An On-Line Flight Reservation System in Java

Class Project, SWE 642-001, Spring 2010
Instructor: Nick Duan

Objective:

- To fully understand the concepts of web technologies via the development of a dynamic web application in Java.
- To obtain hands-on experience with various web client and server technologies, including J2EE application server, HTML/JavaScripts/AJAX/GoogleMaps, Java Servlet/Java Server Pages, as well as connectivity to databases
- To gain knowledge and experience with software lifecycle and development skills with design, implementation, testing and integration in a team environment.

Requirement:

1. The project is to implement an on-line flight reservation application with a web front-end for both the travelers 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.

2. The project is to use Apache Tomcat Server version 6.0.x and Java SE version 6.x. Both are freely downloadable from Apache's web site (http://tomcat.apache.org) and Sun's Java web site (http://java.sun.com), respectively.

3. The following functionalities should be provided for the reservation manager or administrator:

   - A web page for the manager to log in with username *manager* and password *managerpw*.

   - 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 less the actual number is less than 100. For simplicity reasons, it is predefined that there are 10 seats of business class and 30 seats of economy class seats in each flight.

   - The airline code must be a two-letter code defined in http://www.tvlon.com/resources/airlinecodes.htm. The airport location must be one of airport with a three-letter code defined in http://www.orbitz.com/App/global/airportCodes.jsp.

   - When a flight is canceled in the application, the all itineraries that are reserved and booked will become canceled 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.

4. The following functionalities should be provided for on-line consumers:

   - A web page for a traveler 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. As a convenience 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, defined with a valid *mmdd* format an integer, is a denominator of the 16-digit card number. The credit card number will be validated via an online validation service.

   - A web page for a traveler to log in with username and password. Once a traveler is logged in, he/she will be shown with a list of travel itineraries along with the status of each itinerary (reserved, booked, or canceled). A travel itinerary is a travel arrangement with one or more flights. It has the status reserved when it is created by a traveler, booked when it is paid, and canceled when it is deleted by the traveler from his/her itinerary list, or one of the flights in the itinerary becomes canceled by the reservation manager, or payment not received after 1 minutes (simulating the 24 hours holding period in the real world).

   - A web page that allows the user to view airport information on Google map, along with routing information displayed as direct lines linking the airport locations associated with the traveler's itinerary. The airport information (name and lat/long) will be provided by the instructor.

   - A traveler is able to create and book an travel itinerary by going through the following steps:

     - Search for flight information by providing departure/arrival date/time and location, number of passengers, one-way or round trip.

     - A list of available itinerary options will be shown to the on-line traveler, 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. For convenient reasons, the max legs an itinerary contains is limited to 2.

     - Once the traveler selected an itinerary from the list, he or she has the option to reserve it. Before the traveler can reserve the selected itinerary, he or she must be logged in.

     - Once the traveler 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 traveler, he or she will be prompt to enter the credit card information (number and expiration date).

- The credit card information has to be validated first before booking. The validation is done via a validation service, which will be provided by the instructor.
- Once the credit card is validated, the traveler 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, FF is the 3-digit flight number, YYYYYY is the traveler's login name, and ZZZ is a 3-digit sequence number generated by the application. The sequence number is unique for each traveler.
- 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.

- A traveler is able to cancel an itinerary from his/her itinerary list if it has not been paid. Cancellation after payment is not permitted.
- The traveler should be log out at any stage during his/her session. If the traveler is logged out during the preparation of an itinerary without completing the purchase, all itinerary information will be lost. The next time when the traveler is logged in, he or she will have to start a new selection process.

5. The application is to be developed in a multi-tier architecture with each tier to be implemented using the technologies and free software defined as follows.

- The presentation tier is a web application consisting of a combination of HTML/JavaScripts/GoogleMaps and JSP components. The JSPs may contain JavaScripts functions for validating input fields with predefined string and/or number format (e.g. credit card number and expiration date).
- The business tier of the application contains the business logic and processes with connections to the backend database via JDBC, implemented using Java Servlets.
- The database tier of the application is a database server used to store all relevant data of the application. The HyperSonic database (HSQLDB, version 1.8), a lightweight database written in Java, should be used to support the application. The software is freely downloadable from http://hsqldb.org. The in-process mode of HSQL should be used for the application. The database files shall be located in a designated location in the Web Archive (war) hierarchy and the application should not hard-wired to specific data file location. It shall find the location of the data files dynamically.
- In addition to the database, the backend of the application also contains a credit card validation server (provided by the instructor). The validation server

application is implemented as a Servlet application and provided by the instructor. Detailed usage and access APIs is available on class web site.

