Lab 6: Working with Sessions [HTML, CSS, PHP]

Learning Outcomes:

- Master the use of Procedural PHP for creating dynamic web pages.
- Consider the use of PHP Superglobals for processing form data.
- Apply flow control to functions (i.e., Conditionals) and loops in PHP were appropriate.
- Understand how to use Sessions for passing data through pages on a site.
- Understand how PHP is interpreted on the server-side and how to deal with syntax issues.

Instructions:

- For this lab, **you will be extending on the work you did for Lab 5**, therefore, you will be using the files you created when completing Lab 5. You may make any changes you see fit to your files, whether in its functionality and/or look-and feel.

- For Lab 6, you will be focusing on implementing the use of Sessions for implementing a online Shopping Cart. In other words, you will be expected to use Sessions for collecting and handling shopping cart data. For this purpose, you are tasked with:

  (a) **Implementing the use of Sessions in your lab**

  - When a user visits your 'Catalog' or 'Products' page, a session should be created, regardless of whether the user has "logged in" or not.

  (b) **Adding a 'price' to your catalog items**

  - Extend the multidimensional array code you created in Lab 2 so that it includes a 'price' for each of the items in your Catalog.

  (c) **Adding a 'Buy' option to your catalog items**

  - Add a 'Buy' button to each of your Catalog items, similar to the example shown on Figure 1.
  - When user clicks on the 'Buy' button for a given item, this item should be added to that user's Shopping Cart session.

- User's should not be expected to go through a 'log in' process to add items to their Shopping Cart.

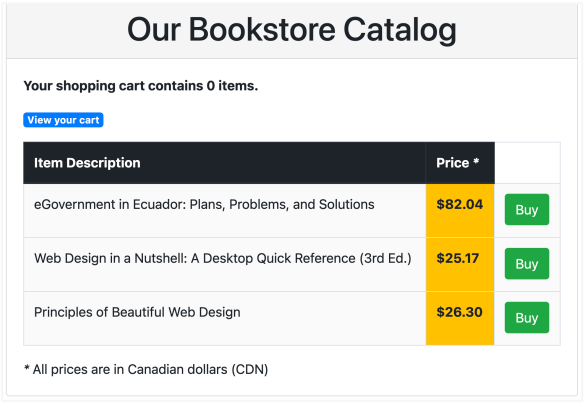

Figure 1. Example of a Catalog page.

(d)**Keeping count of how many items are in a Shopping Cart**

- As your users add items to their Shopping Cart, they should be able to see how many items they have in their shopping cart (e.g., 'You have 3 items in your cart' or the number 3 next to a Shopping Cart icon), similar to the example shown in Figure 1.

- Users should be able to click on a 'Checkout' or 'Shopping Cart' link that takes the user to the 'Checkout' page, similar to the example shown in Figure 1.

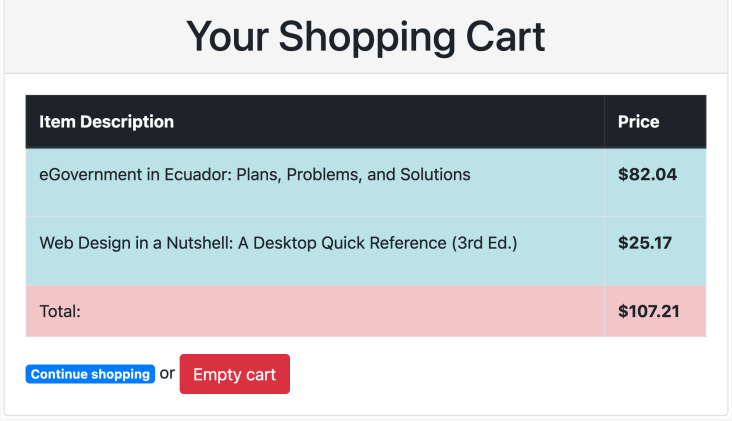

Figure 2. Example of a Shopping Cart page.

(e) **Creating a 'Checkout' page**

- Create a 'Checkout' or 'Shopping Cart' page which lists the items that were 'added to the cart', along with a sub-total or price for each item and a total for all items in the cart. See Figure 2 for an example of a Shopping Cart page.

- Your 'Checkout' or 'Shopping Cart' page should provide the option to the user to empty their shopping cart or to continue shopping.

(f) **Modify your page structure to allow for the use of template functions**

- Explore the use of templating functions to better organize your code (e.g., header, footer, login, catalog, cart, checkout, etc.) and improve the maintainability of your code.

