```xml
<?xml version="1.0" encoding="UTF-8"?>

<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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.cisco</groupId>
  <artifactId>airlinebookingportal</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

  <name>airlinebookingportal Maven Webapp</name>
  <!-- FIXME change it to the project's website -->
  <url>http://www.example.com</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.7</maven.compiler.source>
    <maven.compiler.target>1.7</maven.compiler.target>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
    <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>servlet-api</artifactId>
            <version>3.0-alpha-1</version>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.13</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.4.29.Final</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-entitymanager</artifactId>
            <version>4.1.8.Final</version>
            <exclusions>
                <exclusion>
                    <groupId>org.hibernate.javax.persistence</groupId>
                    <artifactId>hibernate-jpa-2.0-api</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
        <dependency>
            <groupId>javax.persistence</groupId>
            <artifactId>persistence-api</artifactId>
            <version>1.0.2</version>
        </dependency>
        <dependency>
            <groupId>commons-codec</groupId>
            <artifactId>commons-codec</artifactId>
            <version>1.2</version>
        </dependency>
        <dependency>
            <groupId>org.glassfish.jaxb</groupId>
            <artifactId>jaxb-runtime</artifactId>
            <version>2.3.1</version>
        </dependency>
        <dependency>
            <groupId>javax.xml.bind</groupId>
            <artifactId>jaxb-api</artifactId>
            <version>2.3.1</version>
        </dependency>
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jstl</artifactId>
            <version>1.2</version>
        </dependency>
        <dependency>
            <groupId>javax.servlet.jsp</groupId>
            <artifactId>jsp-api</artifactId>
            <version>2.2</version>
            <scope>provided</scope>
        </dependency>
        <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <version>3.0.1</version>
    <scope>provided</scope>
</dependency>

  </dependencies>

  <build>
    <finalName>airlinebookingportal</finalName>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults
(may be moved to parent pom) -->
      <plugins>
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.1.0</version>
        </plugin>
        <!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_war_packaging -->
        <plugin>
          <artifactId>maven-resources-plugin</artifactId>
          <version>3.0.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>3.8.0</version>
        </plugin>
        <plugin>
          <artifactId>maven-surefire-plugin</artifactId>
          <version>2.22.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-war-plugin</artifactId>
          <version>3.2.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-install-plugin</artifactId>
          <version>2.5.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-deploy-plugin</artifactId>
          <version>2.8.2</version>
        </plugin>
      </plugins>
    </pluginManagement>
  </build>
</project>

```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<projectDescription>
   <name>airlinebookingportal</name>
   <comment></comment>
   <projects>
   </projects>
   <buildSpec>
      <buildCommand>
         <name>org.eclipse.jdt.core.javabuilder</name>
         <arguments>
         </arguments>
      </buildCommand>
      <buildCommand>
         <name>org.eclipse.wst.common.project.facet.core.builder</name>
         <arguments>
         </arguments>
      </buildCommand>
      <buildCommand>
         <name>org.eclipse.wst.validation.validationbuilder</name>
         <arguments>
         </arguments>
      </buildCommand>
      <buildCommand>
         <name>org.eclipse.m2e.core.maven2Builder</name>
         <arguments>
         </arguments>
      </buildCommand>
   </buildSpec>
   <natures>
      <nature>org.eclipse.jem.workbench.JavaEMFNature</nature>
      <nature>org.eclipse.wst.common.modulecore.ModuleCoreNature</nature>
      <nature>org.eclipse.jdt.core.javanature</nature>
      <nature>org.eclipse.m2e.core.maven2Nature</nature>
      <nature>org.eclipse.wst.common.project.facet.core.nature</nature>
      <nature>org.eclipse.wst.jsdt.core.jsNature</nature>
   </natures>
</projectDescription>

```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<classpath>
    <classpathentry kind="src" output="target/classes" path="src/main/java">
        <attributes>
            <attribute name="optional" value="true"/>
            <attribute name="maven.pomderived" value="true"/>
        </attributes>
    </classpathentry>
    <classpathentry kind="src" output="target/test-classes" path="src/test/java">
        <attributes>
            <attribute name="test" value="true"/>
            <attribute name="optional" value="true"/>
            <attribute name="maven.pomderived" value="true"/>
        </attributes>
    </classpathentry>
