.AUTO)
10    @Column(name="f_id")
11    private int id;
12
13    @Column(name="number_of_seats")
14    private int numberOfSeats;
15
16    @Column(name="price")
17    private double price;
18
19    @Column(name="source")
20    private String source;
21
22    @Column(name="destination")
23    private String destination;
24
25    @Column(name="date_of_departure")
26    private Date dateOfDeparture;
27
28    @Column(name="date_of_arrival")
29    private Date dateOfArrival;
30
31    @ManyToOne()
32    @JoinColumn(name="a_id", nullable=false)
33    private Airline airline;
34
35    @ManyToMany(mappedBy = "flights")
36    Set<Person> persons;
37
38    public Flight() {}
39
40    public Flight(int numberOfSeats, double price, String source, String destination, Date dateOfDeparture,
41                  Date dateOfArrival, Airline airline, Set<Person> persons) {
42        super();
43        this.numberOfSeats = numberOfSeats;
44        this.price = price;
45        this.source = source;
46        this.destination = destination;
47        this.dateOfDeparture = dateOfDeparture;
48        this.dateOfArrival = dateOfArrival;
49        this.airline = airline;
50        this.persons = persons;
51    }
52
53    public Set<Person> getPersons() {
54        return persons;
55    }
56
57    public void setPersons(Set<Person> persons) {
58        this.persons = persons;
59    }
60
61
62    public int getId() {
63        return id;
64    }
65
66    public void setId(int id) {
67        this.id = id;
68    }
69
70    public int getNumberOfSeats() {
71        return numberOfSeats;
72    }
73
74
75
76
77
78    public int getNumberofSeats() {
79        return numberofSeats;
80    }
81
82
83    public int getNumberOfSeats() {
84        return numberOfSeats;
85    }
86
```

```
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 numberOfRows;
85    }
86
87@    public void setNumberOfSeats(int numberOfRows) {
88        this.numberOfSeats = numberOfRows;
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 X
1 package com.simplilearn.tables;
2
3 import java.sql.Date;
4
5 @Entity
6 @Table(name="payment")
7 public class Payment {
8     @Id
9     @GeneratedValue(strategy=GenerationType.AUTO)
10    @Column(name="pay_id")
11    private int id;
12
13    @Column(name="card_number")
14    private String cardNumber;
15
16    @Column(name="expiration")
17    private Date expiration;
18
19    @Column(name="security_code")
20    private int securityCode;
21
22    @Column(name="card_fname")
23    private String firstNameOnCard;
24
25    @Column(name="card_lname")
26    private String lastNameOnCard;
27
28    @ManyToOne()
29    @JoinColumn(name="per_id", nullable=false)
30    private Person person;
31
32    public Payment() {}
33
34    public Payment(String cardNumber, Date expiration, int securityCode, String firstNameOnCard, String lastNameOnCard,
35                  Person person) {
36        super();
37        this.cardNumber = cardNumber;
38        this.expiration = expiration;
39        this.securityCode = securityCode;
40        this.firstNameOnCard = firstNameOnCard;
41        this.lastNameOnCard = lastNameOnCard;
42        this.person = person;
43    }
44
45    public int getId() {
46        return id;
47    }
48
49    public void setId(int id) {
50        this.id = id;
51    }
52
53    public String getCardNumber() {
54        return cardNumber;
55    }
56
57    public void setCardNumber(String cardNumber) {
58        this.cardNumber = cardNumber;
59    }
60
61    public Date getExpiration() {
62        return expiration;
63    }
64
65    public void setExpiration(Date expiration) {
66        this.expiration = expiration;
67    }
68
69    public int getSecurityCode() {
70        return securityCode;
71    }
72
73    public void setSecurityCode(int securityCode) {
74        this.securityCode = securityCode;
75    }
76
77    public String getFirstNameOnCard() {
78        return firstNameOnCard;
79    }
80
81    public void setFirstNameOnCard(String firstNameOnCard) {
82        this.firstNameOnCard = firstNameOnCard;
83    }
84
85    public String getLastNameOnCard() {
86        return lastNameOnCard;
87    }
88
89    public void setLastNameOnCard(String lastNameOnCard) {
90        this.lastNameOnCard = lastNameOnCard;
91    }
92
93    public Person getPerson() {
94        return person;
95    }
96
