COMP1688 (2015/16)	Service Oriented Web Applications	Header ID 227634	Contribution 80% of course
Course Leader Dr Kevin McManus	Release Date Monday 18/01/2016		Deadline Date Saturday 09/04/2016
This coursework should take an average student who is up-to-date with tutorial work approximately 40 hours Feedback and grades are normally made available within 15 working days of the coursework deadline			
Learning Outcomes: Use XML technologies for developing web services. Select and employ technologies for building service oriented web applications.			

Plagiarism is presenting somebody else's work as your own. It includes: copying information directly from the Web or books without referencing the material; submitting joint coursework as an individual effort; copying another student's coursework; stealing coursework from another student and submitting it as your own work. Suspected plagiarism will be investigated and if found to have occurred will be dealt with according to the procedures set down by the University. Please see your student handbook for further details of what is / isn't plagiarism. Details are also on the Student Intranet.

All material copied or amended from any source (e.g. internet, books) must be referenced correctly according to the reference style you are using.

Your work will be submitted for electronic plagiarism checking. Any attempt to bypass our plagiarism detection systems will be treated as a severe Assessment Offence.

Coursework Submission Requirements

- An electronic copy of your work for this coursework must be fully uploaded by midnight on the Deadline Date of **Saturday 09/04/2016** using the link on the coursework TeachMat page for COMP1688.
- For this coursework you must submit a single Acrobat PDF document.
- In general, any text in the document must not be an image (i.e. must not be scanned) and would normally be generated from other documents (e.g. MS Office using "Save As .. PDF"). More details are on the IT Support pages. An exception to this is hand written mathematical notation, but when scanning do ensure the file size is not excessive.
- There are limits on the file size (current values are on TeachMat and the Student Intranet).
- Make sure that any files you upload are virus-free and not protected by a password or corrupted otherwise they will be treated as null submissions.
- Your work will be marked online and comments on your work and a provisional grade will be available from the Coursework page on TeachMat. A news item will be posted when the comments are available, and also when the grade is available in BannerWeb.
- You must NOT submit a paper copy of this coursework, or include the Banner header sheet.
- All courseworks must be submitted as above. Under no circumstances can they be accepted by academic staff

The University website has details of the current Coursework Regulations, including details of penalties for late submission, procedures for Extenuating Circumstances, and penalties for Assessment Offences. See http://www2.gre.ac.uk/current-students/regs

Detailed Specification

This assignment is worth 80% of the total marks for this course.

This coursework must be completed individually.

Please read this *entire* specification very carefully so that you are fully aware of the requirements.

Imagine that each of the three London boroughs; Greenwich, Lewisham and Bromley, have independently funded and implemented freecycle websites for residents in their respective Boroughs. Owners of unwanted items are able to post details of items that they wish to donate and upload images to advertise the items. Residents of the boroughs are able to browse and search for items to suit their needs.

Map showing the London boroughs of Greenwich Lewisham and Bromley.

In this coursework you are required to demonstrate heterogeneous web services by creating an n-tier service-driven freecycle portal website covering the three London boroughs of Greenwich, Lewisham and Bromley. The portal will allow a single user search to return results of matching items from multiple boroughs.

To begin with you are to create an XML language FML designed specifically to support freecycling. Your web services will then use your FML for communication.

Your first web services will be developed using the Microsoft SQL Server relational database server `sql-server.cms.gre.ac.uk` provided by the department, on which you are to implement a database populated with freecycle data for the London Borough of Lewisham. Queries against this database must be implemented as web services (web methods) using

.NET technology programmed using C# and running from the Microsoft IIS HTTP server `stuiis.cms.gre.ac.uk` provided by the department.

Your next set of services will be developed using the MySQL relational database server `studb.cms.gre.ac.uk` provided by the department, on which you are to implement a database populated with freecycle data for the Royal Borough of Greenwich. Queries against this database must be implemented using PHP running from the Unix Apache HTTP server `stuweb.cms.gre.ac.uk` provided by the department.

You are then required to create a portal facilitating searches for freecycle items merged across both the Lewisham and Greenwich databases by consuming the web services you have previously created. This is to be implemented using PHP running from the Nginx HTTP server `stu-nginx.cms.gre.ac.uk` provided by the department. This portal is to be further developed to provide a design suitable for use on devices such as smartphones and tablets and enhanced with asynchronous partial page updates (AJAX).

