Swinburne University of Technology
School of Science, Computing and Engineering Technologies

COS10026 Computing Technology Inquiry Project

Assignment Part 2, Semester 1, 2023
Server-Side Programming

Important Information

Due Date	**23:59 (VN Time) on Monday, 10 April 2023 in Week 13** (Late submission penalty: 10% of total available marks per day)
Submission Method	**Canvas + Mercury**
Contribution to Final Assessment	50%

Purpose of the Assignment

To gain practical skills and knowledge in coding and using PHP server-side embedded scripting to extend the functionality of the website you developed in Part 1, by creating dynamic webpage content, accessing a separate MySQL database server, and using PHP to connect to this MySQL server.

In particular, you will:

- Use PHP to include common webpage code (eg. menu, header, footer code)
- Understand the ways that 'state' can be maintained between web pages
- Use PHP to validate data sent in the "payment" HTML form to a new "process_order.php" page and if any errors, provide user feedback through a new "fix_order.php" form.
- Use PHP to create and store the order data in a server-side MySQL database table "orders", and provide feedback through a new "receipt.php" page.
- Use PHP to query and update the status of the "orders", through a new "manager.php" page.
- Understand how using a "settings.php" script, and relative links, can enable the website to be ported from a development environment to a testing environment.

As in Parts 1, there is an opportunity to enhance your website beyond the specific requirements.

Specified Requirements

1. Use PHP to reuse common elements in your website

PHP provides us with techniques to modularise and reuse our web application code. Rewrite your web pages so that the common static HTML elements such as menu, header and footer are written in common text files that are then "included" back into your web pages. Name the include file(s) with an `.inc` extension, replace the sections of HTML in your main pages with 'include' statements, and rename your main pages with a `.php` extension, so the php includes will be included.

2. Create an 'orders' database table

Create an 'orders' table in your MySQL database.

This table will record the data sent from the "payment.php" form (including the data gathered from "enquire.php"). This data contains information on the customer, product and payment details as specified in the previous assignments. The format of the database fields should match appropriate validation rules defined in assignments Part 1 and 2. If no rule exists for a particular field, choose an appropriate format.

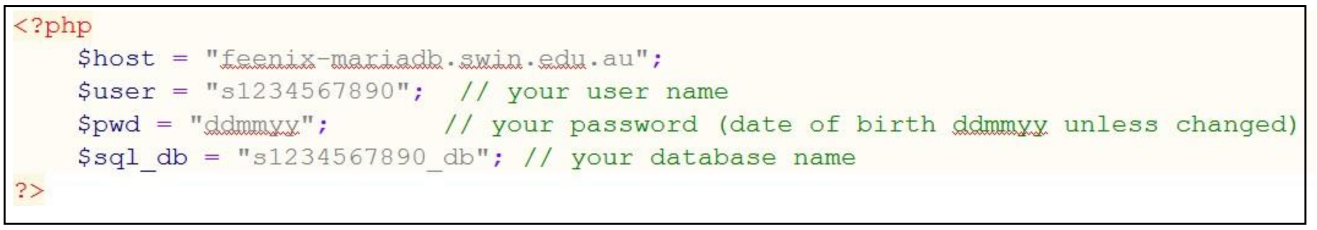
In addition to the fields that record information from the "payment" form, add the following fields with appropriate data-types to the table:

- An auto-generated primary key field called 'order_id'.
- The total cost of the order 'order_cost'
- Date / time of order (generated by the system) 'order_time'.
- A field called 'order_status'.
  An 'order status' can have one of three values: PENDING | FULFILLED | PAID | ARCHIVED.
  When an order is created its status is always set to PENDING.

Later, your "`process_order.php`" page should be able to create this table, if it does not already exist, when the first order is made.

3. Create a file to store your database connection variables "`settings.php`"

To enable your website to be easily ported to our "testing" environment (and in a workplace, ultimately to a production website platform), use a PHP include file, "settings.php" that contains the ***connection variables as shown below***, and use this in your PHP to connect to your MySQL database on the feenix-mariadb database server.



```
<?php
    $host = "feenix-mariadb.swin.edu.au";
    $user = "s1234567890";   // your user name
    $pwd = "ddmmyy";        // your password (date of birth ddmmyy unless changed)
    $sql_db = "s1234567890_db"; // your database name
?>
```

4. Payment "`payment.php`"

4.1 Create the payment form with price, quantity and credit card information

Extend the enquire form to enable a user to purchase a product/service that they have selected. You can **either** 1) rename the enquire.php to payment.php then add the inputs listed below; **or** 2) add a new payment.php, make the enquire page submit to payment.php. Then include the inputs listed below in enquire/payment form accordingly.

