

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6     http://maven.apache.org/xsd/maven-4.0.0.xsd">
7   <modelVersion>4.0.0</modelVersion>
8
9   <groupId>com.cisco</groupId>
10  <artifactId>airlinebookingportal</artifactId>
11  <version>0.0.1-SNAPSHOT</version>
12  <packaging>war</packaging>
13
14  <name>airlinebookingportal Maven Webapp</name>
15  <!-- FIXME change it to the project's website -->
16  <url>http://www.example.com</url>
17
18  <properties>
19    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
20    <maven.compiler.source>1.7</maven.compiler.source>
21    <maven.compiler.target>1.7</maven.compiler.target>
22  </properties>
23
24  <dependencies>
25    <dependency>
26      <groupId>junit</groupId>
27      <artifactId>junit</artifactId>
28      <version>4.11</version>
29      <scope>test</scope>
30    </dependency>
31    <dependency>
32      <groupId>javax.servlet</groupId>
33      <artifactId>servlet-api</artifactId>
34      <version>3.0-alpha-1</version>
35    </dependency>
36    <dependency>
37      <groupId>mysql</groupId>
38      <artifactId>mysql-connector-java</artifactId>
39      <version>8.0.13</version>
40    </dependency>
41    <dependency>
42      <groupId>org.hibernate</groupId>
43      <artifactId>hibernate-core</artifactId>
44      <version>5.4.29.Final</version>
45    </dependency>
46    <dependency>
47      <groupId>org.hibernate</groupId>
48      <artifactId>hibernate-entitymanager</artifactId>
49      <version>4.1.8.Final</version>
50      <exclusions>
51        <exclusion>
52          <groupId>org.hibernate.javax.persistence</groupId>
53          <artifactId>hibernate-jpa-2.0-api</artifactId>
54        </exclusion>
55      </exclusions>
56    </dependency>
57    <dependency>
58      <groupId>javax.persistence</groupId>
59      <artifactId>persistence-api</artifactId>
```

```
58         <version>1.0.2</version>
59     </dependency>
60     <dependency>
61         <groupId>commons-codec</groupId>
62         <artifactId>commons-codec</artifactId>
63         <version>1.2</version>
64     </dependency>
65     <dependency>
66         <groupId>org.glassfish.jaxb</groupId>
67         <artifactId>jaxb-runtime</artifactId>
68         <version>2.3.1</version>
69     </dependency>
70     <dependency>
71         <groupId>javax.xml.bind</groupId>
72         <artifactId>jaxb-api</artifactId>
73         <version>2.3.1</version>
74     </dependency>
75     <dependency>
76         <groupId>javax.servlet</groupId>
77         <artifactId>jstl</artifactId>
78         <version>1.2</version>
79     </dependency>
80     <dependency>
81         <groupId>javax.servlet.jsp</groupId>
82         <artifactId>jsp-api</artifactId>
83         <version>2.2</version>
84         <scope>provided</scope>
85     </dependency>
86     <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
87 <dependency>
88     <groupId>javax.servlet</groupId>
89     <artifactId>javax.servlet-api</artifactId>
90     <version>3.0.1</version>
91     <scope>provided</scope>
92 </dependency>
93
94 </dependencies>
95
96 <build>
97     <finalName>airlinebookingportal</finalName>
98     <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults
99         (may be moved to parent pom) -->
100     <plugins>
101         <plugin>
102             <artifactId>maven-clean-plugin</artifactId>
103             <version>3.1.0</version>
104         </plugin>
105         <!-- see http://maven.apache.org/ref/current/maven-core/default-
106             bindings.html#Plugin_bindings_for_war_packaging -->
107         <plugin>
108             <artifactId>maven-resources-plugin</artifactId>
109             <version>3.0.2</version>
110         </plugin>
111         <plugin>
112             <artifactId>maven-compiler-plugin</artifactId>
113             <version>3.8.0</version>
114         </plugin>
115         <plugin>
116             <artifactId>maven-surefire-plugin</artifactId>
```