97    public void setPerson(Person person) {
98        this.person = person;
99    }
100
101 }
```

Person.java:

```
Person.java X |  
1 package com.simplilearn.tables;  
2  
3+ import java.sql.Date;□  
16  
17 @Entity  
18 @Table(name="person")  
19 public class Person {  
20     @Id  
21     @GeneratedValue(strategy=GenerationType.AUTO)  
22     @Column(name="per_id")  
23     private int id;  
24  
25     @Column(name="first_name")  
26     private String firstName;  
27  
28     @Column(name="last_name")  
29     private String lastName;  
30  
31     @Column(name="email")  
32     private String email;  
33  
34     @Column(name="birthday")  
35     private Date birthday;  
36  
37     @Column(name="time_booked")  
38     private Date timeBooked;  
39  
40     @OneToMany(mappedBy="person")  
41     private Set<Payment> payments;  
42  
43     @ManyToMany  
44     @JoinTable(  
45         name = "personflight",  
46         joinColumns = @JoinColumn(name = "per_id"),  
47         inverseJoinColumns = @JoinColumn(name = "f_id"))  
48     Set<Flight> flights;  
49  
50     public Person () {}  
51  
52     public Person(String firstName, String lastName, String email, Date birthday, Date timeBooked,  
53                     Set<Payment> payments, Set<Flight> flights) {  
54         super();  
55         this.firstName = firstName;  
56         this.lastName = lastName;  
57         this.email = email;  
58         this.birthday = birthday;  
59         this.timeBooked = timeBooked;  
60         this.payments = payments;  
61         this.flights = flights;  
62     }  
63  
64     public int getId() {  
65         return id;  
66     }  
67  
68     public void setId(int id) {  
69         this.id = id;  
70     }  
71  
72     public String getFirstName() {  
73         return firstName;  
74     }  
75  
76     public void setFirstName(String firstName) {  
77         this.firstName = firstName;  
78     }  
79  
80     public String getLastname() {  
81         return lastName;  
82     }  
83  
84     public void setLastName(String lastName) {  
85         this.lastName = lastName;  
86     }  
87  
88     public String getEmail() {  
89         return email;  
90     }  
91  
92     public void setEmail(String email) {  
93         this.email = email;  
94     }  
95  
96     public Date getBirthday() {  
97         return birthday;  
98     }  
99  
100    public void setBirthday(Date birthday) {  
101        this.birthday = birthday;  
102    }  
103  
104    public Date getTimeBooked() {  
105        return timeBooked;  
106    }  
107  
108    public void setTimeBooked(Date timeBooked) {  
109        this.timeBooked = timeBooked;  
110    }  
111  
112    public Set<Payment> getPayments() {  
113        return payments;  
114    }  
115  
116    public void setPayments(Set<Payment> payments) {  
117        this.payments = payments;  
118    }  
119  
120    public Set<Flight> getFlights() {  
121        return flights;  
122    }  
123  
124    public void setFlights(Set<Flight> flights) {  
125        this.flights = flights;  
126    }  
127  
128  
129  
130  
131  
132 }
```

com.simplilearn.util

HibernateUtil.java:

```
J HibernateUtil.java X
1 package com.simplilearn.util;
2
3 import org.hibernate.SessionFactory;
4 import org.hibernate.boot.Metadata;
5 import org.hibernate.boot.MetadataSources;
6 import org.hibernate.boot.registry.StandardServiceRegistry;
7 import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8
9 public class HibernateUtil {
10
11     private static SessionFactory factory;
12
13     public static SessionFactory buildSessionFactory() {
14
15         // 1. Load Configuration
16         StandardServiceRegistry sService = new StandardServiceRegistryBuilder()
17             .configure("hibernate.annotation.cfg.xml").build();
18         Metadata meta = new MetadataSources(sService).getMetadataBuilder().build();
19
20         // 2. Create Session factory
21         factory = meta.getSessionFactoryBuilder().build();
22
23         return factory;
24     }
25
26 }
```

com.simplilearn.sql.files

setupDB.sql

```
1  USE flyawaydb;
2
3  CREATE TABLE airline(
4      a_id INT NOT NULL AUTO_INCREMENT,
5      airline VARCHAR(255) NOT NULL,
6      PRIMARY KEY (a_id)
7  );
8
9  CREATE TABLE flight(
10     f_id INT(11) NOT NULL AUTO_INCREMENT,