You will need to:

- Add some **price details** for various items in the product/service range, and add extra options where applicable.
- Add one or more fields so a user can select the **quantity** of the product/service to be purchased. *For a product, the quantity might represent the number of items. For a service, the quantity might represent the dates of hire, or the number of days.*
- Add more form inputs to include **credit card** payment details:
  a. Credit card type (e.g. Visa, Mastercard, American Express) – *no valid default selection*
  b. Name on credit card
  c. Credit card number
  d. Credit card expiry date (mm-yy)
  e. Card verification value (CVV)
- Have a 'submit' button with caption 'Check Out'.

Set the "payment.php" form action to "process_order.php". When a customer decides to proceed with the order, a **Check Out** button should submit the form, and all the customer and product information to the server-side script. (So you check that all name-value data has been correctly submitted.)

4.2 Disable HTML validation

While client-side validation using HTML5 was used in previous assignments, in order to preserve the integrity of the server data, server-side data format checking should be implemented.
In this assignment HTML5 form validation will be disabled.

So we can test that server-side validation works correctly:

Add the **novalidate="novalidate"** (or simply **novalidate)** attribute into your forms, to disable client-side HTML5.

5. Processing Orders "`process_order.php`"

Ensure the "payment.php" form action to "process_order.php". Having disabled HTML5 form data validation, all form data format checking will now be implemented server-side, using PHP, after the data has been submitted by "payment.php":

1. All values received by "process_order.php" should be sanitized to remove leading and trailing spaces, backslashes and HTML control characters.
2. Before an order is written to the **`orders`** table the data format rules need to be checked. These rules include rules specified in Part 1 (for customer details), rules for credit card (shown below) and other rules you think appropriate.
    - Credit card type must be one of Visa, Mastercard, or American Express
    - Credit card number: exactly 15 or 16 digits. Credit card numbers must be checked against the selected card type according to the following rules:
        - Visa cards have 16 digits and start with a **4**
        - MasterCard have 16 digits and start with digits **51** *through to* **55**
        - American Express has 15 digits and starts with **34** *or* **37.**

If the input data does not meet format requirements, errors should be returned to "**`fix_order.php`**" a form version of the 'payment' page, and display all form control fields filled with data entered in enquire page and payment page, and with errors marked or highlighted. Do not fill the Credit Card details, these will need to be re-entered. The "fix_order.php" form should submit back to "process_order.php", using method post. *Hint: error msg back to fix_order could be string or an array.*

If the input data is correctly validated by "process_orders.php":

1. Calculate the total cost of the order (do not rely on the client to send this information).
2. Store the order in the orders table using a mysqli query.
3. Return an order receipt webpage "**`receipt.php`**" to the user. This page should include all the information stored in the record including the **`order_id`** and **`order_status`**.

The "process_order.php" page should not produce a html page. It should only process data and pass data to other webpages. (During development you might want to have this script echo back data.)

The "process_order.php" page should include a check that if the database table 'orders' does not exist then create it.

The "process_order.php", "fix_order.php" and "receipt.php" pages should not be able to be accessed directly by url through a browser.
Hint: check what data has been set and redirect.

6. Managers Order report and Order Update Page “`manager.php`”

For convenience add an extra menu item to access this new Manager page.

This web page allows the Manager to make queries about orders, display the result in an HTML table and update the status of an order.

For each query clearly display: order number, order date, full details of the **product** including the **cost,** only the customer’s **first** and **last** names, order status. No credit card details should be displayed.

To make the display presentable and easily readable, you might need to concatenate some fields.

The web page should give the manager the option to display:

- All orders
- Orders for a customer based on their name
- Orders for a particular product
- Orders that are pending
- Orders sorted by total cost

The Manager should be able to ‘update’ the status of an order from a link or button next to the order in the table, changing the status from (pending | fulfilled | paid | archived).