```
115         <version>2.22.1</version>
116     </plugin>
117     <plugin>
118         <artifactId>maven-war-plugin</artifactId>
119         <version>3.2.2</version>
120     </plugin>
121     <plugin>
122         <artifactId>maven-install-plugin</artifactId>
123         <version>2.5.2</version>
124     </plugin>
125     <plugin>
126         <artifactId>maven-deploy-plugin</artifactId>
127         <version>2.8.2</version>
128     </plugin>
129 </plugins>
130 </pluginManagement>
131 </build>
132 </project>
133
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <projectDescription>
3   <name>airlinebookingportal</name>
4   <comment></comment>
5   <projects>
6   </projects>
7   <buildSpec>
8     <buildCommand>
9       <name>org.eclipse.jdt.core.javabuilder</name>
10      <arguments>
11      </arguments>
12    </buildCommand>
13    <buildCommand>
14      <name>org.eclipse.wst.common.project.facet.core.builder</
name>
15      <arguments>
16      </arguments>
17    </buildCommand>
18    <buildCommand>
19      <name>org.eclipse.wst.validation.validationbuilder</name>
20      <arguments>
21      </arguments>
22    </buildCommand>
23    <buildCommand>
24      <name>org.eclipse.m2e.core.maven2Builder</name>
25      <arguments>
26      </arguments>
27    </buildCommand>
28  </buildSpec>
29  <natures>
30    <nature>org.eclipse.jem.workbench.JavaEMFNature</nature>
31    <nature>org.eclipse.wst.common.modulecore.ModuleCoreNature
</nature>
32    <nature>org.eclipse.jdt.core.javanature</nature>
33    <nature>org.eclipse.m2e.core.maven2Nature</nature>
34    <nature>org.eclipse.wst.common.project.facet.core.nature</nature
>
35    <nature>org.eclipse.wst.jsdt.core.jsNature</nature>
36  </natures>
37 </projectDescription>
38
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <classpath>
3   <classpathentry kind="src" output="target/classes" path="src/main/
  java">
4     <attributes>
5       <attribute name="optional" value="true"/>
6       <attribute name="maven.pomderived" value="true"/>
7     </attributes>
8   </classpathentry>
9   <classpathentry kind="src" output="target/test-classes" path="src/test
  /java">
10     <attributes>
11       <attribute name="test" value="true"/>
12       <attribute name="optional" value="true"/>
13       <attribute name="maven.pomderived" value="true"/>
14     </attributes>
15   </classpathentry>
16   <classpathentry exported="true" kind="con" path="org.eclipse.jdt.
  launching.JRE_CONTAINER/org.eclipse.jdt.internal.debug.ui.launcher.
  StandardVMType/JavaSE-1.7">
17     <attributes>
18       <attribute name="module" value="true"/>
19       <attribute name="maven.pomderived" value="true"/>
20     </attributes>
21   </classpathentry>
22   <classpathentry exported="true" kind="con" path="org.eclipse.m2e.
  MAVEN2_CLASSPATH_CONTAINER">
23     <attributes>
24       <attribute name="maven.pomderived" value="true"/>
25       <attribute name="org.eclipse.jst.component.dependency" value
  ="/WEB-INF/lib"/>
26     </attributes>
27   </classpathentry>
28   <classpathentry kind="output" path="target/classes"/>
29 </classpath>
30
```

```
1 package models;
2
3 import java.util.ArrayList;
18
19 @Entity
20 @Table(name = "Flight")
21 public class Flight {
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     @Column(name = "flight_id")
26     private int flightId;
27
28     @Column(name = "flight_number")
29     private int flightNumber;
30
31     @Column(name = "flight_name")
32     private String airline;
33
34     @Column(name = "Source")
35     private String origin;
36
37     @Column(name = "Destination")
38     private String target;
39
40     @Column(name = "Boarding_Date")
41     private String dob;
42
43     @Column(name = "Ticket_price")
44     private float price;
45
46     public Flight() {
47     }
48
49     @ManyToMany(cascade = CascadeType.ALL)
50     @JoinTable(name = "flight_passenger",
51         joinColumns = {
52             @JoinColumn(name = "flight_id")
53         },
54         inverseJoinColumns = {
55             @JoinColumn(name = "passenger_id")
56         }
57     )
58     List<Passenger> passenger = new ArrayList<Passenger>();
59
60
61     public Flight(int flightId, int flightNumber, String airline, String origin,
62         String target, String dob, float price) {
63         super();
64         this.flightId = flightId;
65         this.flightNumber = flightNumber;
66         this.airline = airline;
67         this.origin = origin;
68         this.target = target;
69         this.dob = dob;
70         this.price = price;
71     }
72
73     public Flight(int flightNumber, String airline, String origin, String target,
```

