

Final Project FlyAway Specifications

Class: Become a Back-end Expert

Student: Hector Alarcon

Application: LockedMe.com

Date: 10/1/2021

FlyAway

Contents

Project Description:	3
Java technologies utilized:	3
Unique Selling Points	3
Sprint breakdown.....	4
Program Details:	5
GitHub:	5

Project Description:

- This project focused on making a web application

Java technologies utilized:

- Hibernate framework which allows for the mapping on database entities to java objects to communicate directly with requests.
- Maven framework allowing for quickly gathering and allocation of dependencies.
- Servlets containers that are essential in dealing with HTTP requests and other services like authentication and redirection to other server resources.
- Java server pages as well as some JSTLs for easier and less cumbersome acquisition and transmission of data between server pages.
- HTML and multiple tags for user input and data transmission. Mostly using the POST method for security reasons

Unique Selling Points

This excellent application provides the following features to the customer:

- Ease of use, all the menus have user interaction information displayed on the console describing all the options available to the user.
- Security, all of the class variables are set to private, and resources closed and only accessible to the class itself. Only letting the main method access the multiple resources.
- Bullet proof, sentinel values are used for input validation and making sure the user is aware of possible mistakes during inputting data.

Sprint breakdown

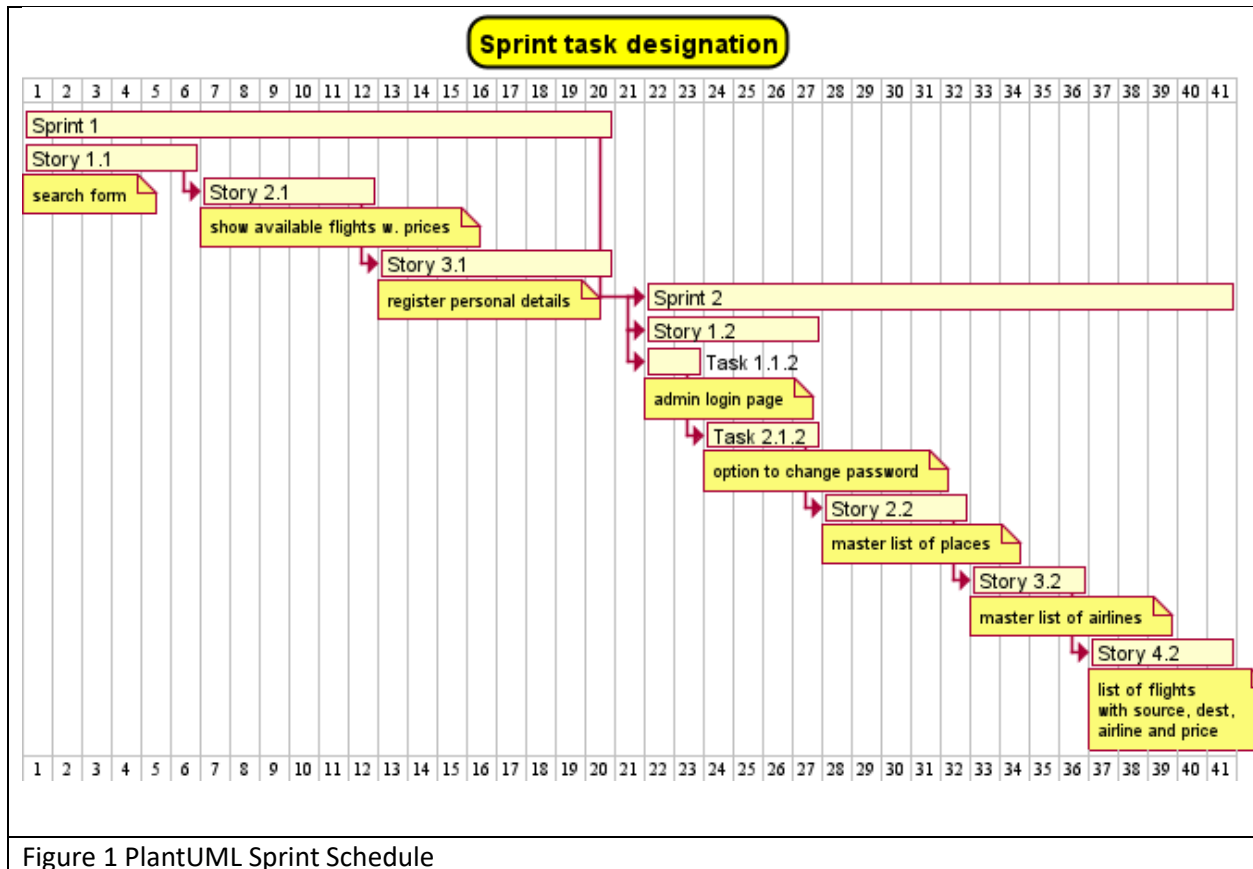


Figure 1 PlantUML Sprint Schedule

Program Details:

GitHub:

<https://github.com/ProgrammedPeinado/PracticeJava/tree/main/Simplilearn/Final%20Projects/flyaway>

Project Folder
<ul style="list-style-type: none">flyaway [PracticeJava main]<ul style="list-style-type: none">Deployment Descriptor: flyawayJAX-WS Web Servicessrc/main/java<ul style="list-style-type: none">com.controllers<ul style="list-style-type: none">AdminServlet.javaConfirmation.javaFlightList.javaFlightServlet.javaPasswordChange.javaValidateFlight.javaValidatePassenger.javacom.DAO<ul style="list-style-type: none">AdminDAO.javaAdminDaoImpl.javaFlightDAO.javaFlightDaoImpl.javaPassengerDAO.javaPassengerDaoImpl.javaPlaneDAO.javaPlaneDaoImpl.javacom.dto<ul style="list-style-type: none">Admin.javaFlight.javaPassenger.javaPlane.javahibernate.cfg.xmlsrc/test/javaMaven DependenciesServer Runtime [Apache Tomcat v8.5]JRE System Library [JavaSE-1.7]Deployed Resourcessrc<ul style="list-style-type: none">main<ul style="list-style-type: none">javawebapp<ul style="list-style-type: none">WEB-INF<ul style="list-style-type: none">lib<ul style="list-style-type: none">web.xmladmin_login.jspadmin_page.jspadmin_list.jspflightbook.jspindex.cssindex.jsppasswordChange.jsppaymentcomplete.jspregistration.jspsummary.jsptestDocumentation

AdminServlet

```
package com.controllers;

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

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

import com.DAO.AdminDAO;
import com.DAO.AdminDaoImpl;
import com.dto.Admin;

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

    /**
     * @see HttpServlet#HttpServlet()
     */
    public AdminServlet() {
        super();
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        List<Admin> admins = null;
        AdminDAO adminCheck = new AdminDaoImpl();
        String user = request.getParameter("user");
    }
}
```

```

        String pass = request.getParameter("pass");

        //List admin then compare
        admins = adminCheck.getAdmins();

        try
        {
            for(Admin a : admins)
            {
                if(a.getUser().equals(user) & a.getPass().equals(pass));
                {
                    response.sendRedirect("admin_page.jsp");
                }
            }
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
    }
}

```

FlightList

```

package com.controllers;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

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

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

```

```

import com.DAO.FlightDAO;
import com.DAO.FlightDaoImpl;
import com.dto.Flight;

import net.bytebuddy.description.type.TypeList.Generic;

/**
 * Servlet implementation class FlightList
 */
public class FlightList extends HttpServlet
{

    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public FlightList() {
        super();
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
    {
        String src = request.getParameter("source_country");
        String dest = request.getParameter("destination_country");
        String date = request.getParameter("book_date");
        int pass = Integer.parseInt(request.getParameter("n_pass"));

        PrintWriter out = response.getWriter();
        FlightDAO flightDAO = new FlightDaoImpl();

        try
        {
            List<Flight> res = flightDAO.listFlights(src, dest, date, pass);
            request.setAttribute("flightList", res);

```



```

        }
        catch(Exception e)
        {
            System.out.println("\n\nStack Trace:");
            e.printStackTrace();
            System.out.println("\n\nMessage:"+e.getMessage());
            System.out.println("\n\nThere are no flights matching your search
criterion.");
        }
        finally
        {
            request.getRequestDispatcher("flightbook.jsp").forward(request,response);
            out.close();
        }
    }
}

```

FlightServlet

```

package com.controllers;

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

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

import com.DAO.FlightDAO;
import com.DAO.FlightDaoImpl;
import com.DAO.PlaneDAO;
import com.DAO.PlaneDaoImpl;
import com.dto.Flight;
import com.dto.Plane;

/**
 * Servlet implementation class FlightServlet
 */
@WebServlet("/FlightServlet")

```

```

public class FlightServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public FlightServlet() {
        super();
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
    {
        String adminOptions = request.getParameter("submission");
        FlightDAO flightDAO = new FlightDaoImpl();
        PlaneDAO planeDAO = new PlaneDaoImpl();
        String source = null; //to identify the parametrized method
        String dest = null; //
        List<Flight> res = null;
        List<Plane> resAir = null;
        PrintWriter out = response.getWriter();

        if(adminOptions ==null)
        {
            adminOptions = "none";
        }
        System.out.println("\n ===== \nThis is the adminOptions output");

        switch(adminOptions)
        {
            case "listSources":
            {
                System.out.println("Getting all sources and destinations");
                String props = "source,destination";
                res = flightDAO.listFlights();

                request.setAttribute("props", props);
                request.setAttribute("List", res);

                request.getRequestDispatcher("admin_list.jsp").forward(request,response);
                break;
            }
            case "listAirlines":
            {