The Manager can also ‘cancel’ (ie. ‘delete’) an order via this page. Only pending orders can be cancelled.

CSS

All pages should be styled appropriately using CSS as in Part 1, and should be valid CSS3.

Enhancements

Please complete all the ***Specified Requirements*** before attempting any enhancements.

See the Marking Guide below.

As with Part 1 you have an opportunity to extend your learning by adding extensions/ enhancements to the main pages of your Web site, using techniques not covered in the tutorials.

Briefly **list** and **describe** each enhancement implemented on a page called **`enhancements3.php`**:

- What it does and how it goes beyond the specified requirements.
- What does a programmer have to do to implement the feature.
  (A reminder to your future self of what you have done.)
- Reference any third party sources used to create the extension/enhancement

Any enhancements that are not listed on the PHP enhancements page will not be assessed.

In this assignment we will consider PHP and MySQL enhancements.
You are encouraged to be creative in thinking up possible enhancements.

Examples of PHP / MySQL enhancements:

- Create Manager security, with a “Manager registration” page with server side validation requiring a unique username and a password rule, and store this information in a table. Create a “Manager Log-in” page to use the stored data, and control access to the manager web pages. Ensure the manager web page cannot be entered directly using a URL.
  Create a “Manager Log-out” page. Provide a ‘log-out’ link on the manager page if ‘logged in’.
- Provide a number of more advanced Manager reports based on compound queries.

For example

- o the most popular product ordered
- o fulfilled orders purchased between two dates the vendor enters
- o the average number of orders per day

- On the table on the Manager page, provide the ability to select a column heading, and re-sort the table in the order of that field. If selected again, reverse the order.
- Store customers' details in a separate 'customers' data table and create a primary-foreign key link between the 'customers' and 'orders' tables.
- Store the product details and options in a separate 'products' data table, and dynamically fill the product page with that data.

Note: If you are going to implement functionalities for the manager, please use "admin" as the username and "password" as the password. Your tutor will need these credentials to login and validate these functionalities.

Up to two enhancements will be assessed, i.e., up to 20 marks will be awarded for enhancements.

Web Site Folder Structure and Deployment Requirements

Your website folder structure should follow a similar structure as in Part 1 and 2.
All files should be under a folder `/assign2`.

```
Assign2/                 You must have this folder – case sensitive!
  index.php
  product.php
  payment.php
  about.php
  enhancements.php
  enhancements2.php
  header.inc      ⎫
  menu.inc        ⎬ You could put these in an
  footer.inc      ⎭ 'includes' folder
  settings.php
  ...other php function or include pages
  process_order.php
  fix_order.php
  receipt.php
  manager.php
  ...other php pages
  images/                 Folder for images for your page content
  styles/
     style.css            other css files
  styles/images/          Folder for images referred to by your CSS files e.g., background
```

Notes:

- PHP/HTML files should only be in the base "assign2/" folder – not anywhere else.
- All links to your files (includes, CSS or images) should be ***relative.*** ***Do not use absolute links***, as these links will be broken when files are transferred for marking. No marks will be allocated if links are broken.

Individual Report (Individual Task)

The individual report is an individual task. Your report must be professionally written (**600-1500 words**). Table 1 presents the suggested structure for the report and some sample content for each section of the report.

Table 1. Components of Report and Requirements

Component	**Content**
Cover Page	• Report title • Your name • Student ID
Introduction	• Objective of the report • Outline of the report's structure
The Website	• Website introduction • Main functionalities of the website, including screenshots • Technical details on the web site development • Highlight the key / innovative features of the website
Your Contribution	• List and discuss your main contributions
Reflection and Discussion	• Reflections on your experience regarding professional purpose, teamwork and effective communication in this project • Discussions on **one** of the following web development related issues: privacy, security, environmental sustainability, commercial or social issues. For example, in this project or in a similar future project, how did/will you consider the privacy, security, environmental sustainability, commercial or social issues?
Conclusion	• Summary of the report
References	• (Optional) List of reference materials if used
Appendix	• (Optional) Information that supports but is not essential to the report

Short Video (Team Task)

Create a short video to introduce and demonstrate your web application.