    <classpathentry exported="true" kind="con" path="org.eclipse.jdt.launching.JRE_CONTAINER/org.eclipse.jdt.internal.debug.ui.launcher.StandardVMType/JavaSE-1.7">
        <attributes>
            <attribute name="module" value="true"/>
            <attribute name="maven.pomderived" value="true"/>
        </attributes>
    </classpathentry>
    <classpathentry exported="true" kind="con" path="org.eclipse.m2e.MAVEN2_CLASSPATH_CONTAINER">
        <attributes>
            <attribute name="maven.pomderived" value="true"/>
            <attribute name="org.eclipse.jst.component.dependency" value="/WEB-INF/lib"/>
        </attributes>
    </classpathentry>
    <classpathentry kind="output" path="target/classes"/>
</classpath>

```

```java
package models;

import java.util.ArrayList;
@Entity
@Table(name = "Flight")
public class Flight {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "flight_id")
    private int flightId;

    @Column(name = "flight_number")
    private int flightNumber;

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

    @Column(name = "Source")
    private String origin;

    @Column(name = "Destination")
    private String target;

    @Column(name = "Boarding_Date")
    private String dob;

    @Column(name = "Ticket_price")
    private float price;

    public Flight() {
    }

    @ManyToMany(cascade = CascadeType.ALL)
    @JoinTable(name = "flight_passenger",
        joinColumns = {
                @JoinColumn(name = "flight_id")
        },
        inverseJoinColumns = {
                @JoinColumn(name = "passenger_id")
        }
    )
    List<Passenger> passenger = new ArrayList<Passenger>();


    public Flight(int flightId, int flightNumber, String airline, String origin,
String target, String dob, float price) {
        super();
        this.flightId = flightId;
        this.flightNumber = flightNumber;
        this.airline = airline;
        this.origin = origin;
        this.target = target;
        this.dob = dob;
        this.price = price;
    }

    public Flight(int flightNumber, String airline, String origin, String target,
   String dob, float price) {
        super();
        this.flightNumber = flightNumber;
        this.airline = airline;
        this.origin = origin;
        this.target = target;
        this.dob = dob;
        this.price = price;
    }

    public int getFlightId() {
        return flightId;
    }

    public void setFlightId(int flightId) {
        this.flightId = flightId;
    }

    public int getFlightNumber() {
        return flightNumber;
    }

    public void setFlightNumber(int flightNumber) {
        this.flightNumber = flightNumber;
    }

    public String getAirline() {
        return airline;
    }

    public void setAirline(String airline) {
        this.airline = airline;
    }

    public String getOrigin() {
        return origin;
    }

    public void setOrigin(String origin) {
        this.origin = origin;
    }

    public String getTarget() {
        return target;
    }

    public void setTarget(String target) {
        this.target = target;
    }

    public String getDob() {
        return dob;
    }

    public void setDob(String dob) {
        this.dob = dob;
    }

    public float getPrice() {
        return price;
    }

    public void setPrice(float price) {
        this.price = price;
    }

    public List<Passenger> getPassenger() {
        return passenger;
    }

    public void setPassenger(List<Passenger> passenger) {
        this.passenger = passenger;
    }

}

```

```java
package models;

public class Password {

    private static String pwd;

    public Password() {
        pwd = "admin";
    }

    public static String getPwd() {
        return pwd;
    }

    public static void setPwd(String pwd) {
        Password.pwd = pwd;
    }

}

```

```java
package models;

import java.util.ArrayList;

@Entity
@Table(name = "Passenger")
public class Passenger {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name="passenger_id")
    private int pId;

    @Column(name="passenger_fname")
    private String fname;

    @Column(name="passenger_lname")
    private String lname;

    @Column(name="passenger_age")
    private int age;

    @Column(name="passenger_mob")
    private long contact;

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

    @ManyToMany(mappedBy = "passenger")//, cascade = CascadeType.MERGE)
    private List<Flight> flight = new ArrayList<Flight>();

    public Passenger() {
    }

    public Passenger(int pId, String fname, String lname, int age, long contact,
String email) {
        super();
        this.pId = pId;
        this.fname = fname;
        this.lname = lname;
        this.age = age;
        this.contact = contact;
        this.email = email;
    }


    public Passenger(String fname, String lname, int age, long contact, String email)
{
        super();
        this.fname = fname;
        this.lname = lname;
        this.age = age;
        this.contact = contact;
        this.email = email;
    }

    public int getpId() {
        return pId;
    }

    public void setpId(int pId) {
        this.pId = pId;
    }

    public String getFname() {
        return fname;
    }

    public void setFname(String fname) {
        this.fname = fname;
    }

    public String getLname() {
        return lname;
    }