```

```

        System.out.println("Getting all airlines");
        String props = "airline";
        resAir = planeDAO.listAirlines();

        request.setAttribute("props", props);
        request.setAttribute("List", resAir);

    request.getRequestDispatcher("admin_list.jsp").forward(request,response);
        break;
    }
    default:
    {
        System.out.println("Getting all flights");
        String props = "id,source,destination,date,seats";
        res = flightDAO.listFlights();

        request.setAttribute("props", props);
        request.setAttribute("List", res);

    request.getRequestDispatcher("admin_list.jsp").forward(request,response);
        break;
    }
}

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

```

ValidateFlight

```

package com.controllers;

import java.io.IOException;

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

```

```

import com.DAO.FlightDaoImpl;
import com.dto.Flight;

/**
 * Servlet implementation class ValidateUser
 */
public class ValidateFlight extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private FlightDaoImpl flightDao;
    /**
     * @see HttpServlet#HttpServlet()
     */
    public ValidateFlight() {
        super();
        flightDao = new FlightDaoImpl();
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
    {
        System.out.println("The selection just hit Validate
flight:"+request.getParameter("selection"));
        //RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/registration.jsp");
        request.setAttribute("selection", request.getParameter("selection"));
        request.getRequestDispatcher("registration.jsp").forward(request, response);

    }
}

```

Confirmation

```
package com.controllers;

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

/**
 * Servlet implementation class Confirmation
 */
public class Confirmation extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public Confirmation() {
        super();
        // TODO Auto-generated constructor stub
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        response.sendRedirect("paymentcomplete.jsp");
    }
}
```

PasswordChange

```
package com.controllers;
```

```

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

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

import com.DAO.AdminDAO;
import com.DAO.AdminDaoImpl;
import com.dto.Admin;

/**
 * Servlet implementation class PasswordChange
 */

public class PasswordChange extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public PasswordChange() {
        super();
        // TODO Auto-generated constructor stub
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        List<Admin> admins = null;
        AdminDAO adminCheck = new AdminDaoImpl();
        String oldpass = request.getParameter("oldpass");
        String newpass = request.getParameter("newpass");
        PrintWriter out = response.getWriter();

```

```

        admins = adminCheck.getAdmins();

        try
        {
            for(Admin a : admins)
            {
                if(a.getPass().equals(oldpass));
                {
                    adminCheck.updateAdmin(a.getUser(), newpass);
                    out.println("Your password has been updated successfully!");
                    out.println("<br><a href=\"\"/flyaway\"\">Book a flight and
register as passenger</a>");
                }
            }
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
    }
}

```

ValidatePassenger

```

package com.controllers;

import java.io.IOException;

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

import com.DAO.FlightDaoImpl;
import com.DAO.PassengerDaoImpl;
import com.dto.Flight;
import com.dto.Passenger;

```

```

/**

```

```

* Servlet implementation class ValidateUser
*/
public class ValidatePassenger extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private PassengerDaoImpl passDao;
    private FlightDaoImpl flightDao;
    /**
     * @see HttpServlet#HttpServlet()
     */
    public ValidatePassenger() {
        super();
        passDao = new PassengerDaoImpl();
        flightDao = new FlightDaoImpl();
    }

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

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
    {
        Passenger pass = new Passenger();
        Flight flight = new Flight();
        FlightDaoImpl fdao = new FlightDaoImpl();

        System.out.println("This is the submission
parameter:"+request.getParameter("submission"));
        pass.setFirstname(request.getParameter("firstname"));
        pass.setLastname(request.getParameter("lastname"));
        pass.setFlight_id(Integer.parseInt(request.getParameter("submission")));
        pass.setSeats_purchased(Integer.parseInt(request.getParameter("seats")));

        flight = fdao.searchFlightById(Integer.parseInt(request.getParameter("submission")));
        System.out.println("\n =====\n DEBUG Back to post after single object
search");

        request.setAttribute("flight", flight);
        RequestDispatcher dispatcher = request.getRequestDispatcher("summary.jsp");
        dispatcher.forward(request, response);
    }
}