- Upload your video to youtube
- Create a hyper link in the index.html page of your website, link it to your youtube video
- Every team member must present in the demonstration video for a similar amount of time
- The total length of the video should be between 4 to 5 minutes.

Assignment Submission (Mercury + Canvas)

Deliverables

The marks are allocated 50% for team task and 50% for individual task in this assignment.

- The web application (team task)
- The short video (team task)
- Project report (individual task)

Assignment Submission (Canvas + Mercury)

Your website should be uploaded to Mercury on or before your deadline.

An electronic copy of your assignment should be submitted through Canvas on or before your deadline.

- Make sure all your website files are in the correct folders and compress your root folder with all your sub-folders with HTML, CSS, PHP and images into a zip file named “assign2.zip”. Submit this to Canvas. When the zip file is decompressed, the entire website should be able to be run from index.html without needing to move any files.
- You don’t need to submit the demonstration Youtube video. You only need to include a hyperlink in the index.html page pointing to your Youtube video.
- Only the team leader needs to submit the assignment zip file.
- Every student needs to submit their individual project report to Canvas.
- Every student needs to submit their peer evaluation form to Canvas.
- You can submit more than once through Canvas. Your last submission will be marked.
- Note that all deliverables must be submitted electronically.

Make sure you complete your Canvas submission process.

Website Requirements Checklist

Requirements	
PHP common static includes: menu, footer, header, static code blocks included ☐ ☐ ☐	
settings.php created, included and used ☐	
Orders Table (view in phpMyAdmin) - scheme can store all the necessary data ☐ - primary key order_id auto generated ☐	
payment.php - client-side HTML5 form validation disabled ☐ ☐	
process_order.php - unable to access directly through URL ☐ - all data read from payment.php ☐ - text data inputs sanitised ☐ **Data errors validated**, displayed back in **fix_order.php** - fnames ☐ lnames ☐ email ☐ str_addr ☐ state ☐ pcode ☐ phone ☐ - preferred contact ☐ qty ☐ product ☐ options ☐ - total cost calculated ☐ - CC type ☐ CC name (alpha) CC number (15-16 digits) ☐ CC expiry date ☐ - CVV ☐ CC formats checked ☐ post code checked against state ☐	
fix_order.php - all data sent from **process_order.php** to fix_order.php - all data displayed in form ☐ *(CC details not sent shown blank)* - errors displayed in page ☐ - form submits all data back to **process_order.php** ☐	
receipt.php - all data sent from **process_order.php** to receipt ☐ - all order data displayed in receipt page ☐ *(Secure CC details shown ****)* - record added to orders table by **process_order.php** *(view in phpMyAdmin)* ☐	
manage.php - linked from menu ☐ - form to make queries, with required options ☐ - results in HTML table order fields (#,date,product,cost,name,status) ☐ Manager can query: - all orders ☐ - orders for a customer based on name ☐ - orders for a particular product ☐ - orders that are pending ☐ - orders sorted by total cost ☐ - can 'update' status of an order (pending | fulfilled | paid | archived) ☐ - can 'cancel' a pending order ☐	
Create database table automatically (in process_order.php) - creates 'orders' table if it doesn't exist on first order ☐ *(can be tested by dropping table in phpMyAdmin)*	

Other Deductions *based on documentation, code and file inspection*

Requirement	
HTML - Deprecated elements/attributes have been used [-5] - Inappropriate use of HTML semantics [-5] (e.g., use of <div> when <section> <article> should be used) - HTML usability does not follow standards (e.g., alt on images, label in forms, tables) [-5] - HTML Image height, width attributes missing or incorrect [-5] - HTML has embedded Style markup. CSS is not fully separated from HTML [-5] - Code comments inadequate to inform later code understanding/maintenance [-5] - HTML webpages not fully valid [-5 each webpage]	
PHP - Inappropriate header comments - do not match in-house standard. [-5] - Code comments inadequate to inform later code understanding/maintenance [-5] - Uses only mysqli commands [-5]	
Web site - Directory and file structure not as specified [-10] - Third party content inadequately acknowledged [-10] - Failure to acknowledge third party code or content at all is plagiarism and may result in zero marks for this assessment, or other penalties in accord with Swinburne policy**. [-100]**	
Other Deductions **-** other deductions not listed above [-100]	

