“Note: My group and I did not use Java, C++ or any language that uses classes and methods. We used HTML/CSS, PHP, and SQL. Instead of the original assignment, we emailed you and you gave us permission to each test out a page on our site and make sure that everything works properly and to report our results for that page. This includes testing out links, forms, functionalities, etc.”

For my portion of the unit test, I will be testing various cart functionalities of the Flip Book website. My unit testing will be testing the following 6 “methods:

1. **“Add to cart” button leads to the proper PHP page**

A screenshot of a computer

Description automatically generated

This button is the submit for a post form that contains a hidden value of the items bookid. (form code below)

This is the form that sends to addToCart.php when the “Add to Cart” button is selected

A black sign with white text

Description automatically generated

This is addToCart.php. Test 1: confirmed

A screenshot of a cell phone

Description automatically generated

1. **The PHP page inserts valid details to the cart table in the database**

From the php page, the session variable “user” and the hidden value passed by the form is inserted into the cart table of the database.

A screenshot of a cell phone

Description automatically generated

This is the result of the insert

**A screenshot of a cell phone

Description automatically generated**

This is the row created in the cart table for the current user based on the book they selected

As you can see from the book selected in 1) and the table in 2), the proper bookid was added to the current users cart table. Test 2: confirmed

**A screenshot of a cell phone

Description automatically generated**

1. **The PHP page redirects to the checkout page**

The last statement of this page redirects to checkout.php

A screenshot of a cell phone

Description automatically generated

As you can see from the url, this is checkout.php. Test 3: confirmed

**A screenshot of a computer

Description automatically generated**

1. **The checkout page retrieves the current user’s specific cart table entries**

Below is the cart details for the current user

**A screenshot of a cell phone

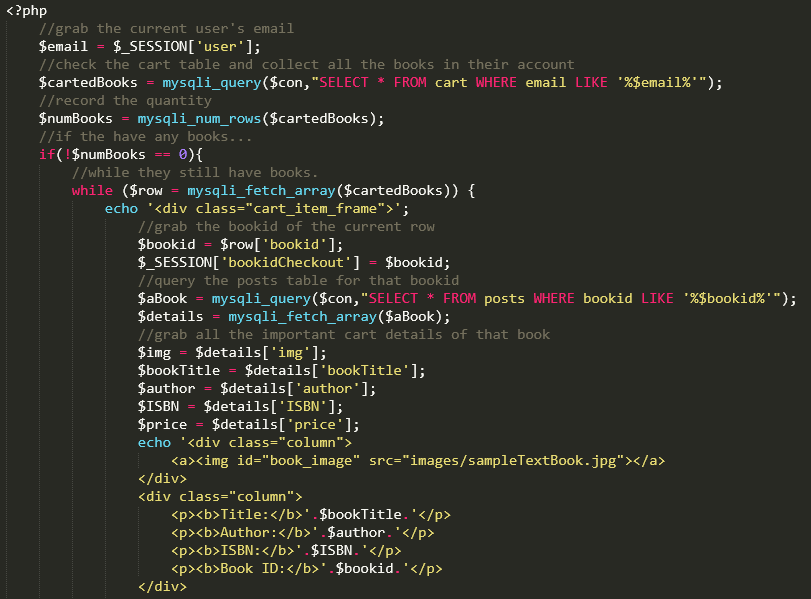
Description automatically generated**

This is the full details of the book that checkout.php retrieves using php and searching by bookid

**A screenshot of a cell phone

Description automatically generated**

Here is the php in checkout.php that grabs the valid details of the book and puts them in the current user’s cart



This is the current user’s checkout page. As you can see, the book this user selected is the accurate book in their cart. Test 4: confirmed

**A screenshot of a computer

Description automatically generated**

1. **The checkout page calculates the grand total of the user’s cart**

Below is a user with 2 books in their cart

**A screenshot of a computer

Description automatically generated**

Below is the php that calculates the grand total of all of the users carted books



As you can see: 11.90 + 12.65 = 24.55 Test 5: confirmed

**A screenshot of a computer

Description automatically generated**

**6) The “remove book” button deletes the correct item from the active user’s cart**

Below is a user with two books in their cart.

**A screenshot of a computer

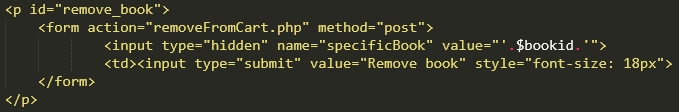
Description automatically generated**

Here is their entries in the cart table

**A screenshot of a social media post

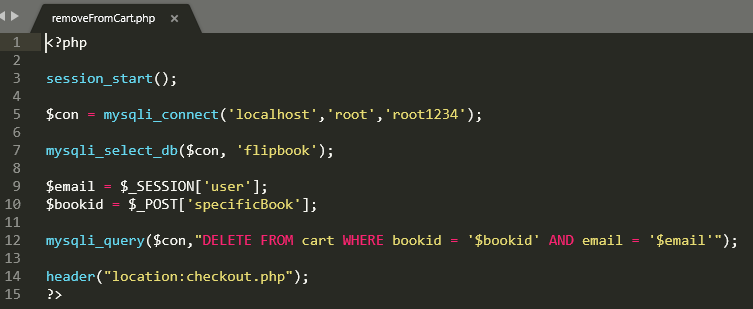
Description automatically generated**

If they select “Remove book”, they are redirected to removeFromCart.php using the following form inside checkout.php



Below is removeFromCart.php

As you can see, the session user and the hidden value of the book is used to delete the record in the cart table

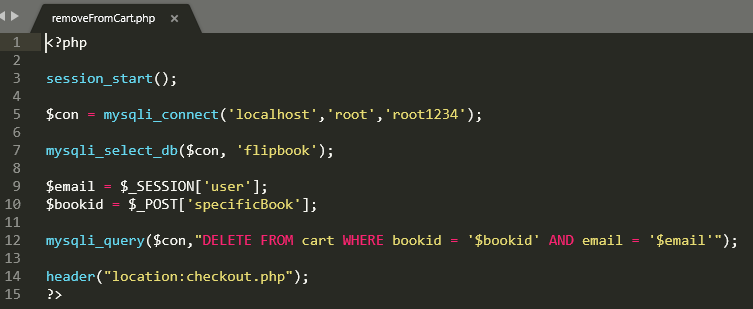


Below is the updated table

A screenshot of a cell phone

Description automatically generated

The last statement redirects back to checkout.php



Below is the updated cart. Test 6: Confirmed

**A screenshot of a computer

Description automatically generated**