    public void setLname(String lname) {
        this.lname = lname;
    }

    public int getAge() {
        return age;
    }

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

    public long getContact() {
        return contact;
    }

    public void setContact(long contact) {
        this.contact = contact;
    }

    public String getEmail() {
        return email;
    }

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

    public List<Flight> getFlight() {
        return flight;
    }

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

}

```

```java
package Controller;

import java.io.IOException;

@WebServlet("/")
public class MasterServlet extends HttpServlet {

    private FlightDAO ob;
    private Flight f;
    private PassengerDAO ob1;
    private Passenger p;
    private int num;
    private String pd = null;
    private int count, flag;
    private List<Passenger> list1;

    private static final long serialVersionUID = 1L;

    public void init() {

        ob = new FlightDAO();
        ob1 = new PassengerDAO();
        pd = "admin";
        count = 0;
        flag = 0;
        list1 = null;

    }

    public MasterServlet() {
        count = 0;
        list1 = null;
    }

    @Override
    public void service(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

        String action = request.getServletPath();
        try {
            switch (action) {

            case "/add":
                showNewForm(request, response);
                break;
            case "/delete":
                deleteDetails(request, response);
                break;
            case "/edit":
                EditDetails(request, response);
                break;
            case "/insert":
                insertDetails(request, response);
                break;
            case "/update":
                updateDetails(request, response);
                break;
            case "/reset":
                changePassword(request, response);
```

```java
                    break;
                case "/register":
                    register(request, response);
                    break;
                case "/storage":
                    storage(request, response);
                    break;
                case "/find":
                    getflightDetailsById(request, response);
                    break;
                case "/login":
                    login(request, response);
                    break;
                case "/passenger":
                    passengerFlightDetails(request, response);
                    break;
                case "/insertPassenger":
                    count++;
                    passengerInsertDetails(request, response);
                    break;
                case "/deletePassenger":
                    count--;
                    passengerDeleteDetails(request, response);
                    break;
                case "/booking":
                    BookingDetails(request, response);
                    break;
                default:
                    showAllDetails(response, request);
                    break;

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

    }

    private void BookingDetails(HttpServletRequest request, HttpServletResponse
response)
                throws ServletException, IOException {

            List<Passenger> list = ob1.getAllDetails();
            list1 = list;
            request.setAttribute("list", list);
            RequestDispatcher rd = request.getRequestDispatcher("RegisterPage.jsp");
            rd.forward(request, response);

    }

    private void passengerInsertDetails(HttpServletRequest request,
HttpServletResponse response) throws IOException {

            if (count > num)
                response.sendRedirect("HomePage.jsp");
            else {
                String fname = request.getParameter("fname");
                String flname = request.getParameter("lname");
                long contact = Long.parseLong(request.getParameter("contact"));
                int age = Integer.parseInt(request.getParameter("age"));
                String email = request.getParameter("email");
                Passenger p = new Passenger(fname, flname, age, contact, email);
                ob1.insertPassengerInDB(p);
                response.sendRedirect("booking");
            }

        }

        private void passengerDeleteDetails(HttpServletRequest request, HttpServletResponse response) throws IOException {

            int id = Integer.parseInt(request.getParameter("id"));
            Passenger p = ob1.getPassengerById(id);
            ob1.deletePassenger(p);
            response.sendRedirect("booking");

        }

        private void EditDetails(HttpServletRequest request, HttpServletResponse response)
                throws SQLException, ServletException, IOException {

            int id = Integer.parseInt(request.getParameter("fid"));
            Flight f = ob.getFlightById(id);
            RequestDispatcher dispatcher = request.getRequestDispatcher("AddFlight.jsp");
            request.setAttribute("f", f);
            dispatcher.forward(request, response);

        }

        private void updateDetails(HttpServletRequest request, HttpServletResponse response)
                throws SQLException, IOException {