This practical work concludes with creating a third set of services for freecycle items in the London Borough of Bromley using a technology of your choosing other than .NET, MS SQL, PHP or MySQL. These services are then used to extend your portal so that it returns results merged from all 3 boroughs.

In completing this coursework it is recommended that you strictly adhere to the specification and keep it simple.

Functionality to be achieved

This is expressed as a number of levels. The level of functionality implemented in your application will determine the maximum possible mark that you can achieve. The actual mark awarded depends on the quality of your work. Make sure that you fully understand the grading and assessment criteria.

It is recommended that in designing your databases, XML, web services and web sites you should allow for all of the features to be implemented. Starting with level 1 you should incrementally enhance your work to include the next level. Make sure that you do not overwrite a level when proceeding to the next level. Each level must be available for assessment individually.

Level 1 : XML : 8 marks
Design an XML language for freecycling, FML, that can be used to describe items (title, description, images and so on) and the people offering the items (name, location, contact details and so on). Create example XML documents with DTD and XSD schema files that can be used to validate these documents. Your design should be able to support lists of many items, with few details of each item, as well as providing complete details of a single item and the person offering the item. Consider carefully how you will handle images in your design.

Note: Your XML schema should reflect a view of your databases rather than reproduce the schema of your databases. Databases and XML files require rather different approaches to data modelling and storage. How can you make effective use of XML attributes? Handling of

XML attributes in a database is not obvious, nor is handling relationships in XML. Do not forget to include your DTD, XSD and example FML files in your final report.

Level 2 : .NET web services : 12 marks
Using the Microsoft SQL Server RDBMS provided by the department, implement a database to store data about freecycle items for the London Borough of Lewisham. This is to be made available as a number of .NET web methods using C# running from the Microsoft IIS web server provided by the department. Your web services should accept search terms such as type to return results as a list of item matches or as an id to return full details of a single item, using the FML language you created in level 1.

Note: These web services should be able to cope with substring matches. Visual Studio usually defaults to SOAP transport only (configuration dependant), which is a less than optimal transport for XML data, and the scaffolding generated by Visual Studio will be of little use to you. Testing and further levels will be significantly more straightforward if you enable POST, GET and SOAP transports (Web.Config). Consider how your services will scale with many thousands or millions of items in your database. Consider how your results may be sorted.

Level 3 : PHP web services : 12 marks
Repeat level 2 for the Royal Borough of Greenwich using PHP running from the Apache HTTP server and MySQL RDBMS provided by the department. While Visual Studio automatically generates scaffolding forms to test your .NET services you will need to create scaffolding forms yourself to test your PHP services.

Note: This could be implemented using either SOAP or REST (you may wish to attempt both) but it is not so easy to create interoperable SOAP services using PHP. Future levels will be simpler if you create REST services and follow a similar approach to your .NET services.

Level 4 : Portal search : 10 marks
Using PHP running from the Nginx HTTP server provided by the department (not Apache, not IIS) implement a means by which a single search for an item type returns results from both your .NET (Lewisham) and your PHP (Greenwich) web services merged into a single XML document (where there are multiple results). You will need to create scaffolding forms to demonstrate this search. Make sure that you have suitably similar but different entries in your two databases so that you are able to easily demonstrate a search that returns results merged from both databases.

Note: This level returns XML (FML) and not a web page. The Nginx server has no database connection capability. This portal obtains its data from the services you created in levels 2 and 3. This could be coded with loops and print statements but you are advised to use DOM programming.

Level 5 : Portal result display : 10 marks
Extend, but do not replace, your solution to level 4 to create a freecycle portal website running from the Nginx server. A typical search should return a brief format list, suitably displayed using DHTML with pagination to limit results to 10 or so results per page. Each

list format result should allow the user to click to see full details of the item including pictures where available.

Note: Think of the merged services you implemented in level 4 as a data source, much the same as a 3-tier architecture, except in this case the data arrives from your distributed services. The user may wish the results to be sorted, for example, by date or location. It will assist development and demonstration of your work if you include links from this site to all other levels.

Level 6 : XSLT : 10 marks
Re-create level 5 using XSLT to transform your FML documents into DHTML pages similar to the pages you created in Level 5. Provide examples to demonstrate your XSLT using both client-side and server-side XSLT processing.

