CSY2028: Web Programming

Due for Issue (week commencing):	**Thursday 19th November 2015**	**Last Date for Submission:**	**Sunday 14th January 2017 23:59:59**
Agreed Date for late submission:		Module Tutor:	Thomas Butler
Student ID:			

Aspect (& weighting)	Excellent A	Good B	Satisfactory C	Needs some more work D	Needs much more work F
Functionality (35%)					
Testing and Error Handling (15%)					
Technical Documentation (20%)					
Evaluation (15%)					
Code Quality and Efficiency (10%)					
Demonstration (5%)					

Specific aspects of the assignment that the marker likes:	Specific aspects of the assignment that need more work:

Tutor's Signature:		Date:		Grade:	

The University of Northampton Policy on Plagiarism & Mitigating Circumstances will be strictly implemented. By submitting this assignment you are asserting that this submission is entirely your own/individual work.

CSY2028 Web Programming

PHP & MySQL Assignment 1

Aims & Objective

The purpose of this assignment is to assess your ability to create a database driven website using PHP and MySQL.

Brief

You are a back end developer for web development agency. A local newspaper wants you to build a website which they can post news articles to. The front end developer has supplied an HTML layout with the relevant CSS and Images. This has been signed off by the client and should be used for the look of the website.

Your task is to implement functionality into the website and allow users at the newspaper to post news articles. Each news article will have a title, a date, an author and the article's text. The company should be able to log in to a secure administration area and add news stories, entering all the information for the article, making it visible to the public.

Each article will fall into a category such as Local News, Sport, Technology or Business. Categories should not be limited to these choices and new categories should be able to be added via the administration area.

The public will then be able to browse news articles and filter by category, for example, seeing all news in the *Sports* category.

Users should be able to add comments to each news article. The comments for each news article should be visible below the article text and comments should only be visible on the news article they were posted to. When a user posts a comment, their name, email address, comment text and date the comment was posted should be logged and displayed on the page.

Along with the working website, you must provide technical documentation so that other developers your company can easily work on the website.

Basic Requirements (Grades D- to C-)

The system must use the layout that was supplied by the designer.

1. Have a password protected administration area that has functionality for: *(20 functionality marks)*
   a) Adding news articles
   b) Adding categories
   c) Assigning articles to categories
   d) Editing an existing article e.g. changing the title or text
   e) Deleting articles from the website
   f) Editing category names
   g) Deleting categories

2. Have a publicly visible front end that allows users to: *(20 functionality marks)*
   a) Browse all the news articles displaying newest first
   b) Views a list of news categories in the drop-down menu in the supplied HTML layout
   c) Click on one of the categories to view news articles in that category (news from other categories should not be visible)

d) Add a comment to a news article
e) See comments added to that article by other users. Comments should be visible on the news article page, and only comments for the selected article should be visible.

There are marks available for usability, you should use select boxes/checkboxes/radio buttons in place of text input where applicable and consider how user friendly the website is. Users should never need to type in or remember numerical IDs.

It is up to you how you structure your application and you may extend it with additional functionality that you think would be useful. Possible enhancements include:

- Moderation of comments. When a comment is added, it's placed in a holding area in the administration area for administrator approval before appearing on the website *(10 functionality marks)*
- Allow uploading an image with the news article and have it appear on the page with the title and news article text *(5 functionality marks)*
- Social media buttons allowing users to easily share news articles *(5 functionality marks)*
- Allow searching news stories by typing in a search term *(5 functionality marks)*
- Allow administrators to manage administrator accounts. Admins should be able to create, update, and delete other admin users who can then log in and post stories. Stories posted should be associated with user who posted them. The news article's author should be visible on the news article's page *(10 functionality marks)*
- Securely store passwords with an appropriate method *(5 functionality marks)*
- Prevent users from adding comments unless they create an account. Users should be able to register an account and log in, once logged in they should be able to comment on an article by entering only their comment text. *(10 functionality marks)*
- Allow comments to be *nested* so that users can comment on other user's comments *(10 functionality marks)*
- Allow users to sign up to be notified via email when new articles are posted *(5 functionality marks) n.b. you will not be able to test this but you can include the code*
- The ability to click on a user and see any comment they have made *(5 functionality marks)*
- The ability to see all news articles posted by a specific author *(5 functionality marks)*

There are marks available for code quality, you should try to avoid repeated code and use Object-Oriented Programming as well as valid HTML and CSS.

Submission Guidelines

1. Your supplied website must be built using the *Vagrant* Virtual Machine used in class. You should zip the *websites* directory and make sure that the file *database.sql* is included. You must halt the virtual machine before submission to trigger a database export.

2. The company's server uses PHP 7.1. The website you build must work on this version and not use functions that have been removed or deprecated (e.g. old mysql functions).

3. Your website must use the layout supplied by the designer. You may make changes to it and extend the CSS but the overall layout should remain the same.

4. Excluding the supplied layout and lecture notes, any code you submit which you did not write **must be referenced and you must make it clear in your report which sections of the code you didn't write and where you found it.**

5. This is an **individual assignment** and any code you submit should be written by you unless properly referenced. This is not a group project and any work submitted should be your own. The University of Northampton Policy on Plagiarism & Mitigating Circumstances will be strictly implemented. By submitting this assignment you are asserting that this submission is entirely your own/individual work.