```



```
}
```

```
}
```

```
AdminDaoImpl
```

```
package com.DAO;
```

```
import java.util.List;
```

```
import javax.persistence.TypedQuery;
```

```
import org.hibernate.Session;
```

```
import org.hibernate.SessionFactory;
```

```
import org.hibernate.Transaction;
```

```
import org.hibernate.boot.Metadata;
```

```
import org.hibernate.boot.MetadataSources;
```

```
import org.hibernate.boot.registry.StandardServiceRegistry;
```

```
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
```

```
import com.dto.Admin;
```

```
public class AdminDaoImpl implements AdminDAO
```

```
{
```

```
    private SessionFactory factory;
```

```
        public AdminDaoImpl()
```

```
        {
```

```
            System.out.println("Config about to load");
```

```
            StandardServiceRegistry ssr = new
```

```
StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();
```

```
            Metadata meta = new MetadataSources(ssr).getMetadataBuilder().build();
```

```
            factory = meta.getSessionFactoryBuilder().build();
```

```
            System.out.println("Config loaded");
```

```
        }
```

```
        @Override
```

```
        public String addAdmin(Admin admin) {
```

```
            String administrator = null;
```

```
            Session session = factory.openSession();
```

```
            Transaction txn = session.beginTransaction();
```

```
            administrator = (String) session.save(admin);
```

```
            txn.commit();
```

```
            session.close();
```

```
            return administrator;
```

```
        }
```

```
        @Override
```

```

    public void updateAdmin(String admin, String pass)
    {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        Admin administrator = session.get(Admin.class, admin);
        administrator.setPass(pass);
        session.update(administrator);
        txn.commit();
        session.close();
    }

    @Override
    public void deleteAdmin(String admin) {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        Admin administrator = session.get(Admin.class, admin);
        session.delete(admin);
        txn.commit();
        session.close();
    }

    @Override
    public List<Admin> getAdmins() {
        List<Admin> admins = null;
        Session session = factory.openSession();

        Transaction txn = session.beginTransaction();
        String hql = "FROM Admin";

        TypedQuery<Admin> query = session.createQuery(hql);
        admins = query.getResultList();
        return admins;
    }
}

```

FlightDaoImpl

```

package com.DAO;

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

import javax.persistence.EntityManager;
import javax.persistence.TypedQuery;

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

```

```

import org.hibernate.boot.Metadata;
import org.hibernate.boot.MetadataSources;
import org.hibernate.boot.registry.StandardServiceRegistry;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;
import org.hibernate.query.Query;

import com.dto.Flight;

public class FlightDaoImpl implements FlightDAO{
    private SessionFactory factory;

    public FlightDaoImpl()
    {
        System.out.println("Config about to load");
        StandardServiceRegistry ssr = new
StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();
        Metadata meta = new MetadataSources(ssr).getMetadataBuilder().build();
        factory = meta.getSessionFactoryBuilder().build();
        System.out.println("Config loaded");
    }

    @Override
    public Integer addFlight(Flight flight) {
        Integer flight_id = null;
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        flight_id = (Integer) session.save(flight);
        txn.commit();
        session.close();
        return flight_id;
    }

    @Override
    public void updateFlight(Integer flightID, int seats) {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();

        Flight flight = session.get(Flight.class, flightID);
        System.out.println(flight);
        flight.setSeats(seats);
        System.out.println(flight);

        session.update(flight);
        txn.commit();
        session.close();
    }
}

```

```

    }

    @Override
    public void deleteFlight(Integer flightID) {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        Flight flight = session.get(Flight.class, flightID);
        session.delete(flight);
        txn.commit();
        session.close();
    }

    @Override
    public List<Flight> listFlights(String src, String dest)
    {
        List<Flight> flights = null;
        Configuration conf = new Configuration().configure();
        conf.addAnnotatedClass(com.dto.Flight.class);
        StandardServiceRegistryBuilder builder = new
StandardServiceRegistryBuilder().applySettings(conf.getProperties());
        SessionFactory factory = conf.buildSessionFactory(builder.build());

        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();

        String hql = "FROM Flight AS fli";

        TypedQuery<Flight> query = session.createQuery(hql);

        flights = query.getResultList();

        session.close();
        return (List<Flight>)flights;
    }

    @Override
    public List<Flight> listFlights(String src, String dest, String date, int seats)
    {
        List<Flight> flights = null;
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();

        String hql = "FROM Flight AS fli WHERE ((fli.source =:source"+
            ") AND (fli.destination =:destination"+
            ") AND (fli.seats >=:seats"+
            ") AND (fli.date >=:date))";

        Query<Flight> query = session.createQuery(hql);