Note: Server side XSLT does not necessarily need to render the entire page. Images should be included but may be referred to by ID or URL in the XML, much the same as an image element in XHTML You will find this easier if you develop using server-side XSLT processing so that you can easily see and validate the resulting markup. Do not forget to include your XSLT file in your report. Provide links (from the level 5 site) to allow rapid demonstration of both client and server side XSLT processing. Client side processing could be implemented by embedding a processing directive into the XML or using a JavaScript XSLT processor.

Level 7 : Ajax portal : 7 marks
Use asynchronous partial page update (AJAX) techniques to enhance the portal you created in level 5/6. There are a number of ways that AJAX could be used, for example; asynchronous loading of search results could increase the response speed to the user, locations could interact asynchronously with a map display,

Note: Do not replace your implementation of levels 5 or 6 when implementing this level. Client side XSLT could be used to provide partial page rendering.

Level 8 : Bromley : 8 marks
Using technologies other than those you have already used (e.g. Java+Derby, Java+Oracle, Python+PostgreSQL, Ruby+Cassandra, Node.js+MongoDB) provide freecycle services for the London Borough of Bromley. You may wish to implement these services using alternatives to XML (e.g. JSON, CSV). You will need to create scaffolding forms to demonstrate these services. Extend (do not replace) your implementation of previous levels to provide search results merged across the three boroughs; Greenwich, Lewisham and Bromley.

Note: A MongoDB server is provided by the department. Using cloud services external to the department is not only permissible but encouraged and potentially advantageous when implementing this level.

Level 9 : Mobile : 6 marks
Extend your portal to operate effectively on a smart phone. Consider ways in which smart phone technology such as geolocation could add value to your portal.

Note: This could be achieved as either a web page (XHTML 1.1 or HTML 5) that will operate on all smart phones but not have access to all device capability, or as a mobile app which can provide access to device capability but is not cross platform (unless Ionic, Cordova or similar is used).

Level 10 : Report : 12 marks
A brief report describing your implementation of levels 1 to 9, as detailed in the following section under 'Deliverables A'.

Note: This report should contain sufficient information to allow another developer to understand and maintain your work.

Use of tools

You are expected to use web authoring tools such as EditPlus, NetBeans, Visual Studio and XMLspy to aid your productivity. Be careful when using code generators that you understand the code that is being generated.

Remember that your application is required to operate correctly on a range of platforms and browsers.

Borrowed material

In creating your websites you are expected to borrow code, text content, images and so on. All borrowed material *must* be clearly identified. Include comments in your source code to clearly identify what code you have borrowed and where you borrowed it from (even if you have adapted the code for your own use). Your code sources must also be identified in your report. Referencing code sources in your report is not sufficient on its own. Copyright *must* be acknowledged where appropriate. Failure to correctly reference your sources may be considered as plagiarism!

Deliverables

A. On Monday 7th March 2016 there will be a peer assessment exercise in which students working in groups of three will assess each others implementation of levels 1 through to 6. A peer assessment sheet is attached to this document, this will be provided in the exercise. This sheet must be completed by you and your peer assessors and handed to your tutor during the peer assessment exercise. Your tutor may ask to see a short demonstration to confirm the accuracy or otherwise of the peer assessment. Your tutor may decide to moderate the peer assessment. Marks from this assessment will contribute to your final overall grade.

Participation in this exercise is compulsory. Non-participation will inevitably result in losing potential marks (unless you have discussed your absence with your tutor).

B. A short report as described in level 10 above submitted by the due date. This report must be a single PDF file containing the following sections **IN THE ORDER** given below. Do not include any other information. Do not include all of your source code.

1. A completed self assessment sheet (see end of this document).
2. A statement of your implementation of the functionality as described in the specification. State what you have implemented and how you have implemented it. If you have not achieved all of a certain level then state what you have not implemented.
3. A description of any bugs in your program (all software has bugs!). Bugs declared in here will lose fewer marks than ones that you don't declare!
4. Reflection on the strengths and weaknesses of your application and development process including discussion of your approach to evaluation and testing. This is your opportunity to draw attention to what you have achieved, why you have done things in a particular way and what you have learned from the process.
5. Brief design documentation including a *diagrammatic* schema for all databases, a list of all files that you have created with a description of what the files are for and supporting UML as appropriate.
6. Clearly labelled source code for levels 1 and 6 (XML, DTD, XSD, XSLT) rendered with readability in mind, e.g. a fixed pitch font, syntax highlighting, no line wrapping.
7. Screen shots of your programs in operation. Provide notes with your screen shots to explain how they illustrate the functionality that you have implemented.