```
String dob, float price) {
73     super();
74     this.flightNumber = flightNumber;
75     this.airline = airline;
76     this.origin = origin;
77     this.target = target;
78     this.dob = dob;
79     this.price = price;
80 }
81
82 public int getFlightId() {
83     return flightId;
84 }
85
86 public void setFlightId(int flightId) {
87     this.flightId = flightId;
88 }
89
90 public int getFlightNumber() {
91     return flightNumber;
92 }
93
94 public void setFlightNumber(int flightNumber) {
95     this.flightNumber = flightNumber;
96 }
97
98 public String getAirline() {
99     return airline;
100 }
101
102 public void setAirline(String airline) {
103     this.airline = airline;
104 }
105
106 public String getOrigin() {
107     return origin;
108 }
109
110 public void setOrigin(String origin) {
111     this.origin = origin;
112 }
113
114 public String getTarget() {
115     return target;
116 }
117
118 public void setTarget(String target) {
119     this.target = target;
120 }
121
122 public String getDob() {
123     return dob;
124 }
125
126 public void setDob(String dob) {
127     this.dob = dob;
128 }
129
130 public float getPrice() {
```

```
131         return price;
132     }
133
134     public void setPrice(float price) {
135         this.price = price;
136     }
137
138     public List<Passenger> getPassenger() {
139         return passenger;
140     }
141
142     public void setPassenger(List<Passenger> passenger) {
143         this.passenger = passenger;
144     }
145
146 }
147
```



```
1 package models;
2
3 public class Password {
4
5     private static String pwd;
6
7     public Password() {
8         pwd = "admin";
9     }
10
11     public static String getPwd() {
12         return pwd;
13     }
14
15     public static void setPwd(String pwd) {
16         Password.pwd = pwd;
17     }
18
19 }
20
```

```
1 package models;
2
3 import java.util.ArrayList;
16
17 @Entity
18 @Table(name = "Passenger")
19 public class Passenger {
20
21     @Id
22     @GeneratedValue(strategy = GenerationType.IDENTITY)
23     @Column(name="passenger_id")
24     private int pId;
25
26     @Column(name="passenger_fname")
27     private String fname;
28
29     @Column(name="passenger_lname")
30     private String lname;
31
32     @Column(name="passenger_age")
33     private int age;
34
35     @Column(name="passenger_mob")
36     private long contact;
37
38     @Column(name="passenger_email")
39     private String email;
40
41     @ManyToMany(mappedBy = "passenger")//, cascade = CascadeType.MERGE)
42     private List<Flight> flight = new ArrayList<Flight>();
43
44     public Passenger() {
45     }
46
47     public Passenger(int pId, String fname, String lname, int age, long contact,
48 String email) {
49         super();
50         this.pId = pId;
51         this.fname = fname;
52         this.lname = lname;
53         this.age = age;
54         this.contact = contact;
55         this.email = email;
56     }
57
58     public Passenger(String fname, String lname, int age, long contact, String email)
59 {
60     super();
61     this.fname = fname;
62     this.lname = lname;
63     this.age = age;
64     this.contact = contact;
65     this.email = email;
66 }
67
68     public int getpId() {
69         return pId;
70     }
71 }
```

```
70
71     public void setpId(int pId) {
72         this.pId = pId;
73     }
74
75     public String getFname() {
76         return fname;
77     }
78
79     public void setFname(String fname) {
80         this.fname = fname;
81     }
82
83     public String getLname() {
84         return lname;
85     }
86
87     public void setLname(String lname) {
88         this.lname = lname;
89     }
90
91     public int getAge() {
92         return age;
93     }
94
95     public void setAge(int age) {
96         this.age = age;
97     }
98
99     public long getContact() {
100         return contact;
101     }
102
103     public void setContact(long contact) {
104         this.contact = contact;
105     }
106
107     public String getEmail() {
108         return email;
109     }
110
111     public void setEmail(String email) {
112         this.email = email;
113     }
114
115     public List<Flight> getFlight() {
116         return flight;
117     }
118
119     public void setFlight(List<Flight> flight) {
120         this.flight = flight;
121     }
122
123 }
124
```

