

FlyAway source code

MasterServlet.java

```
package Controller;

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

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

import models.Flight;
import models.Passenger;
import transferobjectaccess.FlightDAO;
import transferobjectaccess.PassengerDAO;

@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);
        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)
        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>");
        }
    }
}
}

```

Flight.java

```

package models;

import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;

import javax.persistence.CascadeType;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;

@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;
        }
    }
}

```

Passenger.java

```
package models;
```

```
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
```

```
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.ManyToOne;
```

```
@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;
```

```
    @ManyToOne(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;
    }
}

```