Course: COMP1687 **Web Application Development**	**Contribution:** **100% of course**
68: Web Application Development – CW– Term 1 - MAC	**PDF file (report)** **ZIP file (code)** **SQL file (database transfer)**
Greenwich Course Coordinator: Dr. Mahtab Hossain	**Due Date: 26/11/2019**
This coursework should take an average student who is up-to-date with tutorial work approximately 50 hours	
Learning Outcomes: Use client-side technologies for building, usable, accessible, standard compliant web pages. Use server-side technologies for building secure, stateful, database driven web applications. Describe and critically discuss the design, engineering, legal, social, ethical and professional issues and considerations involved in web application development.	

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 or buying coursework from someone else 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.

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 (local time) on the Deadline Date.
- For this coursework you must submit a single Acrobat PDF document of your report. Your report MUST contain the self-assessment sheet as the first page. Your developed "Web Application" should be uploaded as a ZIP file that is hosted inside *localhost* for access. One SQL file for transferring your database should be provided as well (inside the ZIP file).
- 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"). 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. Please ensure that you meet that.
- 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. The grade will be made available in the portal.
- You must NOT submit a paper copy of this coursework.
- All coursework must be submitted as above.

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 coursework is worth 100% 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.

You are to create a web site for Time Banking management system [not a complete system by any means – only with partial specific functionalities as outlined below] for a local community. A Time Bank is generally a community-run system where the time (e.g., hours) to deliver a particular task/service is the unit of the account – not the currency as in traditional bank. It tries to convert unpaid time into a valuable commodity that is targeted at building social capital, and greater community bonding. The functional requirements of this web site are outlined as: visitors to the site will be able to register with the site as members and provide information about their skills and required services/tasks via posts. Casual visitors to the site will be able to search through the posts to see if any posts are of interest to them. While any visitor can search through the various posts, full details of time bankers (i.e., registered members) and their posted tasks'/services' details will only be available after registering to this web site.

To implement the site you ***must*** use XHTML 1.1/HTML5, CSS and JavaScript for the client-side coding. HTML5 extensions are permitted but should be identified and justified in the documentation. PHP ***must*** be used for the server side coding. The site ***must*** run from the Unix Apache web server stuweb.cms.gre.ac.uk and the MySQL database server mysql.cms.gre.ac.uk provided by the department.

In completing this coursework it is recommended that you *strictly adhere to the specification* and *keep it simple*. When designing your web pages you are expected to give serious consideration to usability and how CSS and JavaScript can be used to enhance usability. Your sites are required to display and operate correctly on all popular web browsers, i.e. Mozilla Firefox, Chrome, MS IE, and so on, and on all platforms; desktop, laptop, tablet and cellphone. Client side (JavaScript) and server side (PHP) scripts ***must*** be used to validate input data from ***all*** forms. Client side validation may be supplemented with HTML5 elements. Your site ***must*** operate correctly with and without JavaScript and CSS. Please take time to read carefully the grading and assessment criteria that follow.

Functionality to be achieved

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

It is recommended that in designing your websites (and databases) you should allow for all of the features to be implemented. In building the websites each level should be attempted in increasing order. Starting with level 1 you should incrementally enhance your work to include the next level.

Level 1: Account creation: 15 marks
Create an XHTML/HTML5 form that allows visitors to create a member's account. The form must require only 5 pieces of information from the applicant; their chosen username, their chosen password, their email address, a CAPTCHA string, and their skills [e.g., plumbing, teaching, programming, etc. or even no skills]. Account details are to be stored in your MySQL database. On successfully completing this form the applicant must be presented with the verification form (level 2).

The system must prevent duplicate usernames being chosen. Do not use the email address as a username (to avoid spamming). Newly created accounts must remain inactive until they are verified by handshaking the email details (level 2).

Note: Authentication credentials should be protected from interception in transit. Member passwords should be stored in the database in an encrypted format. You may use one of the many open source CAPTCHA systems or write your own (beware reCAPTCHA is less than friendly and presents usability issues without JavaScript).