C. After you have submitted your report you are required to attend a viva (demonstration) to examine your systems in operation and answer questions about it. This will be used to both assess the level of functionality and the authenticity of your work. Failure to attend a viva will result in loss of marks. You are expected to be pro-active in arranging a viva with your tutor.

Your tutor will moderate your self assessment (deliverable B.1) during your coursework viva. Marks are available for the accuracy of your self-assessment.

Guide notes on completion of the assessment sheet are included in this document. When completing the assessment sheet you should bear in mind that your tutor is looking for honesty and accuracy.

Be advised that you will be required to set up and run your demonstrations from the specified web servers and database servers. You should therefore make sure that your work is set up and tested well in advance so that you do not waste time trying to make it work during the demonstration time. You are strongly advised to develop your work directly on the specified deployment servers as opposed to working offline and then porting your work.

Assessment Criteria

Marks are awarded for:

The functionality that you have achieved. Have you achieved all specified functionality or only some? How well have you achieved the functionality? Have you incorporated any features that were not explicitly included in the requirements but add value to the site? Have you added features that contravene the specification? Have you added features that were not explicitly included in the requirements but detract from the usability of the site?

The quality of the data modelling; to what extent does the design support the required functionality, is it sufficiently flexible to adapt to a range of queries, is it sufficiently restrictive to permit effective validation?

The quality of the design and of the design documentation. Is the design flexible? Would it be easy to add to or amend the application to support additional functionality? How much of the application is hard coded and how much comes from structured file or database resources?

The scalability of the application. For example, is the database appropriately normalised. Will the system be usable with one thousand entries in the database? Will the system be usable with one million entries in the database? Are queries paginated at the database, in the middleware or at the client?

The usability of the application. Is the application easy to use? Is it obvious to the user at each stage what the user needs to do next? Are all messages to the user clear and unambiguous? Is the layout consistent and easy to read? Is navigation though the application clear and straightforward?

The reliability of the application. For example, if it throws an exception every time the user enters invalid input you will lose marks. Faults that you admit to on your bug list (see deliverables) will be looked on more kindly than those that are not declared.

The security of the application. For example, is the database protected from unauthorised access and alteration, is it open to SQL or script injection. Is sensitive data protected in transit? How difficult is it to hack your application? Security holes that you admit to on your bug list (see deliverables) will be looked on more kindly than those that are not declared.

The quality of your code. Have you included meaningful comments, used sensible naming standards (e.g. for variables, functions and files) and code layout (e.g. indentation to make the structure clear). Is the code well structured or a tangle? Have you clearly identified borrowed code with the original source? How many of your pages pass W3C XHTML validation?

The accessibility of the application. Does the application follow WAI and Section 508 accessibility guidelines? Which guidelines have you considered, WCAG 1 or WCAG2? How well does it conform to the guidelines? Priority A or AAA?

Does your application operate correctly on all of the required browsers? Is the page layout elastic, responsive or adaptive? If any features fail on a particular browser, does it fail gracefully or become unusable? Is it usable without CSS? Without JavaScript? Without images?

Appropriate use of technologies, for example, is user data validated on both the client and the server? Has a sensible choice of validation priority been made? Is the validation effective? The specification is intentionally open so that you can decide to a certain extent how to implement each feature.

The quality of the report. Are all the required sections included and completed appropriately? Is the report lengthy and verbose or is it concise but full of significant information? Is the standard of English appropriate? Does the design match the implementation?

You will fail if any of the following are true:

Your code does not run from the required web and database servers.

You do not attend the viva.

You do not electronically submit your documentation by the deadline

Grading Criteria

The specification is given as ten levels including the report. Marks for each of the ten individual levels are provided with the level specifications above, making a total of 95%. The accuracy of your self assessment is worth up to 5%.

Note that the mark you achieve as defined in each level specification sets the maximum possible mark, you may get a mark lower than the maximum possible for each level that you implement depending on how well meet the assessment criteria. Factors that may be taken into account when awarding a grade are described above in the assessment criteria.