(g) **BONUS: Dealing with individual shopping cart items**

- In your 'Shopping Cart' page, you may provide users with a 'quantity' field to illustrate the number of copies of a single item, instead of displaying multiple copies of the same item.
- Users should have the option to edit the number of copies they would like to purchase for a given item (e.g., two copies of the same book instead of one or vice-versa).
- Users should be able to delete a single item from their shopping cart without emptying their entire cart.

- For this lab, you **WILL NOT** need to connect your catalog or shopping cart pages to a database. Instead, you will need to store your catalog items in an array, which you can then use when populating the information in your Catalog page.

  Note: Though you created this multidimensional array in Lab 2, you are free to modify its structure or extend your code in order to meet your vision for this lab. As usual, you have complete creative freedom in regard to colour palettes, imagery, content and typography for your overall design. However, you should strive to try to create an aesthetically pleasing site. You **may use any Front-End CSS Framework** in completing this lab.

- You are free to explore any techniques or approaches we have covered in class so far, and assess their suitability for the approach you would like to take in this lab.

- As you may have noticed, in this lab, you will be expected to create the HTML, CSS, PHP, and parsed-HTML (i.e., HTML generated by your PHP script) necessary for completing this registration form. Therefore, it is important that you ensure your HTML and parsed-HTMl, as well as your CSS, are validated by the W3.

  Note: You have complete creative freedom in regard to colour palettes, imagery, content and typography for your overall design. However, you should strive to try to create an aesthetically pleasing site. You **may use any Front-End CSS Framework** in completing this lab.

Submission Guidelines

- For this lab, you will need to **submit your work through Timberlea, GitLab AND Brightspace** as follows:

Submitting your Work through Timberlea

- As part of this lab, you will need to create a new directory inside your **'csci2170'** directory on Timberlea, named **'lab6'**. ***See Lab 1 instructions*** on how to log onto Timberlea using an FileZilla, and create directories.

  Note: You will need to ensure your new directory has the correct **folder permissions** applied to it (i.e., **755**), and that each of your files also has the correct **file permissions** applied (i.e., **644**).

- Once you have completed your lab, upload your work into your **'lab6'** directory on Timberlea.

  Note: You will need to ensure your submission includes all required files needed for your Lab 6 (i.e., image files, stylesheets, folders).

- Ensure you have set the **proper file and folder permissions** on your **'lab6' directory** and your lab's **individual files**.

  Note: In order for your files to be accessible through a browser for testing and grading, you must ensure you are using the correct file permission settings on your files and folders. On a shared server, such as Bluenose, it is recommended to **use '755' (i.e., rwxr-xr-x) on folders**, and **'644' (i.e., rw-r--r--) on individual files**. You can set your file permissions easily through an FTP client by right clicking on the file or folder you want to set specific permission settings. Depending on your FTP client, you will need to click on **'Get Info'** or **'File Permissions'**. Once on the file permissions window, you can simply enter the numeric value described above.

- Visit **https://web.cs.dal.ca/~yourcsusername/csci2170/lab6/** on any browser and ensure you can view your work.

  Note: Failure to submit your work through Timberlea will result in a grade of **ZERO (0)**. Failure to ensure your work is remotely accessible through a web browser, using the specified URL will result in a grade of **ZERO (0)**.

- Validate your HTML (**https://validator.w3.org/**) and CSS (**https://jigsaw.w3.org/css-validator/**)

  Note: Though you are using PHP, you PHP code should still parse properly structure and valid HTML code. Correct your code as necessary based on the results provided by the HTML Validator and ensure

your code does not have **errors**. You are encouraged to correct **warnings** where possible, but you will not loose marks for any warnings still present in your code after submission.

- Test your lab to ensure cross-browser compatibility. Though minor differences between browsers is expected and acceptable, you do want to make sure no exaggerated differences occur (e.g., a table is visible on one browser and not visible on another).

Submitting your Work through FCS GitLab

- Within your **'labs'** directory, in your Git Lab account's **CSCI 2170 Project**, create a directory for your current lab, in this case you'll be naming your directory **'lab6'**.

- Within your **'lab6'** directory, upload ALL the files you created in completing Lab 6, i.e., your **'lab6'** directory must include all the files you used in completing this lab.

- **Ensure** your **'lab6'** directory includes a README.txt or README.md file, follow the guidelines specified in the README template provided.

- If you have not already done so, add the course Instructor and Markers as **maintainers** to your **CSCI 2170 Project.** ***See Lab 1 Handout instructions*** for information on Instructor and Markers' usernames.

Submitting your Work through Brightspace

- Download the **README template** available on Brightspace. ***See Resources section*** on left-hand side menu on Brightspace. There are TWO versions of this template, you may use whichever you feel more comfortable with.