Level 2: Verify and Authentication: 15 marks
Account verification will require sending a message to the email address provided in level 1. This email message should include a 5 character activation code which enables an applicant to activate their newly created account. Members should not be allowed to make use of the site's member facilities until they have verified their account.

You are to create an XHTML/HTML5 account verification form providing only a single field allowing an applicant to enter the activation code retrieved from their email. On successful verification, an initial time-bank credit of 100 will be given to the newly registered member. Accounts must remain inactive until the correct information is provided.

For authentication, provide an XHTML/HTML5 login form that allows returning verified members to authenticate with the site using their username and password. These credentials should be compared with the information recorded in your MySQL database. Users who have already applied to be members but have not yet verified their account and attempt to log in using this from must be presented with the verification.

Note: You will need to initiate some form of session state to prevent unauthorised access to member activity within the site. Authentication credentials should be protected from interception in transit. You will find it useful to implement some form of logout mechanism if you are to be able to test this login process.

Level 3: Member post: 15 marks
Provide XHTML/HTML5 form that allows authenticated members to post information about their needed services/tasks. These forms should allow members to upload structured information including the service/task type [plumbing, accounting, baby-sitting, etc.], status [open/completed], skills required [none, programming, math, etc.], service's/task's location, and the credit/point that may be awarded to the person who completes it. The system must allow for members to post one or more service/task posts. The system must provide for editing [for example, after completion, the member may change the status field from open to completed], and also keep provisions for deleting of a post. There should be provisions to allow authenticated members to upload images to accompany their post (e.g., images related to the tasks/services required, e.g., broken boiler, clogged kitchen sink, etc.). The system

must allow for multiple images to be uploaded (not necessarily all together, one at a time may be simpler) against each post with a means of deleting or replacing uploaded images.

Note: Editing information is not the same as re-entering information, the member may only be seeking to correct a spelling mistake and so should not be required to re-enter complete data. Images may be stored as either files on the server or as records in the MySQL database. Remember that images need suitable alternate text content when included in a web page (the file name is seldom appropriate as alternate text). Do not force the member to upload an image, each member may have zero or more uploaded images.

Some of this data is best handled with HTML form elements other than input type text. Consider carefully the most appropriate form element to gather this user input. Remember that some characters (notably the apostrophe) can cause problems with your SQL strings.

Level 4: Member search: 10 marks
Provide a means for casual visitors (i.e., not registered) to enter a skill or location or combined [skill+location] to search for matching or nearby tasks'/services' posts. Search results must be returned in a paginated list brief format where each entry in the list can be clicked to, if they are registered as a member, show full details of the matching time banker, otherwise (if not registered) redirect to the level 3 login form.

Search results should be filterable to sort by, for example, Euclidean distance from the casual visitor's input location, exclude posts without images, etc.

Note: A casual visitor should not be expected to authenticate with the site. This form can attract members so should be clearly visible from the home page. You should not expect search terms to be an exact match. Consider carefully how you may sensibly match to location. Result lists may become lengthy (e.g, searching with an empty string may return all the existing entries of your database), and therefore must be paginated. Make sure that you have sufficient items in your database to demonstrate pagination.

Level 5: Cookie: 5 marks
Use a cookie to remember the member's username but not the password. In addition use a cookie to remember the last search term. Sites that store cookies must conform to EU cookie law (e-privacy directive).

Note: This could be implemented using either client side or server side code. While cookie handling is arguably implemented rather better in PHP than JavaScript, you must bear in mind that the cookie is stored on the client and server side manipulation of a cookie can therefore be problematic.

Level 6: Usability: 10 marks
The developed application should incorporate the contemporary issues of Web development in terms of user experience (i.e. usability). For example, is it responsive/adaptive to address device heterogeneity issues for access? How is the Web interface's information architecture in terms of finding relevant items and navigation? Have contemporary issues of Web development been addressed?