Failure to follow the submission guidelines may result in a capped or fail grade.

Deliverables

1. Technical report (Guideline length 2000 – 2500 words)
   a) Standard front sheet (Attached to this assignment. Do not include whole assignment brief)
   b) Table of contents.
   c) Evidence of testing:
      i. Test logs providing information of all the tests carried out (including any failed tests for functionality not implemented)
      ii. List of any bugs/weaknesses you have discovered along with what you did to fix them, or what you would do to fix them given enough time

   d) Design & Technical documentation. Provide information that would be useful for another developer (not an end user!) if they were given the site to build on top of. You should explain the reasoning behind design decisions and the code structure. Provide instructions and any useful information for another developer if they wanted to:
      i. Add a new page to the website
      ii. Add a new field (e.g. location) to the news article page
      iii. Add a new field (e.g. rating) to the comment form
      iv. Extend the website to allow editing comments
      v. Any other features which you have implemented e.g. password hashing or security
      vi. Document any design decisions you made which were intended to make future developments easier

   e) Evaluation
      i. What you think has been well implemented
      ii. Where improvements could be made
      iii. What you would do if you had more time
      iv. How well is the code written? How easy is it to make changes to the website in the future? Does making a change require editing multiple files? Is it easy for another developer to find code that performs each task?

   f) References (use Harvard referencing). If you have used code from a book or that you have found online you **must** indicate clearly which parts of your code they are and include references. Failure to reference code you have used may result in a capped grade or failing the assignment.

2. Source code
   a) The source code must be well documented with relevant comments. Consistent and clear indentation of the code is also important. You must submit the source code in two forms:
      i. A zip file containing the *websites* directory directories as well as the Vagrantfile. The marker should be able to extract the files and type “vagrant up” and access your website. You will lose marks if your code contains references to files using paths that are only relevant to your computer. E.g. C:\Users\Name\Documents\CSY2028\file.php all references to files should use relative names.
      ii. A commented full listing as a word document named “Appendix”

3. Video Demonstration
    a) In Addition to the report you must provide a video demo of your assignment. The demo should be 5-10 minutes (No longer than 15 minutes) and uploaded to Kaltura.

Marking Criteria

The grade for this assignment will form 50% of the overall assignment grade for the module. The Standard Front Sheet of the assignment gives an indication of how the marks are split. In general the following criteria will act as a guide to what you should expect:

	A	B	C	D	F	G
Functionality (35%)	75+ functionality marks	60-74 functionality marks	50-59 functionality marks	40-49 functionality marks	0 – 39 functionality marks or Basic requirements not met. or Administrators cannot add questions to the website without writing PHP code.	No submission or no submission of merit
Testing & Error Handling (15%)	All functionality has been tested. Tests include information on exactly what was tested: The value entered, the expected outcome and the actual outcome. Tests cover all functionality and no unidentified bugs remain. There is enough information for someone else to replicate the test exactly.	Most functionality has been tested. The value entered, the expected outcome and the actual outcome. Tests cover all functionality and no unidentified bugs remain.	Tests have been carried out on most features and documented but lack critical information on exactly what was tested e.g. the value entered and expected outcome are not present in the test logs	Some features have been tested but the testing strategy is not comprehensive and obvious bugs have not been found during the testing process.	Unspecific, ambiguous, needs some overhaul	No submission or no submission of merit
Technical Documentation (20%)	Excellent documentation with in-depth explanation of file structures, code structure and where to look for important parts of the code. Excellent explanations of what would be needed to make changes in future. Includes detailed reasoning behind file/code structure and comparisons between different approaches.	Excellent documentation with in explanations of file structures, code structure and where to look for important parts of the code. Good explanations of what would be needed to make changes in future. Includes some reasoning behind file/code structure and contains at least one comparison of different approaches which could have been taken.	Some documentation of file/code structure with a good explanation of how another developer can make changes to the website.	Minimal documentation of code. Explains file/code structure and some instructions for other developers have been included.	Does not explain technical details of supplied code and/or does not provide instructions for another developer to make changes.	No submission or no submission of merit
Evaluation (15%)	Answers all questions from the brief in detail. Explains any shortcomings of code and suggests solutions for any issues identified.	Answers all questions from the brief in detail. Explains any issues with code structure.	Answers all questions from the brief in detail.	Answers questions from brief with little detail or does not answer all questions.	Does not answer several questions or does not provide any specifics.	No submission or no submission of merit
Code Quality & Efficiency (10%)	Excellent use of available tools: Arrays, Loops, Objects and Classses. Making changes to the website does not require making the same change in multiple locations.	Some use of OOP and thought has been put into the structure of the application	Program works but functions/arrays/loops/objects would reduce complexity. Any repeated code (e.g. database connections) are placed in their own files and included when necessary.	Program works but functions/arrays/loops/objects would reduce repeated code.	Sections of the code are never used/irrelevant. Code is repeated frequently and no thought has been made into the structure of the code.	No submission or no submission of merit
Demonstration (5%)	Covers all implemented features in sufficient detail. Any known bugs are highlighted. Validation is tested (e.g. entering invalid values)	Covers all implemented features in sufficient detail.	Covers the functionality from the basic requirements in detail. Any known bugs are highlighted.	Covers the functionality from the basic requirements but needs more detail.	Video does not demonstrate all of the basic requirements.	No submission or no submission of merit