- A Web service shall be implemented to provide flight inventory information via standard Web Service interfaces that supports two query operations. One is to get the total number of seats for both business class and economic class for a given date; the other is to get the total number of seats of both business and economic class for a date range from the current date. The WSDL file for the application will be published by the instructor on the class web site. The inventory Web Service should be accessible by anyone without any security constraints. The Web Service application shall be developed using Apache Axis2 and packaged in a jar file (following Axis2's deployment standard), and bundled within the deliverable war file before project due date.

Team Work:

Team Formation. The project MUST be completed in a team setting with up to four students. Once a project team is formed (by the first deliverable due date), it should not be changed (e.g. adding/deleting/change members). Each team shall have a team lead whose name and ID are going to be used for identify the team and for submitting all deliverables.

Scoring. All members of a group will receive the same score for the project after the final deliverable is completed. However, the instructor reserves the right to lower the scores of certain group member(s) if he/she did not contribute to the project and meet the group expectations.

Project Deliverables:

The project is to be delivered in three (3) phases in sequence at different dates with the final phase due on the date of project presentation (May 4th, 2010). Each deliverable is an augmentation of the previous one, with the final deliverable containing the entire product. Each deliverable contains a standard Web ARchive (war) file. The war file should be delivered via mail to GTA before the due date. Please read the following instructions carefully since the project delivery (both content and format) will affect your project scores directly.

File Naming Convention and Delivery Method:

All deliverable file names shall have the format of *projXXXX.war* or *projwsXXXX.jar* (for the web application and the web services application, respectively), where *XXXX* is the last four digits of the lead person's GMU student ID number. The deliverable war file is to be sent to GTA via email prior or at 3 PM on the due date.

Phase I Deliverable (Due on 3/2/2010):

The deliverable of Phase I is a design document containing the following contents. All HTML pages shall be hyper-linked together for easy navigation. The deliverable shall be bundled in a war file with the aforementioned file naming convention.

- A cover/home page containing your name and student ID. In case of a team, clearly indicates the team leader's name and ID number. In addition, it also contains the travel administrator's username and password. The cover page shall be the default index.html page as the starting page for the project, accessible via project URL: http://serverurl:8080/projXXXX, where XXXX is the last four digit of the lead person's G#.
- Table of contents, with hyperlink to each topic.
- Overview of your application, e.g. what it does and why it is good (be a salesman/woman), and a brief description of the system architecture and interconnection among the tiers.
- Web page design, including a general navigation chart of all pages.
- Database design, with description of each table and an E-R diagram, or a domain diagram to describe the relationships.
- A task list and a brief project schedule, including design, implementation and testing.
- Description of each team member's responsibilities in accordance to the tasks.

Phase II Deliverable (Due on 4/6/2010):

The deliverable of phase II contains the augmented war file submitted in Phase I. The war file should have the following content: JSPs with JavaScripts/GoogleMaps of the application, and database file (HSQL data files) that contain all database tables and the necessary data to support the application. To reduce the file size, only the HSQL data files (.properties, .scripts and .data files) are to be included in the war file.

The design document of Phase I shall be augmented with the following contents:

- Design of web components with a brief description of how the web component technologies are used (Servlet, JSPs, JavaScripts/AJAX, GoogleMaps).
- Page flow and/or workflow of the application.
- Design of database access APIs via JDBC. Use of database access object or framework is also permitted.

The war file shall contain only the presentation tier components.

- All web pages (html or JSPs) along with JavaScripts/AJAX/GoogleMaps components.
- A hyper-link to the default starting page for customers or administrator shall be made on the cover page or the page of table of contents. Each page shall be navigable via the hyperlinks with possible dummy data displayed to show the look-and-feel of the page.

- The web page with Google map with airport locators and the following information: airport name, airport ID, city/county, and state.