```

```

        query.setParameter("source", src);
        query.setParameter("destination", dest);
        query.setParameter("seats", seats);
        query.setParameter("date", date);

        flights = query.list();

        //flights = query.getResultList();

        session.close();
        return (List<Flight>)flights;
    }

    @SuppressWarnings("unchecked")
    @Override
    public List<Flight>listFlights()
    {
        List<Flight> flights = null;
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();

        String hql = "From Flight";

        TypedQuery<Flight> query = session.createQuery(hql);

        flights = query.getResultList();
        session.close();
        return (List<Flight>)flights;
    }

    @Override
    public Flight searchFlightById(Integer flightID) {
        Flight flight = new Flight();
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        String hql = "FROM Flight AS fli WHERE fli.id = :flight_id";
        System.out.println("\n ===== DEBUG: "+hql+" =====\n");
        Query<Flight> query = session.createQuery(hql);
        query.setParameter("flight_id", flightID);
        flight = query.getSingleResult();
        System.out.println("\n ===== DEBUG \nLeaving the single search");
        return flight;
    }
}

```

PassengerDaoImpl

```
package com.DAO;

import java.util.List;

import javax.persistence.TypedQuery;

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

import com.dto.Passenger;

public class PassengerDaoImpl implements PassengerDAO
{
    private SessionFactory factory;

    public PassengerDaoImpl()
    {
        System.out.println("Config about to load");
        StandardServiceRegistry ssr = new
StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();
        Metadata meta = new MetadataSources(ssr).getMetadataBuilder().build();
        factory = meta.getSessionFactoryBuilder().build();
        System.out.println("Config loaded");
    }

    @Override
    public Integer addPassenger(Passenger passenger)
    {
        Integer pass_id = null;
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        pass_id = (Integer) session.save(passenger);
        txn.commit();
        session.close();
        return pass_id;
    }

    @Override
    public void updatePassenger(Integer passengerID, int seats) {
        Session session = factory.openSession();
```

```

        Transaction txn = session.beginTransaction();

        Passenger passenger = session.get(Passenger.class, passengerID);
        System.out.println(passenger);
        passenger.setSeats_purchased(seats);
        System.out.println(passenger);

        session.update(passenger);
        txn.commit();
        session.close();
    }

    @Override
    public void deletePassenger(Integer passengerID) {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        Passenger passenger = session.get(Passenger.class, passengerID);
        session.delete(passenger);
        txn.commit();
        session.close();
    }

    @Override
    public List<Passenger> listPassengers(String src, String dest, String date, int seats)
    {
        List<Passenger> passengers = null;
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();

        System.out.println("Source: "+src+
                           "\nDestination: "+dest+
                           "\nDate: "+date+
                           "\nSeats: "+seats);

        String hql = "SELECT fli.source, fli.destination, fli.date, fli.seats FROM Passenger AS fli
WHERE ((fli.source =:source"+
        " ) AND (fli.destination =:destination"+
        " ) AND (fli.seats >=:seats"+
        " ) AND (fli.date >=:date))";

        //System.out.println(sql);
        //Query query = session.createQuery(hql);
        TypedQuery<Passenger> query = session.createQuery(hql);
        query.setParameter("source", src);
        query.setParameter("destination", dest);
    }

```

```

        query.setParameter("seats", seats);
        query.setParameter("date", date);

        //TypedQuery<Passenger> query = session.createSQLQuery(sql);
        System.out.println("Query created");
        passengers = query.getResultList();
        System.out.println("Query completed, leaving the method.");
        session.close();
        return passengers;
    }

    @Override
    public Passenger searchPassengerById(Integer passengerID) {
        Session session = factory.openSession();
        Transaction txn = session.beginTransaction();
        String hql = "FROM Passenger = "+ passengerID;
        TypedQuery<Passenger> query = session.createQuery(hql);
        Passenger passenger = query.getSingleResult();
        return passenger;
    }
}

```

PlaneDaoImpl

```

package com.DAO;

import java.util.List;

import javax.persistence.TypedQuery;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.boot.Metadata;
import org.hibernate.boot.MetadataSources;
import org.hibernate.boot.registry.StandardServiceRegistry;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.query.Query;

import com.dto.Passenger;
import com.dto.Plane;

public class PlaneDaoImpl implements PlaneDAO
{
    private SessionFactory factory;

    public PlaneDaoImpl()
    {

