31284 Web Services Development

Assignment – Autumn Session 2017

DUE DATE Week 11, 11:00 PM Monday 29 May 2017

Weighting:	40% of your final grade
Group work:	This is a group assignment, normally to be done in teams of three (3). Other team sizes will only be permitted where the number of students in a class is not divisible by three, and only with the permission of the subject coordinator. Team sizes other than three may have the requirements described in this document adjusted to reflect the number of team members. This assignment must be the original work of your team. Plagiarism, including copying or sharing code between teams or taken from other sources, may result in an allegation of academic misconduct being made against you. Please read the section on Academic Standards.
Late submissions/ Special considerations	Assignments submitted after the deadline will be deducted 4 marks out of 40 per calendar day (including weekends, holidays, working days). If the assignment is submitted more than seven days late (including weekends), the assignment will receive zero. Special consideration, for late submission, must be arranged with the Subject Coordinator before the assignment deadline. When requesting special consideration, documented evidence of illness or misadventure must be provided.
Objectives	This assignment addresses the following subject objectives: 1. Describe and evaluate typical application architectures and requirements for web-based applications 2. Discuss some of the issues of designing web-based applications in an e-business context 3. Describe the roles and uses of web-based applications in organizational contexts 4. Apply concepts of information representation and parsing, in the context of XML and other relevant standards for information interchange 5. Describe and evaluate different technology options available for the development of web-based applications 6. Develop a small, distributed web-based application based on existing software libraries

Assignment Overview

This assignment consists of 2 parts: development of a medium-sized web application incorporating web services and production of a report that contains analysis and reflection.

Part 1 – Web application and Web Services

For Part 1 of the assignment, you are to develop a web application and web services using technologies and techniques taught in this subject as per the guidelines below.
Your task is to develop a web site and web services for allowing people look-up and book available domestic flights in Australia for an airline company (name of your choosing).

The web application allows three types of users:

- Viewer: A non-registered user that could search and view flights using "Main" page search form.
- Customer: A registered user that could perform searching and viewing of flights also booking and cancelling flights (Manage a booking).
- Admin: A flight administrator that has the options (view, edit, and cancel a booking)

I. The web interface should support the following minimum functionality:

- Main page: Shows the main search form for flights. The search results of the "Main" page are displayed on the "Results" page.
  A user can search for flights using the following parameters:
  a- between two cities in Australia (Sydney, Melbourne, Brisbane, Perth …)
  b- The type of flight (either "business", "economy")
  c- Departure date
  d- Return date

  There are three possible scenarios for the "Main" page:
  - First Scenario (No login).
  - Second Scenario (Customer logged in).
  - Third Scenario (Admin logged in).

- Login page: This page is a HTML form where a registered customer can login to flight center. Login form require the following unique customer parameters:
  a- Email (Username).
  b- Password.

  The login form is processed by verifying the customer's login data against existing data in an XML. If login is successful, it takes the customer to the "Main" page at the Second Scenario. After logging in, if the customer performs a search on the "Main" page, the flights list displayed on the "Results" page become clickable or selectable. (If a customer select a flight it should take them to the "Booking" page)

- Register page: This page is an HTML form where a user interested in booking a flight can enter the following details in order to become a flight customer (flight membership):
  a- Full name
  b- Email (username)
  c- Password
  d- Date of Birth (DOB)
  The registration form is processed by adding the new user to an XML. After the viewer is registered as a customer, there are two options:
  - First option: The customer has performed a search on the "Main" page and obtained flight list as result then clicked register link. In this case, the registration form process takes the customer to the "Results" page at the Second Scenario.
  - Second option: The customer has not performed a search on the "Main" page. In this case, the registration process takes the customer to the "Main" page at the Second Scenario.

- Results page: This page shows the search results from the "Main" page. The search results should display a list of available flights with the following information on each flight:
  a- The flight departure date/time and return date/time (only return flights available)
  b- The flight price
  c- The flight available seats
  d- The flight origin and destination cities.
  e- The flight type.
  There are two possible scenarios for the "Results" page: "No login" and "Customer logged in":
  - First Scenario (No login). The results page shows:
    A list of available flights (Non selectable of clickable) obtained from "Main" page.
  - Second Scenario (Customer logged in). The results page shows:
    A list of available flights selectable or clickable displayed as result of a search performed by a customer on the "Main" page. (Selecting/clicking on a flight takes a customer to "Booking" page)

- Booking page: Flight customers can access this page from "Results" page or the "Main" page (if customer is logged in). They can perform the following booking management functionalities:
  a- Make a booking (only if a flight has available seats)
  b- View booking (existing customer booking)
  c- Edit booking (A customer cannot change booking details on the same flight date)
  d- Cancel a booking (A customer cannot cancel a booking on the same flight date)