Level 7: LSEPi consideration: 25 marks
You have used Cookie in level 5. If Cookie is used in a European Union website, it generally displays a ‘Cookie Consent’ notice (e-privacy directive) to the visitor. In the context of this scenario [Cookie], please discuss the Legal, Social, Ethical and Professional (LSEP) issues and considerations (LSEPic) and implications that may arise in Web development that uses Cookie. You may wish to extend your discussion to include Political, Philosophical and Economic issues and considerations (PPEic) as well. Your discussion should address both general and specific LSEPi and PPEi in the context of Cookie usage in a Web application.

This discussion must be suitably structured, written in your own words, and develop a clear narrative or argument using appropriate language. All assertions are expected to be supported by references (using Greenwich Harvard formatting) or otherwise justified.

The discussion should be within 2,000 words plus or minus 5% (100 words) not including the title page or any preamble and not including the references.

If you are in any way unclear about this specification you should discuss this with your tutor.

Use of tools

You are free to use web authoring tools such as Brackets or NetBeans to aid your productivity. If you wish, you may make use of WYSIWYG tools such as Dreamweaver or Expression Web. Do not become distracted into spending valuable time on the appearance of your work or gold plating the specification. Be careful when using code generators that you understand the code that is being generated.

Remember that your application must operate correctly in a range of desktop, pad and mobile browsers such as Mozilla, Chrome, Android and Internet Explorer, with and without JavaScript enabled.

Borrowed material

In creating your websites you are expected to borrow code, text content, images and so on. Be careful when using borrowed code such as PHP or CSS frameworks (e.g. CodeIgniter, Fusebox, Baseguide, Bootstrap) that you understand the code and it functions correctly from the specified deployment server. 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 submission. Referencing code sources in your submission 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. A PDF document submitted by the due date 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 cover page.
2. A completed self assessment sheet (see Page 10 of this document).
3. A statement of the functionality that you have achieved as described in the specification. If you have not achieved all of a certain level then specify the sub-parts of it e.g. all of level 1, 2 and 3 plus some of level 4.
4. 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!
5. Level 7's discussion.

B. A ZIP file containing your web application code.

C. An SQL file for transferring/migrating your local database to another server/machine. Details can be found near the end of this document under the title "Database Server" of "Important details for locally hosted web and database server" section. You can include this SQL file inside the code's ZIP file.

D. After you have submitted your report you are required to attend a viva to examine your system in operation and answer question about it. This will be used to both assess the level of functionality and the authenticity of your work.

Your tutor will moderate your self assessment (deliverable A.2) 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 application from the specified web server and database server. 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 viva 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

In terms of the developed Web application (Level 1-6):

The 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 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 accessibility of the application. Does the application try to follow WAI and Section 508 accessibility guidelines?

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 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 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?

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.

You MUST ensure the following:

Your code should run from the required web and database servers as mentioned in this specification.
Attend the demonstration viva with your tutor.
Submit the coursework documentation by the deadline electronically.

In terms of the submitted LSEPi discussion (Level 7):

Marks are awarded in ***equal*** measure for the following ***five*** criteria.

The quality of the language used. Is the language clear and unambiguous or is it difficult to follow, with poor sentence structure and grammatical errors? Is the language at an appropriate level using a technical vocabulary or is it too simplistic or overly familiar?

The structure. Is the text in a single paragraph or is it organised into sections and subsections? Are the paragraphs and sections sensibly chosen? Is the text overly compartmentalised? Is there a narrative or argument or is it merely a collection of facts and assertions? Is the content contextualised or just a collection of bullet points?

The quality of the content. Is there adequate discussion of legal, social, ethical and professional matters or perhaps only one of these four? Are they discussed superficially or does the discussion have depth and demonstrate understanding of how these aspects are interrelated? Are more interesting or important aspects discussed?

The scope of the discussion. Does the account provide generic, specific and reflective discussion of the subject or only one of these three? Are these discussed separately or contextualised to demonstrate understanding and insight?

