CSY2028: Web Programming

Date of Issue	**Tuesday, 23rd October 2018**	**Last Date for Submission:**	**Sunday, 13th January 2019 23:59**
		Module Tutor:	Thomas Butler
This assignment is weighted as 50% of the Module's assessment			
Assessment Feedback is provided on the rubric via NILE			

Submission Guidelines – Please read carefully

1. The University of Northampton's Policy on Plagiarism & Mitigating Circumstances will be strictly implemented.

2. This is not a group project, by submitting this assignment you are asserting that this submission is entirely your own individual work. You may discuss the assignment with other students but any code written should be your own. **Sharing your work with another student, or submitting code that was written by someone else may be deemed academic misconduct.**

3. If you have used any code that you did not write you must:
   i. Correctly reference the code in the report (use Harvard referencing)
   ii. In your report, clearly document which lines of code you used, where you used them and what they are used for.

4. You must supply **all four** items of assessment and **upload them to the correct submission points**. All work must be uploaded to turnitin
   i. Report word document (uploaded to turnitin)
   ii. Source code (zip file). The marker must be able to download and run your code. *Do not include your video in your zip file*
   iii. Source code word document (uploaded to turnitin)
   iv. Video demonstration

Please make sure you double check all submissions. It is your responsibility as a student to ensure these guidelines have been met.

Failure to follow the submission guidelines may result in a capped grade of F-.

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 working for web development agency. A local electronics shop wants you to build a website which they can use to advertise products. 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 to find information about products the shop has in stock. Each product will have a title, price, manufacturer and a longer description. The company should be able to log in to a secure administration area and add new products, entering all the information about the product, making it visible to the public.

Each product will fall into a category such as TVs, Cameras, Phones or Gaming. Categories should not be limited to these choices and new categories should be able to be added via the administration area.

One product will be marked as *Featured* and displayed in the sidebar. This will be used for special offers and discounts.

The public will then be able to browse products and filter by category, for example, seeing all Phones that the shop has in stock.

Members of the public should be able to add reviews to each product. The reviews for each product should be visible below the product description and reviews should only be visible on the page of the product they were posted to. When a user posts a review, their name, email address, review text and date the review was posted should be logged and displayed on the page.

Along with the working website, you must provide technical documentation so that other developers at your company can easily work on the website when you are not there.

Basic Requirements (Grades D- to C-)

The system must use the layout that was supplied by the designer and:

1. Have a password protected administration area that has functionality for: *(20 functionality marks)*
   a) Add products
   b) Adding categories
   c) Assigning products to categories
   d) Editing an existing product e.g. changing the title or text
   e) Deleting products from the website
   f) Editing category names
   g) Deleting categories
   h) Marking a product as “Featured” to have its title and description display in the sidebar

2. Have a publicly visible front end that allows users to: *(20 functionality marks)*
   a) Browse all the products displaying newest first
   b) Views a list of product categories in the drop-down menu in the supplied HTML layout

c) Click on one of the categories to view products in that category (products from other categories should not be visible)
d) Add a review to a product
e) See reviews added to that product by other users. Reviews should be visible on the product page, and only reviews for the selected product should be visible.

Marks will be lost for poor 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, edit PHP files, manually change the database, or type in information that is already in the system.

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 reviews. When a reviews 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 products and have it appear on the page with the title and product text *(5 functionality marks)*
- Social media buttons allowing users to easily share reviews *(5 functionality marks)*
- Allow searching for products 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 products. The system should track who posted which product and show it in the administration area *(10 functionality marks)*
- Securely store passwords with an appropriate method *(5 functionality marks)*
- Prevent users from adding reviews unless they create an account. Users should be able to register an account and log in, once logged in they should be able to review a product by entering only their review text. *(10 functionality marks)*
- Allow users to sign up to be notified via email when new products 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 review they have posted *(5 functionality marks)*
- Add a rating to reviews and allow users to sort products in a category by highest rated first (8 functionality marks)
- Implement a shopping basket so users can place an order with one or more products (and more than one of each product if required). The should appears in the administration area with the users address/shipping address and order then be marked as "Shipped". Administrators should be able to see any pending orders or orders which have already been shipped (15 functionality marks)
- Implement PayPal payments in the shopping basket. Make sure you use PayPal's developer mode (8 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.

Technical Report

In addition to the website you are required to provide technical documentation for other developers who will be working on the website.

Part (1) of the techical report is a checklist of implemented features. (See checklist.docx) that follows this format:

Checklist

You must include a checklist of functionality you have included.

Functionality	**Completed? (Yes/No/Partially)**	**Relevant files containing code for this feature**	**Any comments (e.g. usernames/passwords, known bugs/issues) or URL if different to filename**
Administrator log in	(e.g.) Yes	(e.g.) admin.php	(e.g.) Username: admin, password: admin
Add products			
Add categories			
Assign product to category			
Edit product			
Delete product			
Edit category			
Delete category			
Browse products (public)			
View categories (public)			
Filter products by category (public)			
Add review (public)			
View product reviews (admin)			

Please extend the checklist with any additional functionality you have implemented.

Along with the checklist, you should provide step by step instructions, with code examples where relevant, teaching another developer how to:

1. Add a new page to the website (e.g. an FAQs page) and link it from the navigation.
2. Add a new form field to the “Add product” form
3. Add a new form field (e.g. rating) to the “Add review” form

You should also document any security features you have implemented and design decisions you have made while developing the website (e.g. file/folder structure, how you have structured the code so it can be reused)

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 web server uses PHP 7.2. The website you build must work on this version and not use functions that have been removed or deprecated (e.g. old mysql functions. Using removed functions such as mysql_connect() will result in an F grade as the website will not work on the company's web server). *If the website works in the virtual machine, it will work on the company's web server.*

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 code from CSY2028 lecture notes, any code you submit which you did not produce yourself **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 must 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 1500 – 2000 words)
    a) Table of contents.
    b) Checklist of completed features
    c) Evidence of testing:
        i. Test logs providing information of all the tests carried out (including any failed tests for functionality not implemented).
        ii. How you fixed any bugs/weaknesses you discovered, or what you would do to fix them given enough time

    d) 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 along with instructions for another developer to complete the tasks required in the *Technical Report* section of the assignment brief.

e) 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. You must demonstrate every feature you have implemented and do not need to discuss code. 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 rubric below 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 (50%)	75+ functionality marks	60-74 functionality marks	50-59 functionality marks	40-49 functionality marks	0 – 39 functionality marks or Basic requirements not met. Marks will be lost if the website is unnecessarily difficult to use e.g.: • Users having to remember numerical IDS • Users having to type in information that is already in the database • Adding/modifying data on the website requires manually amending the database or editing PHP files	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 (15%)	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
Code Quality & Efficiency (15%)	Everything for (B) plus 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.	Consitent and clear file/folder structure with a focus on security: Relevant files outside the public directory and good password hashing	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.	Some attempt has been made to reduce repeated code.	Marks are lost if: Sections of the code are never used/irrelevant. Code is repeated frequently and no thought has been made into the structure of the code. Chunks of code are repeated on every page or very inefficient, e.g. changing a password requires editing every page. Links are hardcoded and would not work if run from a different URL Adding products/categories to the website requires editing PHP files or manually changing the database	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