- Admin page: Only specific registered users (e.g. "airline staff") can access this page from the "Main" page (if admin is logged in) and perform the following functions:
  a- View all bookings
  b- View booking details
  c- Edit a booking
  d- Cancel a booking

Your web application should allow users to:

a- Login/Logout/Register/Back/Cancel/Admin/Booking links or buttons. All pages should have useful navigation options. A user should not be forced to use the back button to go to the previous page, to cancel a form, go back to the "Main" page. A user should be able to navigate from the "Main" page to: ("Admin", "Booking", Register" and "Login" pages). All registered flight user pages should give the user an option to logout.
b- When logged in, a line that shows information about the logged in user is displayed on that page (*i.e. logged in as < name >*)
c- Create and close a flight listing. (The customer who created a listing should have the ability to view it and close it)
d- View a list of own flight listings (available and unavailable). When clicking on a listing display the flights of that listing.

Note: Each flight has a status which is either "available" or "unavailable". A flight status is set to "available" if the number of seats is greater than one (numofseats >=1). Otherwise, it is set to "unavailable" (numofseats = zero)

Note: Customers can only make one booking. In order to make a different booking, a customer must cancel their initial booking or have it cancelled by airline admin staff. In case of booking cancellation, the flight seats should be restored to the flight and its status set to "available".

Note: You have the choice to add more pages to your web application. For example, you have the choice to perform the admin functions and display the results data on the same "Admin" page or create other page(s) once you click on a function link.

Note: Your web application should also maintain details about the customers listing flights. The information to collect is: customer name, customer email, and customer DOB.

Note: behind the scenes, the following information should be stored in XML file(s): The flight listing details, customer's details, and booking details.
Each team can design their own XML format, but it should at least store a unique ID for each flight viewing, booking and customer.

Note: Viewers are the general public who don't need to login. Flight customers are registered flight users. The list of available flights should be visible even to users who have not logged in, however a flight customer (logged in user) should be able to view all of their own flight listings, both available and unavailable.

Note: It is sufficient to create a predefined set of flight customers in XML. For this assignment, 4 customers would suffice. At any given time, a viewer could register as an airline customer and their login details will be added in the customers XML. You also need to predefine airline staff in your XML (maybe separate XML). For this assignment, two airline admin staff will suffice. It is required that you predefine a list of flights between different cities and store them in XML file (separate XML).

II. The web service side, you will need to do the following:

- Create a REST web service allowing a client to:
  Fetch a list of flights in XML format, according to given URL parameters.
  Possible parameters are:
  a- The customer (a username of a flight customer)
  b- The status (available or unavailable) depending on number of seats (numofseats).
  c- Number of listed flights (numofflights).
  d- Number of customers. (numofcustomers)
  Only show listings that can accommodate at least the specified number of customers. Any or all (or none) of these parameters may be specified simultaneously, and your web service must filter the returned list according to all parameters that were given. If no parameters are given, all available listings should be returned.

- Create a SOAP web service that allows customers to:
  a- Login/Logout
  b- Create a listing
  c- Create a booking
  d- View a list of flights, with parameters as above (using SOAP rather than REST)
  e- Close a listing
  User authentication information (for flight customers) must be provided for creating and closing listings, as only flight customers have permission to perform these operations. Retrieving a list of flights can be done without authentication. If the username and password provided are incorrect, the operation should not be performed. If successful, the method for creating a listing should return the ID number of the listing just created. For the method to close a listing, if the specified listing to be closed does not belong to the authenticated person (i.e. it was a listing created by another person), then the operation should not be performed. Note: for this assignment, it is adequate to pass the username and password in an unencrypted form as a SOAP message parameter.

- Create SOAP web service CLIENT as a standalone Java application (i.e. with a main method) that invokes each of the SOAP web service operations. Your SOAP client enable the flight customer to perform the same flight operations allowed by the SOAP web service.

III. Additional requirements:

- All XML files should have a corresponding XML Schema, and you should write custom simple types to describe the format of your data precisely.
- A reasonable amount of HTML output should be generated using XSL Transforms. It is OK if not every page is generated using XSL, but the main data-driven information should be presented using XSL.
- Your web application and web services should perform validation of inputs to prevent system crashes. In the case of your web application, you should display an appropriate error message if the user has inputted incorrect data, allowing them to re-enter the data.

- The data validation should be server-side not client side. (Do not use Java Script for data input validation)
- Your user interface should be well thought out, providing a consistent look and feel on all pages, and providing useful navigation links. The user should be able to get to where he or she wants to go without ever having to click the browser's **back** button.
- Your code should be well designed. Do not place all code into your JSPs. Create reusable Java packages, and reusable beans. These beans can then be reused by both your web application and your web services.
- Your code should be commented and neatly formatted.