```



```
        System.out.println("Config about to load");
        StandardServiceRegistry ssr = new
StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();
        Metadata meta = new MetadataSources(ssr).getMetadataBuilder().build();
        factory = meta.getSessionFactoryBuilder().build();
        System.out.println("Config loaded");
    }
```

```
@Override
public Integer addPlane(Plane plane) {
    Integer plane_id = null;
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();
    plane_id = (Integer) session.save(plane);
    txn.commit();
    session.close();
    return plane_id;
}
```

```
@Override
public void updatePlane(String airline, Integer flightID)
{
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();

    Plane plane = session.get(Plane.class, airline);
    System.out.println(plane);
    plane.setFlight_id(flightID);
    System.out.println(plane);

    session.update(plane);
    txn.commit();
    session.close();
}
```

```
@Override
public void deletePlane(Integer planeID)
{
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();
    Plane plane = session.get(Plane.class, planeID);
    session.delete(plane);
    txn.commit();
    session.close();
}
```

```

@Override
public List<Plane> listPlanes()
{
    List<Plane> planes = null;
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();

    String hql = "From Plane";

    TypedQuery<Plane> query = session.createQuery(hql);

    planes = query.getResultList();
    session.close();
    return planes;
}

@Override
public List<Plane> listAirlines()
{
    List<Plane> planes = null;
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();

    String hql = "Select distinct p From Plane p";

    Query<Plane> query = session.createQuery(hql);

    planes = query.list();
    session.close();
    return planes;
}

@Override
public Plane searchPlaneByID(Integer flightID)
{
    Session session = factory.openSession();
    Transaction txn = session.beginTransaction();
    String hql = "FROM Passenger = "+ flightID;
    TypedQuery<Plane> query = session.createQuery(hql);
    Plane plane = query.getSingleResult();
    return plane;
}
}

```

Admin

package com.dto;

import javax.persistence.Column;

```

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="administrators")
public class Admin
{
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="username")
    private String user;
    @Column(name="password")
    private String pass;

    public String getUser() {
        return user;
    }
    public void setUser(String user) {
        this.user = user;
    }
    public String getPass() {
        return pass;
    }
    public void setPass(String pass) {
        this.pass = pass;
    }

    @Override
    public String toString() {
        return "Admin [user=" + user + ", pass=" + pass + "]\n";
    }
}

```

Flight

```

package com.dto;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity

```

```
@Table(name="avail_flights")
public class Flight
{
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="flight_id")
    private Integer id;
    @Column(name="src_point")
    private String source;
    @Column(name="dest_point")
    private String destination;
    @Column(name="travel_date")
    private String date;
    @Column(name="time_to_board")
    private String time;
    @Column(name="price")
    private Double price;
    @Column(name="seat_vacancy")
    private Integer seats;

    public Flight()
    {
    }

    public Flight(String id, String source, String destination, String date, String time, String price,
String seats)
    {
        super();
        this.id = Integer.parseInt(id);
        this.source = source;
        this.destination = destination;
        this.date = date;
        this.time = time;
        this.price = Double.parseDouble(price);
        this.seats = Integer.parseInt(seats);
    }

    public Flight(String id, String source, String destination, String seats, String date)
    {
        super();
        this.source = source;
        this.destination = destination;
        this.seats = Integer.parseInt(seats);
        this.date = date;
    }

    public Flight(String source, String destination, String seats, String date)
    {
    }
}
```

```
        super();
        this.source = source;
        this.destination = destination;
        this.seats = Integer.parseInt(seats);
        this.date = date;
    }

    public Integer getId() {
        return id;
    }
    public void setId(String id) {
        this.id = Integer.parseInt(id);
    }
    public String getSource() {
        return source;
    }
    public void setSource(String source) {
        this.source = source;
    }
    public String getDestination() {
        return destination;
    }
    public void setDestination(String destination) {
        this.destination = destination;
    }
    public String getDate() {
        return date;
    }
    public void setDate(String date) {
        this.date = date;
    }
    public String getTime() {
        return time;
    }
    public void setTime(String time) {
        this.time = time;
    }
    public Double getPrice() {
        return price;
    }
    public void setPrice(Double price) {
        this.price = price;
    }
    public Integer getSeats() {
        return seats;
    }
    public void setSeats(Integer seats) {
        this.seats = seats;
    }
}
```

```

    }
    @Override
    public String toString() {
        return "Flight [id=" + id + ", source=" + source + ", destination=" + destination + ",
date=" + date + ", time="
                                + time + ", price=" + price + "]";
    }
}

```

Passenger

```

package com.dto;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="passengers")
public class Passenger
{
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="passenger_id")
    private Integer passenger_id;
    @Column(name="flight_id")
    private Integer flight_id;
    @Column(name="firstname")
    private String firstname;
    @Column(name="lastname")
    private String lastname;
    @Column(name="seats_purchased")
    private Integer seats_purchased;

    public Passenger()
    {
    }

    public Passenger(String firstname, String lastname, int flight_id, int seats_purchased)
    {
        this.firstname = firstname;
        this.lastname = lastname;
        this.flight_id = flight_id;
        this.seats_purchased = seats_purchased;
    }

    public Integer getPassenger_id() {