```
1 package Controller;
2
3 import java.io.IOException;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21 @WebServlet("/")
22 public class MasterServlet extends HttpServlet {
23
24     private FlightDAO ob;
25     private Flight f;
26     private PassengerDAO obl;
27     private Passenger p;
28     private int num;
29     private String pd = null;
30     private int count, flag;
31     private List<Passenger> list1;
32
33     private static final long serialVersionUID = 1L;
34
35     public void init() {
36
37         ob = new FlightDAO();
38         obl = new PassengerDAO();
39         pd = "admin";
40         count = 0;
41         flag = 0;
42         list1 = null;
43
44     }
45
46     public MasterServlet() {
47         count = 0;
48         list1 = null;
49     }
50
51     @Override
52     public void service(HttpServletRequest request, HttpServletResponse response)
53         throws ServletException, IOException {
54
55         String action = request.getServletPath();
56         try {
57             switch (action) {
58
59                 case "/add":
60                     showNewForm(request, response);
61                     break;
62                 case "/delete":
63                     deleteDetails(request, response);
64                     break;
65                 case "/edit":
66                     EditDetails(request, response);
67                     break;
68                 case "/insert":
69                     insertDetails(request, response);
70                     break;
71                 case "/update":
72                     updateDetails(request, response);
73                     break;
74                 case "/reset":
75                     changePassword(request, response);
```

```
75         break;
76     case "/register":
77         register(request, response);
78         break;
79     case "/storage":
80         storage(request, response);
81         break;
82     case "/find":
83         getflightDetailsById(request, response);
84         break;
85     case "/login":
86         login(request, response);
87         break;
88     case "/passenger":
89         passengerFlightDetails(request, response);
90         break;
91     case "/insertPassenger":
92         count++;
93         passengerInsertDetails(request, response);
94         break;
95     case "/deletePassenger":
96         count--;
97         passengerDeleteDetails(request, response);
98         break;
99     case "/booking":
100         BookingDetails(request, response);
101         break;
102     default:
103         showAllDetails(response, request);
104         break;
105     }
106 } catch (Exception e) {
107     e.printStackTrace();
108 }
109 }
110 }
111 }
112
113 private void BookingDetails(HttpServletRequest request, HttpServletResponse
response)
114     throws ServletException, IOException {
115
116     List<Passenger> list = obl.getAllDetails();
117     list1 = list;
118     request.setAttribute("list", list);
119     RequestDispatcher rd = request.getRequestDispatcher("RegisterPage.jsp");
120     rd.forward(request, response);
121 }
122 }
123
124 private void passengerInsertDetails(HttpServletRequest request,
HttpServletResponse response) throws IOException {
125
126     if (count > num)
127         response.sendRedirect("HomePage.jsp");
128     else {
129         String fname = request.getParameter("fname");
130         String flname = request.getParameter("lname");
131         long contact = Long.parseLong(request.getParameter("contact"));
```

```
132         int age = Integer.parseInt(request.getParameter("age"));
133         String email = request.getParameter("email");
134         Passenger p = new Passenger(fname, flname, age, contact, email);
135         obl.insertPassengerInDB(p);
136         response.sendRedirect("booking");
137     }
138
139 }
140
141 private void passengerDeleteDetails(HttpServletRequest request,
    HttpServletResponse response) throws IOException {
142
143     int id = Integer.parseInt(request.getParameter("id"));
144     Passenger p = obl.getPassengerById(id);
145     obl.deletePassenger(p);
146     response.sendRedirect("booking");
147
148 }
149
150 private void EditDetails(HttpServletRequest request, HttpServletResponse response)
    throws SQLException, ServletException, IOException {
151
152     int id = Integer.parseInt(request.getParameter("fid"));
153     Flight f = ob.getFlightById(id);
154     RequestDispatcher dispatcher = request.getRequestDispatcher("AddFlight.jsp");
155     request.setAttribute("f", f);
156     dispatcher.forward(request, response);
157
158 }
159
160
161 private void updateDetails(HttpServletRequest request, HttpServletResponse
    response)
    throws SQLException, IOException {
162
163     int fid = Integer.parseInt(request.getParameter("fid"));
164     int fnumber = Integer.parseInt(request.getParameter("fnumber"));
165     String fname = request.getParameter("fname");
166     String forigin = request.getParameter("forigin");
167     String ftarget = request.getParameter("ftarget");
168     String date = request.getParameter("fdate");
169     float fprice = Float.parseFloat(request.getParameter("fprice"));
170     Flight fl = new Flight(fid, fnumber, fname, forigin, ftarget, date, fprice);
171     ob.updateFlight(fl);
172     response.sendRedirect("view");
173
174 }
175
176
177 private void deleteDetails(HttpServletRequest req, HttpServletResponse resp)
    throws IOException {
178
179     int id = Integer.parseInt(req.getParameter("fid"));
180     Flight f = ob.getFlightById(id);
181     ob.deleteFlight(f);
182     resp.sendRedirect("view");
183
184 }
185
186 private void showAllDetails(HttpServletResponse response, HttpServletRequest
    request)
```