Website Marking Rubric

PHP include and table structure (5 marks)	5 marks	4 marks	3 marks	2 marks	0 - 1 marks
	The requirements are implemented with high quality and follow the in-house standard.	Most of the requirements are implemented and follow the in-house standard.	Some requirements are not implemented correctly or not follow the in-house standard.	Limited requirements are implemented.	No or very limited requirements are implemented.
Process_order, fix_order, receipt page (20 marks)	17 - 20 marks	13 - 16 marks	9 - 12 marks	5 - 8 marks	0 - 4 marks
	The requirements are implemented with high quality and follow the in-house standard.	Most of the requirements are implemented and follow the in-house standard.	Some requirements are not implemented correctly or not follow the in-house standard.	Limited requirements are implemented.	No or very limited requirements are implemented.
Manage page (10 marks)	9 - 10 marks	7 - 8 marks	5 - 6 marks	3 - 4 marks	0 - 2 marks
	The requirements are implemented with high quality and follow the in-house standard.	Most of the requirements are implemented and follow the in-house standard.	Some requirements are not implemented correctly or not follow the in-house standard.	Limited requirements are implemented.	No or very limited requirements are implemented.
Enhancement (10 marks)	9 - 10 marks	7 - 8 marks	5 - 6 marks	3 - 4 marks	0 - 2 marks
	Excellent enhancement. It enhances the functionality of the website. It is technically advanced.	Very good enhancement. It enhances the functionality of the website.	Implemented some simple enhancements and/or they don't meet some of the requirements listed in the enhancement section.	Limited work and/or they don't meet some of the requirements listed in the enhancement section.	No or very limited work.
Video Demonstration (5 marks)	5 marks	4 marks	3 marks	2 marks	0 - 1 marks
	Excellent knowledge of the project is demonstrated. Presentation is highly creative and accomplished.	Very good knowledge of the project is demonstrated Presentation is skilful and creative.	Adequate knowledge of the project is demonstrated. Presentation is clear and adequate.	Not enough information is presented. More practice is needed.	No or very limited information is presented. More practice is needed.

Individual Report Marking Rubric

	9 - 10 marks	7 - 8 marks	5 - 6 marks	3 - 4 marks	0 - 2 marks
Overall Presentation of Report (10 marks)	Quality of report is truly professional, clear and easy to follow, has good structure, cohesive and well thought with the reader(s) in mind.	Quality of report is professional, clear and easy to follow and has good structure.	Quality of report is clear and has good structure.	Quality of report is somewhat clear, but still difficult to follow and/or it contains some spelling or grammatical errors.	Quality of report is unprofessional, difficult to follow and/or it contains numerous spelling or grammatical errors.
	13 - 15 marks	10 - 12 marks	7 - 9 marks	4 - 6 marks	0 - 3 marks
The Website (15 marks)	Very detailed information allows reader to understand the functionality and technical details of this website.	Detailed information allows reader to understand the functionality and technical details of this website.	Adequate information allows reader to understand the functionality and technical details of this website.	Limited or insufficient information on the functionality and technical details of this website.	Minimal information does NOT allow reader to understand the functionality and technical details.
	13 - 15 marks	10 - 12 marks	7 - 9 marks	4 - 6 marks	0 - 3 marks
Your contribution (15 marks)	Your contributions are clearly listed and well discussed.	Your contributions are listed and explained.	Adequate information about your contribution to the project.	Limited or insufficient information on your contribution to the project.	Minimal information does NOT allow reader to understand your contribution to the project.
	9 - 10 marks	7 - 8 marks	5 - 6 marks	3 - 4 marks	0 - 2 marks
Reflection and Comments (10 marks)	- Reflection demonstrates a deep commitment to the reflection process. - Convincing, critical, well-articulated and rigorous discussions on the web development related issues.	- Reflection demonstrates a commitment to the reflection process. - Sound discussions on the web development related issues.	- Reflection demonstrates some commitment to the reflection process. - Satisfactory discussions on the web development related issues.	- Reflection demonstrates an interest to the reflection process. - Limited or insufficient discussions on the web development related issues.	- Reflection demonstrates little interest to the reflection process. - No or very limited discussions on the web development related issues.