Concepts Assignment 5

CS441

*This is exercises based on the textbook ones for Chapters 5 and 6. Note, there are additional steps or requirements that might not be in the book, or steps removed from the book, so be sure to follow this document vs directly following the textbook.*

*Note, all required student files are located in the Week 1 Module of Canvas. Look for the “Student Download files” and add to your htdocs folder. Remember that the weekly code will be copied back and forth from your local Github repository and the htdocs folder.*

*Before starting, make sure to copy the directories “book\_apps” and “ex\_starts” to your local computer in the following directory: C:\*xampp\htdocs\

**Instructions:**

# Exercise 5-1 Enhance the Guitar Shop Application

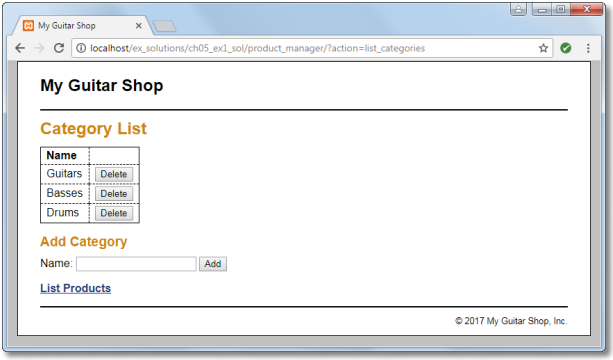
This exercise has you enhance the Guitar Shop application. That will give you a chance to use some of the skills that were presented in this chapter.

**Test the Guitar Shop application**

1. Start the Chrome browser and run the application in the ex\_starts/ch05\_ex1 directory. This should display a menu that lets you navigate to the Product Manager application or the Product Catalog application.
2. Use the Product Manager application to add a new product to the database with Guitars as the category, test1 as the code, Test Product 2211 as the name, and 550.00 as the list price.
3. Go to the Product Catalog application. To do that, you can use the Back button or run the application again. Then, view the product that you just added. Note that it doesn’t display an image for the product. To fix that, go to the images directory and change the name of the test.png file to test1.png.
4. Go to the Product Manager application again. Then, click on the List Categories link at the bottom of the page. Note that this link doesn’t display a page, even though it is coded correctly. You’ll fix this later when you enhance the index.php page for this application. Now, click the Back button.

**Enhance the Product Manager application**

Now, you’ll add a page to the Product Manager application that lets you add or delete categories. This is similar to what you did in exercise 4-1, but using the MVC pattern. The new page should look like this:



1. Open the category\_list.php file in the product\_manager directory. It contains some of the headings that you’ll need for this page, and a link back to the Product List page.
2. Open the index.php file in the product\_manager directory and add an action that displays the Category List page. Then, test this to make sure it works correctly. At this point, the page should only display some of the headings.
3. Open the model/category\_db.php file and add two functions to it that add and delete categories. (Changed order from text to do functions first)
4. In the category\_list.php file, write the code that creates the category table shown above with all of the category names in the first column and Delete buttons in the second column. Then, test this to make sure the page displays the table correctly.
5. In the category\_list.php file, write the code that lets the user add a category to the database. This code should consist of a form that accepts the name for a new category followed by a Submit button that displays “Add”.
6. Open the model/category\_db.php file and add two functions to it that add and delete categories.
7. Open the index.php file in the product\_manager directory and add two new actions to it. The first action should add a category to the database. And the second action should delete a category from the database.

*Tip: To return to the Category List page after adding or deleting a category, you can pass an action to the controller with a statement like this:*

*header('Location: .?action=list\_categories');*

1. Test the application by adding two categories. Then, navigate to the Product List page and note that the list of categories includes the new categories. Next, navigate to the Add Product page and note that the drop-down list includes the new categories.
2. Test the application by deleting the categories that you just added. However, don’t delete any of the existing categories because that will lead to products without categories. If necessary, though, you can restore the database by running the create\_db.sql script again as described in the appendixes.
3. If the formatting of your page isn’t exactly like the one above, don’t worry about that. The focus here is on web programming, not HTML and CSS.

**Optional Extra Credit: Refactor the Product Catalog application**

1. Open the product\_list.php and product\_view.php files in the product\_catalog directory. Note that these files use the same code in the <nav> tag to display the list of navigation links for each category.
2. Create a file named categories\_nav.php in the view directory, and copy all the code for the <nav> tag from the product\_list.php file to the category\_nav.php file. Then, replace the code for the <nav> tag in the product\_list.php and product\_view.php files with the appropriate include statements.

* *Make sure to include this file in your submission!*

1. Test these changes to make sure your refactoring works.

***Note, Exercise 6-1 continued on the next page!***

# Exercise 6-1 Trace with Echo Statements

*In this exercise, you’ll use echo statements to trace the execution of the Future Value application.*

1. Open the display\_results.php file for the Future Value application that’s stored in the ex\_starts\ch06\_ex1 directory.
2. Add echo statements like those in figure 6-3 to trace the execution of the code. Then, run the application to see how the echo statements work. Include a screenshot below and your modified code in the submission.

***Commit this document and all edited code to your Week 4 Github repository by the deadline. Any late work must be communicated.***