```
187         throws ServletException, IOException {
188
189         List<Flight> list = ob.getAllDetails();
190         System.out.println(list);
191         request.setAttribute("list", list);
192         RequestDispatcher rd = request.getRequestDispatcher("FlightDetails.jsp");
193         rd.forward(request, response);
194     }
195
196     private void showNewForm(HttpServletRequest request, HttpServletResponse response)
197         throws ServletException, IOException {
198
199         RequestDispatcher rd = request.getRequestDispatcher("AddFlight.jsp");
200         rd.forward(request, response);
201     }
202
203     private void insertDetails(HttpServletRequest request, HttpServletResponse
204 response) throws IOException {
205
206         int fnumber = Integer.parseInt(request.getParameter("fnumber"));
207         String fname = request.getParameter("fname");
208         String forigin = request.getParameter("forigin");
209         String ftarget = request.getParameter("ftarget");
210         String date = request.getParameter("fdate");
211         float fprice = Float.parseFloat(request.getParameter("fprice"));
212         Flight fl = new Flight(fnumber, fname, forigin, ftarget, date, fprice);
213         ob.insertFlightInDB(fl);
214         response.sendRedirect("view");
215     }
216
217     private void passengerFlightDetails(HttpServletRequest request,
218 HttpServletResponse response)
219         throws ServletException, IOException {
220
221         String origin = request.getParameter("origin");
222         String target = request.getParameter("target");
223         String date = request.getParameter("date");
224         num = Integer.parseInt(request.getParameter("qty"));
225
226         List<Flight> list = obl.getAllDetailsByOriginDate(origin, date, target);
227         // System.out.println(list);
228         // request.setAttribute("num", num);
229         request.setAttribute("Selectedlist", list);
230         RequestDispatcher rd = request.getRequestDispatcher("PassengerFlights.jsp");
231         rd.forward(request, response);
232     }
233
234     private void getflightDetailsById(HttpServletRequest request, HttpServletResponse
235 response)
236         throws ServletException, IOException {
237
238         int id = Integer.parseInt(request.getParameter("fid"));
239         FlightDAO x = new FlightDAO();
240         f = x.getFlightById(id);
241         // request.setAttribute("num", num);
```

```
243     RequestDispatcher rd = request.getRequestDispatcher("booking");
244     rd.forward(request, response);
245
246 }
247
248 private void register(HttpServletRequest request, HttpServletResponse response)
249     throws ServletException, IOException {
250
251     if (count != 0) {
252         ob.relation(f, list1);
253         request.setAttribute("n", num);
254         request.setAttribute("f", f);
255         // request.setAttribute("p", p);
256         RequestDispatcher rd = request.getRequestDispatcher("SummaryPage.jsp");
257         rd.forward(request, response);
258     } else
259         response.sendRedirect("HomePage.jsp");
260
261 }
262
263 private void storage(HttpServletRequest request, HttpServletResponse response)
264     throws IOException {
265     obl.insertPassengerInDB(p);
266     response.sendRedirect("HomePage.jsp");
267
268 }
269
270 private void changePassword(HttpServletRequest request, HttpServletResponse
271     response)
272     throws IOException, ServletException {
273
274     String newpwd = request.getParameter("newpwd");
275     String confpwd = request.getParameter("confpwd");
276
277     if (newpwd.compareTo(confpwd) == 0) {
278         pd = newpwd;
279         response.sendRedirect("AdminLogin.jsp");
280     } else {
281         RequestDispatcher rd = request.getRequestDispatcher("ResetPage.jsp");
282         PrintWriter out = response.getWriter();
283         rd.include(request, response);
284         out.println("<center> <span style='color:red'> Invalid Credentials!!!
285         </span></center>");
286     }
287
288 }
289
290 private void login(HttpServletRequest request, HttpServletResponse response)
291     throws ServletException, IOException {
292
293     String email = request.getParameter("email");
294     String pwd = request.getParameter("pwd");
295
296     RequestDispatcher rd = null;
297
298     if (email.equalsIgnoreCase("admin@test.com") && pwd.equals(pd)) {
299         rd = request.getRequestDispatcher("view");
300         rd.forward(request, response);
301     }
302 }
```



```
298         } else {
299             rd = request.getRequestDispatcher("AdminLogin.jsp");
300             PrintWriter out = response.getWriter();
301             rd.include(request, response);
302             out.println("<center> <span style='color:red'> Invalid Credentials!!!
    </span></center>");
303         }
304
305     }
306
307 }
308
```