```

```

        return passenger_id;
    }
    public void setPassenger_id(Integer passenger_id) {
        this.passenger_id = passenger_id;
    }
    public Integer getFlight_id() {
        return flight_id;
    }
    public void setFlight_id(Integer flight_id) {
        this.flight_id = flight_id;
    }
    public String getFirstname() {
        return firstname;
    }
    public void setFirstname(String firstname) {
        this.firstname = firstname;
    }
    public String getLastname() {
        return lastname;
    }
    public void setLastname(String lastname) {
        this.lastname = lastname;
    }
    public int getSeats_purchased() {
        return seats_purchased;
    }
    public void setSeats_purchased(int seats_purchased) {
        this.seats_purchased = seats_purchased;
    }

    @Override
    public String toString() {
        return "Passenger [passenger_id=" + passenger_id + ", flight_id=" + flight_id + ",
firstname=" + firstname
                                + ", lastname=" + lastname + ", seats_purchased=" + seats_purchased
+ "]\n";
    }
}

```

Plane

```
package com.dto;
```

```

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;

```

```

import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="plane")
public class Plane
{
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="flight_id")
    private Integer flight_id;
    @Column(name="plane_id")
    private Integer plane_id;
    @Column(name="airline")
    private String airline;


    public int getFlight_id() {
        return flight_id;
    }
    public void setFlight_id(int flight_id) {
        this.flight_id = flight_id;
    }
    public int getPlane_id() {
        return plane_id;
    }
    public void setPlane_id(int plane_id) {
        this.plane_id = plane_id;
    }
    public String getAirline() {
        return airline;
    }
    public void setAirline(String airline) {
        this.airline = airline;
    }

    @Override
    public String toString() {
        return "Plane [flight_id=" + flight_id + ", plane_id=" + plane_id + ", airline=" + airline +
        "];";
    }
}

```


Screenshots

Admin Page




FlyAway

List all flights

List all sources and destinations

List all airlines

Change password



FlyAway

Username:

Password:

Submit form

Book a flight and register as passenger



FlyAway

id	source	destination	date	seats
1	China	India	2021-09-30 00:00:00	138
2	China	US	2021-09-30 00:00:00	138
3	China	India	2021-10-01 00:00:00	138
4	China	US	2021-10-01 00:00:00	138
5	China	India	2021-10-01 00:00:00	138
6	China	US	2021-10-01 00:00:00	138
7	China	India	2021-10-06 00:00:00	138
8	China	US	2021-10-06 00:00:00	138
9	US	China	2021-10-03 00:00:00	138
10	US	India	2021-10-03 00:00:00	138
11	US	China	2021-10-04 00:00:00	138
12	US	India	2021-10-04 00:00:00	138
13	US	China	2021-10-06 00:00:00	138
14	US	India	2021-10-06 00:00:00	138
15	US	China	2021-10-07 00:00:00	138
16	US	India	2021-10-07 00:00:00	138
17	India	China	2021-09-29 00:00:00	138
18	India	US	2021-09-29 00:00:00	138
19	India	China	2021-09-30 00:00:00	138
20	India	US	2021-09-30 00:00:00	138
21	India	China	2021-10-01 00:00:00	138
22	India	US	2021-10-01 00:00:00	138
23	India	China	2021-09-29 00:00:00	138
24	India	US	2021-09-29 00:00:00	138



FlyAway

source

China
China
China
China
China
China
China
China
US
US
US
US
US
US
US
US
India
India
India
India
India
India
India
India

destination

India
US
India
US
India
US
India
US
China
India
China
India
China
India
China
India
China
US
China
US
China
US
China
US
China
US



FlyAway

airline

Chinese International Airlines
Chinese International Airlines
Chinese International Airlines
International Eagle Airlines
Chinese International Airlines
Chinese International Airlines
Best International Airline
Chinese International Airlines
International Eagle Airlines
Best International Airline
International Eagle Airlines
International Eagle Airlines
International Eagle Airlines
International Eagle Airlines
Chinese International Airlines
International Eagle Airlines
Best International Airline
Best International Airline
Chinese International Airlines
Best International Airline
Best International Airline
Best International Airline
Best International Airline
International Eagle Airlines