The self assessment sheet below requires that you record a grade for each level as a number between 0 and 10. The weighting for each level is applied later.

7 … 10	1st Class, distinctive, outstanding in all elements.
6	Upper Second Class, meritorious, good overall standard
5	Lower Second Class, adequate, largely meets the requirements
4	Third Class, pass, acceptable, largely achieves the learning outcomes
3	Compensatable fail, not acceptable, achieved some learning outcomes
0 … 2	Fail, does not meet level 6 undergraduate degree standard.

Assessment sheets

The self assessment sheets are to be completed by the student. The factors that should be taken into account when completing the assessment are described above in the grading criteria.

The assessment sheets are an eleven point Likert scale requiring a circle to be drawn around one of the records for each row on the sheet.

0 – missing or may as well be, no real attempt made
1 – some limited evidence of an attempt
2 – fail, some evidence of work but falling way below the required standard
3 – an attempt has been made but insufficient to pass
4 – a bare pass, lacking in essential aspects but not a fail
5 – acceptable, a clear pass but barely acceptable, lacking aspects of the grading criteria
6 – good, largely meets the specification but lacking in quality
7 – excellent, achieves most of the specification
8 – outstanding, achieves all of the specification
9 – exemplary achieves all of the specification to a high standard
10 – faultless, or at least difficult to criticise

The difference between these eleven categories should be perfectly clear. You should seek timely guidance from your tutor if you require clarification. If you find yourself wanting to circle 7 or above then please read ***all*** of the specification document very carefully and consider what is meant by faultless.

Clearly there is a degree of academic judgement in making any assessment. The assessment sheets are intended to help in making an objective assessment so please consider each record carefully. It is in your interest to be honest and accurate in your assessment.

COMP1688 Peer Assessment Sheet for the 201516 Coursework

This sheet must be completed in the peer assessment exercise

Assessed student: _____ **Student ID 000** _____ **Sign**_____

Peer student : _____ **Student ID 000** _____ **Sign**_____

Peer student : _____ **Student ID 000** _____ **Sign**_____

.NET web services
`http://stuiis.cms.gre.ac.uk/`_____

PHP web services
`http://stuweb.cms.gre.ac.uk/~`_____

PHP Portal
`http://stu-nginx.cms.gre.ac.uk/~`_____

Student Use												
Level 1	XML	8	0	1	2	3	4	5	6	7	8	9	10
Level 2	.NET web services	12	0	1	2	3	4	5	6	7	8	9	10
Level 3	PHP web services	12	0	1	2	3	4	5	6	7	8	9	10
Level 4	Portal search	10	0	1	2	3	4	5	6	7	8	9	10
Level 5	Portal result display	10	0	1	2	3	4	5	6	7	8	9	10
Level 6	XSLT	10	0	1	2	3	4	5	6	7	8	9	10
Staff Use												
Self-assessment		5	0	1	2	3	4	5	6	7	8	9	10
Comments												

COMP1688 Self Assessment Sheet for the 201516 Coursework

This sheet must be completed and submitted with your report

Student name: _____ Student ID 000 _____

.NET web services
`http://stuiis.cms.gre.ac.uk/`_____

PHP web services
`http://stuweb.cms.gre.ac.uk/~`_____

PHP Portal
`http://stu-nginx.cms.gre.ac.uk/~`_____

Bromley
`http://`_____

Student Use												
Level 1	XML	8	0	1	2	3	4	5	6	7	8	9	10
Level 2	.NET web services	12	0	1	2	3	4	5	6	7	8	9	10
Level 3	PHP web services	12	0	1	2	3	4	5	6	7	8	9	10
Level 4	Portal search	10	0	1	2	3	4	5	6	7	8	9	10
Level 5	Portal result display	10	0	1	2	3	4	5	6	7	8	9	10
Level 6	XSLT	10	0	1	2	3	4	5	6	7	8	9	10
Level 7	AJAX portal	7	0	1	2	3	4	5	6	7	8	9	10
Level 8	Bromley	8	0	1	2	3	4	5	6	7	8	9	10
Level 9	Mobile	6	0	1	2	3	4	5	6	7	8	9	10
Level 10	Report	12	0	1	2	3	4	5	6	7	8	9	10
Staff Use													
Self-assessment		5	0	1	2	3	4	5	6	7	8	9	10
Comments (continue on next sheet if necessary)													