- Edit the README template to include any citations for your code and/or images used for this Lab.

  Note: If the work you are submitting as part of your Lab is work done by you without the use of any external sources, then please specify so within your README file. If you have used external resources, then make sure you note them in your README file as illustrated on the template. The README templates were created to fit a variety of scenarios, feel free to edit as you see fit to meet your needs.

- Depending on the version of the template you chose, rename your README file as:

  L#_LastName_FirstName_README.md OR L#_LastName_FirstName_README.txt

  Note: Ensure your README file includes the URL to your Lab for remote access.

- Submit ONLY your **L#_LastName_FirstName_README.txt** through the corresponding Lab assignment dropbox on Brightspace.

Marking Rubric:

The following grading criteria will be used for marking your lab:

Dimensions	Does Not Meet Expectations	Somewhat Meets Expectations	Meets Expectations	Exceeds Expectations
Catalog Items Array (0.5%)	Lab does not include an items array( ) and/or array does not include a price for each item. **(0 points)**			Lab includes an items array( ) that includes the price of each item, and can be used to populate the items into the Catalog Page **(0.5 points)**
Buy button (0.5%)	Catalog page does not include a 'Buy' button for each of the items listed. **(0 points)**			Student's catalog page includes a 'Buy' button for ALL the items listed. **(0.5 points)**
Shopping Cart Item Count (1%)	Student does not provide the user with a count of the items currently in the Shopping Cart. **(0 points)**	Student provide the user with an inaccurate count of the items currently in the Shopping Cart. **(0.5 points)**		Student provides the user with an accurate count of the items currently in the Shopping Cart. **(1 points)**
Shopping Cart Session (4%)	Lab does not include the use of a Session in order to keep track of the items added to a shopping cart. **(0 points)**	Lab uses a Session in order to keep track of the items, thought its implementation is inconsistent (i.e., Session does not always start). **(1 points)**	Lab uses a Session in order to keep track of the items added to a shopping cart, but the information gathered is incomplete (i.e., Product IDs missing). **(3 points)**	Lab uses a Session in order to keep track of the items added to a shopping cart, **(4 points)**
Shopping Cart Page (3%)	Shopping Cart page does display an accurate list of all items in the cart, nor the total amount due to be paid by the user. **(0 points)**	Shopping Cart page includes a list of all items in the cart, but does not include the total amount due. **(1 points)**	Shopping Cart page includes a list of all items in the cart, but the total amount due is inaccurate or in the wrong format (e.g., no decimal points used). **(2 points)**	Shopping Cart page successfully includes a list of all items in the cart, keeping track of the total amount due to be paid by the user. **(3 points)**
Empty Cart Option (1%)	Student's lab does not give the option to the user to empty their shopping cart. **(0 points)**			Student's lab provides the option to empty the shopping cart. **(1 points)**
[BONUS] Edit Quantity of Items in Shopping Cart (4%)	Shopping Cart does not include an option to edit the quantity of repeated items **(0 points)**	Shopping Cart includes a field or option to edit the quantity of repeated items, but this field does not properly update the shopping cart (e.g., update the cart's total) **(1 - 2 points)**	Shopping Cart successfully includes a field or option to properly edit the quantity of repeated items BUT does allow the user to delete a single item. **(3 points)**	Shopping Cart successfully includes a field or option to properly edit the quantity of repeated items AND delete a single item. **(4 points)**
Templating Functions	Student does not make use of templating functions to better structure their code. **(-2 points)**			Student makes use of templating functions to better structure their code. **(0 points)**
CrossBrowser Compatibility	Student's lab is not cross-browser compatible, noticeable and distracting differences are visible. **(-2 points)**			Student's lab is cross-browser compatible, any visible differences are subtle and/or not distracting. **(0 points)**

Validation	Student's lab does not validate using the W3C HTML or CSS Validator. **(-5 points)**			Student's lab is properly validated by the W3C HTML and CSS Validator. **(0 points)**
Git Lab Submission	Student did not submit their work on Git Lab and/or did not give 'maintainer' access to course Instructor/ Markers. **(-5 points)**			Student submitted their work on Git Lab and granted 'maintainer' access to course Instructor/Markers. **(0 points)**
Remote Access (Timberlea)	Student's index.php file is not remotely accessible using expected URL. **(-10 points)**			Student's index.php file is remotely accessible using the expected URL. **(0 points)**
Readme File Submission (Brightspace)	README file not provided or contents left blank/ incorrect or student submitted unedited template. **(-5 points)**			Readme file has been provided with proper citations. **(0 points)**