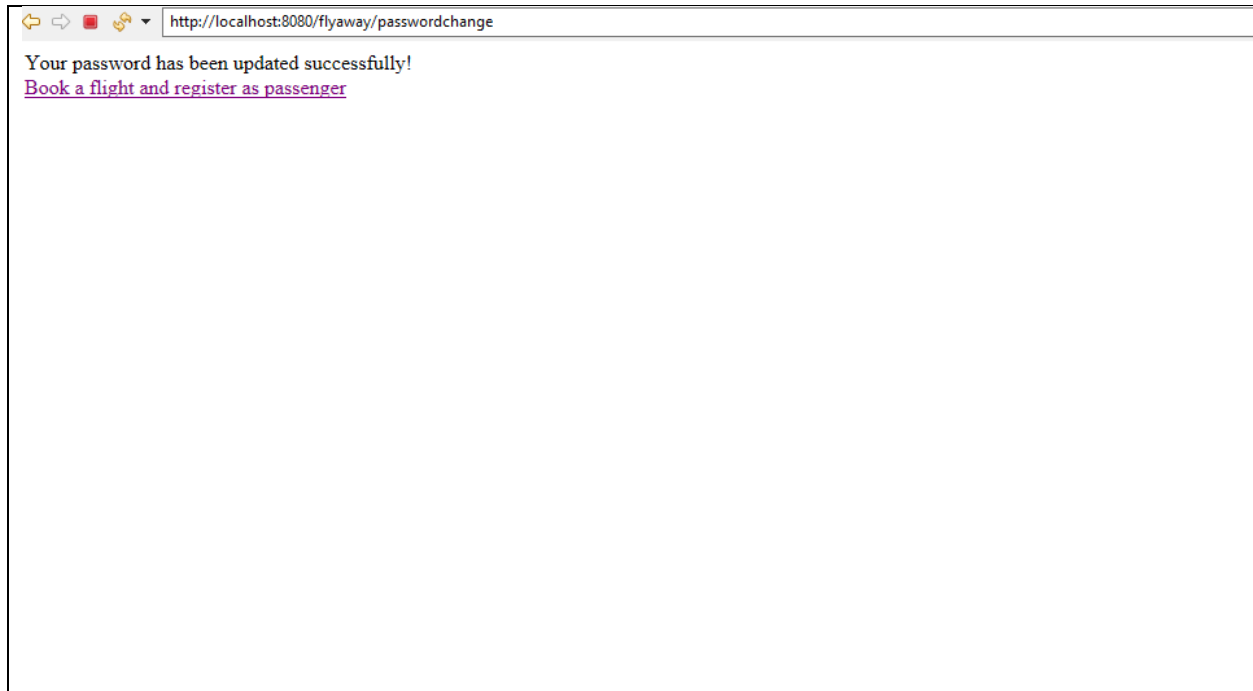
FlyAway

Admin password change

Enter old password:

Enter new password:

[Book a flight and register as passenger](#)



Passenger pages



FlyAway

Fill out the following form to find a flights that matches your schedule

Please select the date you would like to book for :

Source: Destination:

Number of passengers:

[Login as an Administrator](#)



FlyAway

Passenger information

Firstname:

Lastname:

Purchased seats:



FlyAway

Flight#	Source	Destination	Flight_date	Available_seats	Price	Select
9	US	China	2021-10-03 00:00:00	138	1000.0	<input type="button" value="Book It"/>
11	US	China	2021-10-04 00:00:00	138	1100.0	<input type="button" value="Book It"/>
13	US	China	2021-10-06 00:00:00	138	1100.0	<input type="button" value="Book It"/>
15	US	China	2021-10-07 00:00:00	138	900.0	<input type="button" value="Book It"/>



FlyAway

Passenger information

Firstname:

Lastname:

Purchased seats:



FlyAway

Please revise the details of your flight below

Flight #	Source	Destination	Flight date	Available seats	Select
11	US	China	2021-10-04 00:00:00	138	

[Confirm Payment](#)



FlyAway

Thank you for using FlyAway's online booking portal

[Book a new flight](#)

Database hierarchy

- ▼ administrators
 - ▼ Columns
 - ◆ username
 - ◆ password
 - ▶ Indexes
 - ▶ Foreign Keys
 - ▶ Triggers
- ▼ avail_flights
 - ▼ Columns
 - ◆ flight_id
 - ◆ src_point
 - ◆ dest_point
 - ◆ travel_date
 - ◆ time_to_board
 - ◆ price
 - ◆ seat_vacancy
 - ▶ Indexes
 - ▶ Foreign Keys
 - ▶ Triggers
- ▼ hibernate_sequence
 - ▶ Columns
 - ▶ Indexes
 - ▶ Foreign Keys
 - ▶ Triggers
- ▼ passengers
 - ▼ Columns
 - ◆ passenger_id
 - ◆ flight_id
 - ◆ firstname
 - ◆ lastname
 - ◆ seats_purchased
 - ▶ Indexes
 - ▶ Foreign Keys
 - ▶ Triggers
- ▼ plane
 - ▼ Columns
 - ◆ plane_id
 - ◆ flight_id
 - ◆ airline
 - ▶ Indexes