UNIVERSITY OF TECHNOLOGY SYDNEY

31284 Web Services Development

Assignment 2 – Autumn Session 2016

DUE DATE:	**Week 10, 9:00am Tuesday 31 May 2016**
Weighting:	40% of your final grade
Groupwork:	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 consideration:	Assignments submitted after the deadline will be deducted 3% out of 40% per working day. 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 organisational 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 **helping people agree on a meeting time, through a poll**.

The web interface should support the following minimum functionality:

- Main page: Shows a list of open polls. For each open poll, the minimum information to be listed is the title of the poll, and the name of the person who created it. Users can click on a poll title on the main page to be taken to the poll detail page.
- Poll detail page: In addition to the poll title and creator's name, the top section of this page should also show the creation date of the poll, the meeting location, and a short description. Below this should be displayed a list of possible meeting times. The user viewing the page should have the ability to enter their name, and then use checkboxes to select which of the available meeting times (possibly multiple) that they are able to attend. When they press Submit, they are then taken to a summary of poll responses page.
- Summary of poll responses page: This page shows the same poll details as on the poll detail page, but now at the bottom of the page it should show a list of ALL users who have participated in this poll, and what their response was for each available meeting time.

Note: behind the scenes, the poll details and user responses should be stored in XML. Each team can design their own XML format, but it should at least store a unique id for each poll, and there should be a way to uniquely identify the user who created the poll (e.g. username).

Your web application should also maintain information about the poll creators and should allow a poll creator to:

- Login and logout.
- Create a new poll.
- Close a poll. Each poll has a status which is either "open" or "closed". When a poll is created, its initial status should be "open". Only the person who created a poll should have the ability to close it.
- View a list of their own existing polls (both open and closed).
- When clicking on a poll, show the poll detail page and summary of poll responses page, as described above. The difference is that a logged-in user (a poll creator) can also view the poll detail page and summary of poll responses for their own closed polls, as well as any open ones.

Note: users are the general public who don't need to login. Poll creators are people who must first log in. The list of **open** polls should be visible even to users who have not logged in, however a poll creator (logged in user) should be able to view all of their own polls, both open and closed. You do not need to allow new poll creators to register. It is sufficient to create a predefined set of poll creators in XML. For this assignment, 3 or 4 poll creators would suffice.

On the web service side, you will need to do the following:

- Create a REST web service allowing a client to fetch a list of polls in XML format, according to given URL parameters. Possible parameters are the *creator* (a unique ID of a poll creator), the *status* (open or closed), or *minResponses* – only show polls that have so far received a minimum number of responses indicated (e.g. *minResponses=2* would only show polls that have received at least 2 responses so far). 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 **open** polls should be returned.
- Create a SOAP web service that allows clients to:
    - create a poll
    - view a list of polls, with parameters as above (but now using SOAP rather than REST)
    - close a poll

  User authentication information (for poll creators) must be provided for creating and closing polls, as only poll creators have permission to perform these operations. Retrieving a list of polls 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 poll should return the id number of the poll just created. For the method to close a poll, if the specified poll to be closed does not belong to the authenticated person (i.e. it was a poll created by another poll creator), 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 a SOAP web service CLIENT as a standalone Java application (i.e. with a main method) that invokes each of the SOAP web service operations.

Some additional requirements are as follows:

- All XML files should have a corresponding XML Schema, and you should write custom simple types to describe the format of your data precisely.
- All HTML output should be generated using XSL Transforms.
- 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.
- 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.

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.

Bonus (up to 4 marks):
When completing any of these bonus tasks, you will have the opportunity to earn an extra 4 marks (the maximum marks for the subject is 100).

- (up to 2 marks): Allow new poll creators to register.
- (up to 2 marks): Allow users to choose how they want data sorted, both in the web pages and web services. For example, polls might be sorted by poll title, by poll creator, or by poll creation date.

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). While you can use other technologies (e.g. JavaScript), they should only be used for non-essential functionality that enhances the overall user experience. However it is neither required nor expected for you to use technologies that were not taught in this subject.

Part 2 – Report

Your team will need to write a report 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, summarising 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 the lab class on **31 May 2016 or 7 June 2016 (Tuesday)**. 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.

At the end of the task (Week 11, 7 June 2016), 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	15%
Coding style and quality	10%
Report	15%

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/start/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 <http://www.lib.uts.edu.au/help/referencing>.

Any collaboration with another person should be limited to those described in the "Acceptable Behaviour" 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 assignment 2:

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 Eclipse (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 behaviour 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 behaviour 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 >.