Note: that the essential functionality of the assignment must be implemented using technologies and techniques taught in this subject (Java, JSP, XML, XML Schema, XSLT, SOAP, REST, etc.). You can use other technologies (e.g. JavaScript), but they should only be used for non-essential functionality that enhances the overall user experience **(Do not use Java Script for data input validation).** However it is neither required nor expected for you to use technologies that were not taught in this subject.

Note: In teams of three, each person in the team must take responsibility for roughly one third of the work. It is up to your team exactly how you decide to split your work, and it is ultimately every team member's responsibility to ensure that your team is working effectively. One possible breakdown of the work is as follows:

- One student takes charge of the "data access logic", including designing XML file formats, reading and writing XML files, writing XML Schemas, and providing Java classes for these operations this that can be used by other members of the team.
- One student takes charge of the "business logic", including authenticating users, and writing code to perform searches and filtering the data. The business logic should also be provided as Java classes that can be used by other members of the team.
- One student takes charge of the "presentation logic", including all of the XSL transforms to display the web pages, basic error checking on data the user has input, and ensuring web pages are secure (restricted operations are not accessible to users who are not logged in, and there are no passwords in the URL!).
- All three students should share the work of building the web services (REST and SOAP) equally. As well as taking primary responsibility for one part of the assignment, students are also expected to work together and support each other as a team.

IV. Bonus (up to 8 marks):

When completing any of these bonus tasks below, you will have the opportunity to earn an extra marks (the maximum marks for the subject is 100).

1. (up to 2 marks): Allow a customer to cancel their membership with the flight center
2. (up to 2 marks): Allow users to filter the listings shown based on the flight price (minimum and maximum values).
3. (up to 2 marks): Allow an admin to cancel a customer's membership with the flight center.
4. (up to 2 marks): The SOAP Web Service allows customers to cancel a booking.

Part 2 – Report

Your team will need to write a report (2500-3500) words describing the design and architecture of your web application and web services. In your report, you will also need to reflect on your experiences in developing your application, and discuss issues and challenges faced.

Describing the application design and architecture

Each team member should contribute to the report a description of the application architecture of the parts they developed. There should also be a section that describes the overall architecture of the web application. Diagrams are encouraged.

Reflection on experiences

Each team member should write their own reflection of their experiences in developing the application. The reflection may contain (but is not limited to) which topics were most valuable, which topics were most challenging, which topics were necessary to understand to make others fit into place.

Discuss issues and challenges

This part of the report should be a group effort, summarizing what you feel are some of the key issues in the development of web applications and web services, and what tools, design patterns and other support is available to simplify these issues or challenges. This part will require some independent research.

Overall

The report should make it clear as to which team member wrote which section. This can either be done in a separate section of the report stating each person's contribution, or identified throughout the report (for example, by putting team member's names next to report headings).

In writing the report, take care to correctly reference all sources of information that you use, as described in the section on "Academic Standards" below. In particular, note that you should not cut-and-paste significant blocks of text from any other source (including web pages). Short quotes are acceptable, as long as they are clearly identified as quotes, and include an appropriate citation. Paraphrasing text with a citation included is also acceptable. Failure to properly acknowledge your sources may be regarded as plagiarism.

You do not need to explicitly reference lecture or lab notes used in this subject.

The overall report body should be around 2500-3500 words in length, excluding the cover page, table of contents, abstract, bibliography, statement of contributions, etc., if these are used. The report body is an average of 1000 words per person.

Assignment Submission and Demonstration

The two parts of the assignment are to be submitted separately. The web application and web service are to be submitted as a single WAR file, and submitted to PLATE before the deadline. The WAR file must include source code for your system.

The web application must also be demonstrated in **person to your tutor in your lab class during Week 11 starting Monday 29 May 2017 or Week 12 starting Monday 05 June 2017**. All members of the team must be present for the demonstration. Any team member not present during the demonstration (without being granted prior permission) may have up to 50% of the marks for task deducted. During the demonstration, the tutor will ask a variety of questions about the design and development of the application. If you are unable to answer questions about code that you have written, it may impact your marks.

The report is to be submitted to UTSOnline via Turnitin. Turnitin will be used to check the report for material taken from other sources. Failure to properly acknowledge your sources may be regarded as plagiarism.

Self and Peer Assessment

Group projects are a part of this subject because effective groups pool complementary skills and knowledge to achieve better results. This is a highly desired attribute in all professions.

To give you feedback on your ability to work in a group and to modify your group mark for your individual effort, we will ask you to rate your own contributions relative to the contributions of your peers. SPARK is a web-supported program for collecting these ratings. SPARK calculates various factors which are used to adjust your group mark and help you understand your ability and/or effort to work in a group. There is a link to access SPARK from within this subject on UTSOnline.

Initially this assessment task will be given a group mark. However your individual mark for the task will be adjusted according to the ratings from SPARK.