11     number_of_seats INT(11) NOT NULL,
12     price DECIMAL(10,2) NOT NULL,
13     source VARCHAR(255) NOT NULL,
14     destination VARCHAR(255) NOT NULL,
15     date_of_departure DATE NOT NULL,
16     date_of_arrival DATE NOT NULL,
17     a_id INT(11) NOT NULL,
18     PRIMARY KEY (f_id),
19     CONSTRAINT airline_fk FOREIGN KEY (a_id) REFERENCES airline (a_id)
20 );
21
22 CREATE TABLE person(
23     per_id INT(11) NOT NULL AUTO_INCREMENT,
24     first_name VARCHAR(255) NOT NULL,
25     last_name VARCHAR(255) NOT NULL,
26     email VARCHAR(255) NOT NULL,
27     birthday DATE NOT NULL,
28     time_booked DATETIME DEFAULT CURRENT_TIMESTAMP NOT NULL,
29     PRIMARY KEY (per_id)
30 );
31
32
33 CREATE TABLE personflight(
34     per_id INT(11) NOT NULL,
35     f_id INT(11) NOT NULL,
36     PRIMARY KEY (per_id, f_id),
37     CONSTRAINT person_fk FOREIGN KEY (per_id) REFERENCES person (per_id),
38     CONSTRAINT flights_fk FOREIGN KEY (f_id) REFERENCES flight (f_id)
39 );
40
41 CREATE TABLE payment(
42     pay_id INT(11) NOT NULL AUTO_INCREMENT,
43     card_number VARCHAR(255) NOT NULL,
44     expiration DATE NOT NULL,
45     security_code INT(11) NOT NULL,
46     card_fname VARCHAR(255) NOT NULL,
47     card_lname VARCHAR(255) NOT NULL,
48     per_id INT(11) NOT NULL,
49     PRIMARY KEY (pay_id),
50     CONSTRAINT person_payment_fk FOREIGN KEY (per_id) REFERENCES person (per_id)
51 );
52
53 CREATE TABLE admin(
54     admin_id INT(11) NOT NULL AUTO_INCREMENT,
55     first_name VARCHAR(255) NOT NULL,
56     last_name VARCHAR(255) NOT NULL,
57     email VARCHAR(255) NOT NULL,
58     password VARCHAR(255) NOT NULL,
59     PRIMARY KEY (admin_id)
60 );
61
62 INSERT INTO airline (airline)
63     VALUES ("Emirates"), ("Delta"), ("United Airlines"), ("Turkish Airlines"), ("Uganda Airlines");
64
65 INSERT INTO flight (number_of_seats, price, source, destination, date_of_departure, date_of_arrival, a_id)
66     VALUES (180, 500, "ORD", "DBX", "2020-10-20", "2020-10-22",
67         (SELECT a_id FROM airline WHERE airline = "Emirates"),
68         (10, 1200, "EBB", "LAX", "2020-10-23", "2020-10-24",
69             (SELECT a_id FROM airline WHERE airline = "Uganda Airlines")),
70
71         (10, 1200, "EBB", "LAX", "2020-10-23", "2020-10-24",
72             (SELECT a_id FROM airline WHERE airline = "Uganda Airlines")),
73
74         (150, 200, "ORD", "LAX", "2020-10-21", "2020-10-21",
75             (SELECT a_id FROM airline WHERE airline = "Delta")),
76
77         (20, 150, "ORD", "LAX", "2020-10-20", "2020-10-21",
78             (SELECT a_id FROM airline WHERE airline = "United Airlines")),
79
80         (120, 700, "DBX", "IATA", "2020-10-24", "2020-10-25",
81             (SELECT a_id FROM airline WHERE airline = "Turkish Airlines")),
82
83         (123, 250, "DBX", "EBB", "2020-10-22", "2020-10-22",
84             (SELECT a_id FROM airline WHERE airline = "Emirates")));
85
86 INSERT INTO person (first_name, last_name, email, birthday)
87     VALUES ("Pablo", "Smith", "pablo@gmail.com", "1990-02-01"),
88     ("Pablo", "Tisasirana", "pablo@gmail.com", "1990-02-01"),
89     ("Veronica", "Villagomez", "veronica@gmail.com", "1990-02-01"),
90     ("Shay", "Finn", "shay@gmail.com", "1990-02-01"),
91     ("Nick", "Lorance", "nick@gmail.com", "1990-02-01"),
92     ("John", "Brook", "john@gmail.com", "1990-02-01"),
93     ("Albert", "Gaca", "albert@gmail.com", "1990-02-01"),
94     ("Chris", "Smith", "chris@gmail.com", "1990-02-01");
95
96 INSERT INTO personflight (per_id, f_id)
97     VALUES (1, 3),
98         (2, 2),
99         (3, 1),
100        (4, 5),
101        (5, 2),
102        (6, 3),
103        (7, 6),
104        (8, 7);
105
106 INSERT INTO payment (card_number, expiration, security_code, card_fname, card_lname, per_id)
107     VALUES ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 1),
108     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 2),
109     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 3),
110     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 4),
111     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 5),
112     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 6),
113     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 7),
114     ("1234 5678 1021 9182", "2022-10-31", 123, "Pablo", "Smith", 8);
```

Pushing the code to GitHub repository