Phase III Deliverable (Due on 5/4/2010):

The final deliverable contains the final application with the complete design document and database files, as well as the jar file of the Web Service implementation in Apache Axis2. The design document shall be augmented with the following contents:

- Summary of lessons learned and ideas for future enhancement (use your imagination and creativity).
- A complete API documentation of all your Java classes, generated by *javadoc*. It must be accessible from the home page.

The final war file shall contain the followings:

- GoogleMap applications that displays routing information of a traveler's itinerary. All Servlet and JSP components along with the database files of the application. The war file shall contain all the class files along with the source code. The source code (java files) can be located in the same directory as the class files or in a different directory (e.g. src directory). Proper coding and comment standard should be followed.

The Web Service jar file shall contain the service class files, source code, as well as the necessary WSDL and service.xml file.

Configuration Requirements for Phase III Deliverable

Although the configuration parameters are required only for phase III deliverable, it is recommended you use these configuration parameters during your project development.

1. Please use the username *tomcat* and password *tomcat* as the username/password for credit card validation (using basic authentication). No need to submit the validation server as part of the deliverable. The validation server will be provided by the demo server and running on local host. Use the usergroup *managergroup* to define the user *tomcat* in your app server.
2. J2EE basic authentication shall be used to authenticate the travel manager in your demo application, with the username *manager* with the password *managerpw* for logging in. Configure a usergroup named *managergroup* to include the user *manager* in your Tomcat server.
3. The server url should be configured as http://localhost:8080. This url should also be used as the base url for creating your GoogleMaps key.
4. The file naming convention must be followed.
5. Please ensure that the war file and the jar file deliverables are indeed deployable. Any war/jar file that can't be deployed onto the application server will not be considered for evaluation.

Project Presentation:

The project demo will be conducted on a designated machine provided by the instructor. No personal equipment shall be allowed for the demo. The instructor and/or the GTA will set up the demo for each project team prior to the demo. If doing a group project, at least one person shall be doing the presentation. Each demo shall not take more than 12 minutes. A sign-up sheet of the demo will be provided one week prior to the demo in class. The demo shall include at least the following functions:

- GoogleMaps to display airport location and route information
- Traveler to search and create an itinerary
- Traveler to book a flight using a credit card
- Manager to login and create a flight
- Manager to cancel a flight
- Manager to create an inventory report

Acceptance Policy:

Each deliverable is due at 3:00 PM on the due date. There will be NO exceptions.

Grading policy:

The entire project takes 40% of the class score. An additional 4% may be given to projects completed with exceptional quality and innovation.

Deliverable Phase	Deliverable Contents	Percentage of project
Phase I	Required Content of Design Document	10%
	Proper Deliverable Format (war file)	2%
Phase II	Required Content of Design Document	3%
	Required Content and Functionalities of Web Components, including GoogleMaps	5%
	Database Files	5%
	Proper Deliverable Format (war)	3%
Phase III	Additional Content of Design Document	2%
	Reservation Manager Functions	20%
	Traveler Functions	20%
	Credit Card Validation	5%
	Database Access	10%
	Web Services function (inventory queries)	5%
	Proper Deliverable Format (war, jar)	5%
Presentation	In class Presentation	5%

Suggested Development Steps:

1. It is recommended that you follow the standard software development process, i.e. from analysis to design, then to implementation and testing. A good start would be trying to understand the application requirement and to layout the web pages and their relationship to servlets/JSPs, and data entities. Don't rush to implementation. A good design would make your implementation much easier.
2. You need to divide the application into modules according to their tiers. Once you have a thorough design completed and the interface among these modules defined, you may proceed to implement and test each module one at a time, without having them interact with each other. Once you have each module fully implemented and tested, you can proceed with the integration. This is especially true if you have multiple people in a group and you want to have one person responsible for one or two modules. This would be also a good opportunity to develop your teamwork skills.
3. It is recommended to complete your design, and then start implementing your system based on your design (not vise 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. Update your design document as you redesign and code.
4. Before the delivering the final package, please test the war files and database files on another machine other than the one used for development, and see if everything deploys and works.

Software Required:

JDK 1.6.x (Java SE 6.x):	http://java.sun.com
Apache Tomcat Server:	http://tomcat.apache.org
HyperSonic SQL:	http://hsqldb.org
Apache Axis2:	http://ws.apache.org/axis2