From beginning of (**Week 11, 29 May 2017**) until the beginning of **(Week 12, 05/06/2017)**, you will be asked to rate yourself and your team members against the following 4 criteria:

- Performing tasks properly and on time as agreed by the group
- Overall contribution to meeting the application's functional requirements
- Overall contribution to meeting the application's quality requirements
- Contribution of content for inclusion in the written report

The SPARK ratings collected after the task will be used to adjust your individual mark. Completing SPARK ratings is a requirement of this assessment task and marks will be deducted for non-completion.

Marking Criteria

The broad marking criteria for this assignment are as follows:

Criterion	Weighting (%)
Functional requirements	40%
Code style and quality	20%
Report	40%

The requirements specified in this document are the **minimum** requirements for your solution. In other words, a solution that just meets the requirements documented here will receive 50% of the available marks.

To achieve higher marks, your deliverables need to demonstrate more than the minimum requirements. Note that this does not necessarily mean a larger **quantity** of features– preference will be given to solutions that demonstrate a higher **quality** over those that provide more features but do so poorly.

The marking criteria described here relate to the "group mark" for your project. As mentioned above, your individual mark will be calculated by taking your group mark as a starting point and using information from SPARK to adjust it up or down.

Note: Providing ratings in SPARK is a requirement of this assessment task**. If you do not provide final ratings in SPARK within the required time frame, 5% will be deducted from your own scaled individual mark** (but no deduction for your team members).

Meeting the functional requirements is primarily about being able to demonstrate that your web application and web services are able to perform the tasks required. Marks will also be awarded here for choosing appropriate technologies or approaches for implementation. This will be primarily assessed during the demonstration, and through your tutor asking questions of each team member about their part of the implementation.

Coding style and quality is judged based on the following criteria:

- code modularity and reusability (code duplication kept to absolute minimum)
- appropriate structuring of code into classes, methods, etc, or into templates for XSLT
- code readability (formatting)
- code documentation (comments)
- layout and design (of web pages) simple but professional
- error or warning messages should be human-friendly or machine-friendly as required

For the report, marks will be awarded according to the following criteria:

- the accuracy of the architecture description;
- the quality and depth of reflection; and
- appropriate identification of issues and challenges;
- research to identify what is available to simplify issues and challenges;
- Bibliography (quality of reference sources, correct referencing technique).

Academic Standards

Students are reminded of the principles laid down in the Faculty's Statement of Academic Integrity - Good Practice and Ethics in Informal Assessment found at; <wiki.it.uts.edu.au/Academic_Integrity>

The University's rules regarding academic misconduct can be found at;
<http://www.gsu.uts.edu.au/rules/student/section-16.html>

Assignments in this Subject should be your own original work. The inclusion in assessable work of any material such as code, graphics or essay text obtained from other persons or sources without citation of the source is plagiarism and is a breach of University Rule 16.2.

Asking or paying anyone else to write any part of your assignments is considered cheating. This especially includes using outsourcing websites, and accesses to the subject data are monitored. If you are unsure if your activities (or other student's activities) will be considered cheating, ask your tutor or lecturer for advice.

All text written in assignments must be your own words (or that of your team members), except for short, quoted, and clearly referenced sections. Text copied from web pages, articles or other sources, and not referenced, will be viewed as plagiarism and reported as an allegation of academic misconduct.

Referencing styles may be found at the UTS Library web site. <fttp://www.lib.uts.edu.au/help/referencing>

Any collaboration with another person should be limited to those described in the "Acceptable Behavior" section of the Statement of Academic Integrity. Similarly, group work for this assignment should be the result of collaboration only within the group. Any infringement by a student will be considered a breach of discipline and will be dealt with in accordance with the Rules and By-Laws of the University.

Students are not to give to or receive from any other persons outside their own group copies of their assessable work in any form (hard copy or an electronic file). To do so is 'academic misconduct' and is a breach of University Rule 16.2. That is, assisting other students to cheat or to act dishonestly in a submitted assignment.

Accidental submission of another group's work as your own is considered to be a breach of University Rule 16.2 in that you are acting dishonestly since you should not have a copy of another group's work.
Specific conditions for this assignment:

If any parts of your solution are based on existing code that is found on the web, in a book or generated by any software application apart from NetBeans (or code taken from any other source), you MUST acknowledge the source as a comment in your code .The only exception to this rule is for code supplied in the subject's lab exercises. Acknowledgement is required even if you start with existing code and make modifications. Failure to acknowledge sources will be regarded as unacceptable behavior in the context of the Statement of Academic Integrity referenced above.

Using code that belongs to other students currently or formerly enrolled in this subject (i.e. copying or sharing code) is totally unacceptable behavior and is considered cheating.

The Faculty penalty for proven and serial misconduct of this nature is zero marks for the Subject. For more information go to; <wiki.it.uts.edu.au/start/Academic_Integrity>