The academic standard of the writing. Is it entirely original or is there evidence of pasted content? Are citations provided to support facts and assertions? Are references provided to match the citations? Are all references cited in the article? Are the references appropriately formatted?

Grading Criteria

The specification is given as seven levels. Marks for each of the seven 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.

8 … 10	Exceptional in all elements.
7	Excellent in all elements.
6	Very good overall standard
5	Good, largely meets the requirements
4	Satisfactory, achieves the learning outcomes
3	Fail, not acceptable, achieved some learning outcomes
0 … 2	Fail, does not meet level 6 undergraduate degree standard.

COMP1687 Self Assessment Sheet for the 201920 Coursework
This sheet must be completed and submitted with your coursework

Student name: _____ Student ID 00 _____

URL
`http://localhost/`_____

Student Use												
		Total Mark	Please circle ONE of the grades (0 to 10) for each level below [Level 1 to Level 6]									
Level 1	Account creation	15	0	1	2	3	4	5	6	7	8	9	10
Level 2	Verify/Authentication	15	0	1	2	3	4	5	6	7	8	9	10
Level 3	Post	15	0	1	2	3	4	5	6	7	8	9	10
Level 4	Search	10	0	1	2	3	4	5	6	7	8	9	10
Level 5	Cookie	5	0	1	2	3	4	5	6	7	8	9	10
Level 6	Usability	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
Level 7	LSEPi discussion	25										
Comments												

Important details for locally hosted web and database server

1. Web Server

If /var/www/html/ is the ServerRoot (it can be any other folder depending on the installed Apache distribution), make sure all subfolders inside it have both "read" and "executable" permissions, e.g., if ha07 is a subfolder inside it, its property may look like:
drwxr-xr-x 2 root root 4096 Oct 4 10:52 ha07/

All your coursework related files should at least have read permission enabled. If "info.php" is a file, its property may look like:
-rw-r--r-- 1 root root 41 Oct 3 23:05 info.php

For viewing info.php (if it's inside the folder ha07) using Web Browser, one should type:
http://localhost/ha07/info.php

*You should upload the ha07.zip, i.e., code of your web application (**Deliverables B**). You can see that all the subfolders and files within it will be inside that compressed file. If you are saving the images in Web Server (not inside database) – those images should also be inside this folder of your developed Web application. Remember to replace ha07 by your own username, and use it as a subfolder @ ServerRoot.*

2. Database Server

MySQL Administrator details (Saved inside a configuration file of code, dbConnect.php):
Database Host: localhost
Database name: ha07
username: root
password: root

Ensure that you keep the above administrative particulars inside a configuration file of your code, and name it as dbConnect.php (NO EXCEPTION)!!!

You should create the MySQL database, and manage the necessary tables inside it. Please ensure that your database name is ***ha07***.

For the locally hosted database server, you are required to transfer the whole database in an SQL file (***Deliverables C***) with the following requirements. Please follow the instructions carefully.

First, create a database named 'ha07' in your database server (assuming the relevant MySQL database tools are already installed). The creation of 'ha07' database can be easily done by logging into your local MySQL database server. Assuming a root account, the following command can be used to log into the mysql shell by entering root username's password:
mysql –u root –p
Then, create the *ha07* database for your tables using the shell:
mysql> create database ha07;

All your related tables of the web application should be associated with the database ***ha07*** (NO EXCEPTIONS). Follow the lecture materials for creating the tables. Later on, you need to transfer this database as an SQL file by issuing the following command (+giving password

of root username):
mysqldump -u root -p ha07 > ha07_backup.sql

This ***ha07_backup.sql*** file you need to upload as well as part of your submission (***Deliverables C***). By doing so, your local database can be transferred seamlessly to other MySQL servers' database named "ha07" by your tutor or anyone with the following command (+giving password of root username @ their servers):
mysql -u root -p ha07 < ha07_backup.sql