            int fid = Integer.parseInt(request.getParameter("fid"));
            int fnumber = Integer.parseInt(request.getParameter("fnumber"));
            String fname = request.getParameter("fname");
            String forigin = request.getParameter("forigin");
            String ftarget = request.getParameter("ftarget");
            String date = request.getParameter("fdate");
            float fprice = Float.parseFloat(request.getParameter("fprice"));
            Flight fl = new Flight(fid, fnumber, fname, forigin, ftarget, date, fprice);
            ob.updateFlight(fl);
            response.sendRedirect("view");

        }

        private void deleteDetails(HttpServletRequest req, HttpServletResponse resp) throws IOException {

            int id = Integer.parseInt(req.getParameter("fid"));
            Flight f = ob.getFlightById(id);
            ob.deleteFlight(f);
            resp.sendRedirect("view");

        }

        private void showAllDetails(HttpServletResponse response, HttpServletRequest request)
```

```java
            throws ServletException, IOException {

        List<Flight> list = ob.getAllDetails();
        System.out.println(list);
        request.setAttribute("list", list);
        RequestDispatcher rd = request.getRequestDispatcher("FlightDetails.jsp");
        rd.forward(request, response);

    }

    private void showNewForm(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        RequestDispatcher rd = request.getRequestDispatcher("AddFlight.jsp");
        rd.forward(request, response);

    }

    private void insertDetails(HttpServletRequest request, HttpServletResponse response) throws IOException {

        int fnumber = Integer.parseInt(request.getParameter("fnumber"));
        String fname = request.getParameter("fname");
        String forigin = request.getParameter("forigin");
        String ftarget = request.getParameter("ftarget");
        String date = request.getParameter("fdate");
        float fprice = Float.parseFloat(request.getParameter("fprice"));
        Flight fl = new Flight(fnumber, fname, forigin, ftarget, date, fprice);
        ob.insertFlightInDB(fl);
        response.sendRedirect("view");

    }

    private void passengerFlightDetails(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        String origin = request.getParameter("origin");
        String target = request.getParameter("target");
        String date = request.getParameter("date");
        num = Integer.parseInt(request.getParameter("qty"));

        List<Flight> list = ob1.getAllDetailsByOriginDate(origin, date, target);
        // System.out.println(list);
        // request.setAttribute("num", num);
        request.setAttribute("Selectedlist", list);
        RequestDispatcher rd = request.getRequestDispatcher("PassengerFlights.jsp");
        rd.forward(request, response);

    }

    private void getflightDetailsById(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        int id = Integer.parseInt(request.getParameter("fid"));
        FlightDAO x = new FlightDAO();
        f = x.getFlightById(id);
        // request.setAttribute("num", num);
        RequestDispatcher rd = request.getRequestDispatcher("booking");
        rd.forward(request, response);

    }

    private void register(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        if (count != 0) {
            ob.relation(f, list1);
            request.setAttribute("n", num);
            request.setAttribute("f", f);
            // request.setAttribute("p", p);
            RequestDispatcher rd = request.getRequestDispatcher("SummaryPage.jsp");
            rd.forward(request, response);
        } else
            response.sendRedirect("HomePage.jsp");

    }

    private void storage(HttpServletRequest request, HttpServletResponse response)
throws IOException {

        ob1.insertPassengerInDB(p);
        response.sendRedirect("HomePage.jsp");

    }

    private void changePassword(HttpServletRequest request, HttpServletResponse
response)
            throws IOException, ServletException {

        String newpwd = request.getParameter("newpwd");
        String confpwd = request.getParameter("confpwd");

        if (newpwd.compareTo(confpwd) == 0) {
            pd = newpwd;
            response.sendRedirect("AdminLogin.jsp");
        } else {
            RequestDispatcher rd = request.getRequestDispatcher("ResetPage.jsp");
            PrintWriter out = response.getWriter();
            rd.include(request, response);
            out.println("<center> <span style='color:red'> Invalid Credentials!!!
</span></center>");
        }

    }

    private void login(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

        String email = request.getParameter("email");
        String pwd = request.getParameter("pwd");

        RequestDispatcher rd = null;

        if (email.equalsIgnoreCase("admin@test.com") && pwd.equals(pd)) {
            rd = request.getRequestDispatcher("view");
            rd.forward(request, response);
        } else {
            rd = request.getRequestDispatcher("AdminLogin.jsp");
            PrintWriter out = response.getWriter();
            rd.include(request, response);
            out.println("<center> <span style='color:red'> Invalid Credentials!!! </span></center>");
        }

    }

}

```