An On-Line Travel Reservation System

CBSD and EC (Programming pathway) coursework

February 2016

Objectives

- To understand the concepts of component software engineering and to use appropriate technologies (e.g., JEE, Spring, etc.) via the development of a distributed application.
- To obtain hands-on experience with both EJB components and web components with a compliant application server.

Scenario

The problem is to design and implement an on-line travel reservation application with web-based functions for both the travellers and the reservation manager. The customer should be able to register him/herself with the reservation site, enter personal profile information (name, address, email, and credit card information), prepare for itineraries, and book flights. The reservation manager should be able to publish and update flight information, and generate inventory report(s).

Required functionality

Reservation manager

- A web page for the manager to log in with username and password.
- One or more web pages for the user to add/cancel/list flights with the following information: airline code/name, flight number, departure location, departure day of the week/time, arrival location, arrival day of the week/time, cost of business class, and cost of economy class ticket. The flight number is a 3-digit number, prefixed with 0's if the actual number is less than 100. For the simplicity reasons, it is predefined that there are 10 seats of business class and 90 seats of economy class seats in each flight (small planes!).
- The airline code must be a two-letter code as described at: http://en.wikipedia.org/wiki/List_of_airline_codes.
- The airport location must be a three-letter code described at: http://www.world-airport-codes.com.
- When a flight is cancelled in the application, all itineraries that are reserved and booked will become cancelled as well.

- The inventory report should contain a summary of all fights in the reservation system that still have any unsold seats, the total number of unsold seats of business class and economy class.
- The user should be able to log out at any stage during his/her session.

On-line travellers

- A web page for a traveller to register him/herself with name, address, username and password, email address, as well as credit card information (optional), including a 16-digit card number and a 4-digit expiration date.
- A web page for a traveller to log in with username and password. Once a traveller is logged in, he/she will be shown with a list of travel itineraries along with the status of each itinerary (reserved, booked, or cancelled). A travel itinerary is a travel arrangement with one or more flights. It has the status reserved when it is created by a traveller, booked when it is paid, and cancelled when it is deleted by the traveller from his/her itinerary list, or one of the flights in the itinerary becomes cancelled by the reservation manager, or payment not received after 2 minutes (simulating the 24 hours holding period in the real world).
- A traveller should be able to create and book a travel itinerary by going through the following steps:
    1. Search for flight information by providing departure/arrival date/time and location, number of passengers, one-way or round trip, and the max number of stops.
    2. A list of available itinerary options will be shown to the on-line traveller, with departure/arrival and cost information. The departure time will be plus or minus hours within the specified departure time. It is possible that one itinerary contains one or more flights from one or more different airlines.
    3. Once the traveller selected an itinerary from the list, he or she has the option to reserve it. Before the traveller can reserve the selected itinerary, he or she must be logged in.
    4. Once the traveller has reserved an itinerary, he or she has the option to book it by providing payment information via credit card. If the credit card information is not on-file for the traveller, he or she will be prompt to enter the credit card information (number and expiration date).
    5. The credit card information has to be validated first before booking. As a simplicity for this project, a valid combination of credit card number and expiration date is defined in such a way that the 4-digit expiration date is defined with a valid *mmdd* format, and as an integer is a denominator of the 16-digit card number.
    6. If the credit card number is invalid, a web page with an error message will be displayed to the user and ask the user to re-enter the credit card information.
    7. Once the credit card is validated, the traveller will be shown with the actual ticket information. The ticket number is automatically generated by the application in the format of `XX-FFF-YYYYYY-ZZZ`, where `XX` is the airline code, `FFF` is the 3-digit flight number, `YYYYYY` is the traveller's login name, and `ZZZ` is a 3-digit sequence number generated by the application. The sequence number is unique for each traveller.
- A traveller is able to cancel an itinerary from his/her itinerary list if it has not been paid. Cancellation after payment is not permitted.

- The traveller should be log out at any stage during his/her session. If the traveller is logged out during the preparation of an itinerary, the next time when the traveller is logged in again, he or she should be able to continue with the existing preparation process.

Required implementation details

The application should be developed using JEE 7.0 and above, or an appropriate server based architecture (needs prior agreement if not JEE or Spring). The application is to be developed in a multi-tier architecture with each tier to be implemented using the technologies and software defined as follows:

- The presentation tier is a web application based on Java Servlet/JSF technologies. The JSF layer may contain functions for validating input fields with predefined string and/or number format (e.g., credit card number and expiration date). The *Glassfish* application server should be used.
- The business tier of the application contains the business logic and processes with connections to the backend persistence layer. You should ensure you use the appropriate EJB types. (A Spring based solution is not acceptable.)
- The persistence tier of the application is a database server used to store all relevant data of the application. You may use any database management system which operates with the Glassfish application server.

Submission

The work should be stored in a private repository on `bitbucket` or `github` and you should grant `keithmannock` cloning access to the repo. You will need to upload the repository address to Moodle well before the submission date.

You should include a `README.md` file in your repository explaining:

1. Overview of your application and a brief description of the system architecture and interconnection among the tiers.
2. How to run your application.
3. Design of the components with reasons of why a particular (bean) technology is used in the application.

You should ensure you have complete API documentation of all your Java classes (using `javadoc`).

Suggested development steps

You are, of course, at liberty to approach this coursework in any way you see fit but here are a list of steps you might like to consider:

1. It is recommended that you follow the standard software development process, whether that be an agile approach, or a more *conventional* route (e.g., from analysis to design, then to implementation and testing).

2. A good starting point would be trying to understand the application requirement and to layout the web pages and their relationship to Servlets, EJBs, and data entities. A good design will make your implementation much easier.

3. You need to divide the application into components according to their tiers. Once you have a thorough design completed and the interface among these components defined, you may proceed to implement and test each component one at a time, without having them interact with each other. Once you have each component fully implemented and tested, you can proceed with the integration.

4. It is recommended to complete your design, and then start implementing your system based on your design (not vice-versa). If anything you found out during the implementation stage that something is wrong with your design, go back to your documentation and correct it before continuing with the implementation.

5. Before delivering the final package, please test the application on another machine, or simply delete your existing EJB application and database files, and then re-deploy. This is to check that everything deploys and works the same as before (and has a chance of working in my setup).

Credits

This coursework has been around for many years and I've forgotten who I worked with on refining the specification. Thanks to those who helped and apologies to those I've omitted to credit.