

Group name: 404 NOT FOUND

Members: John Elmer Bobier Jr., Charls Emil C. Barquin, Christian Jeric Non, Rod Rañola, Vincent Macauyam

Course Yr. & Block: BSIS - 2A

Problem Number: 1

Problem Statement

Remove the Add Quantity Input if the Default Cart Item Qty is 1.

CODE CHANGES

Old Cart.php vs the New cart.php

We added select all/select checkbox:

```
<td><input type="checkbox"
name="selected_items[]" value="<?php echo
$row['id']; ?>" class="item-checkbox"
data-price="<?php echo $row['price'] *
$row['quantity']; ?>"></td>
```

And the major issue fix:

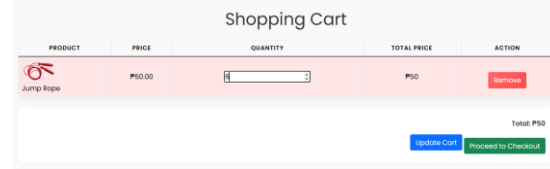
```
<form action="checkout-system.php"
method="post">
```

This is the reason why items were not passed from cart to checkout

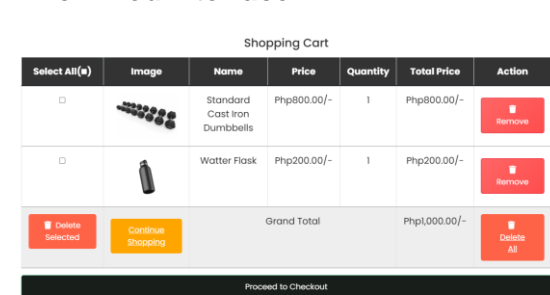
Above are the changed that make the cart functionality works

INTERFACE CHANGES

Old cart interface.



Error fixed interface.



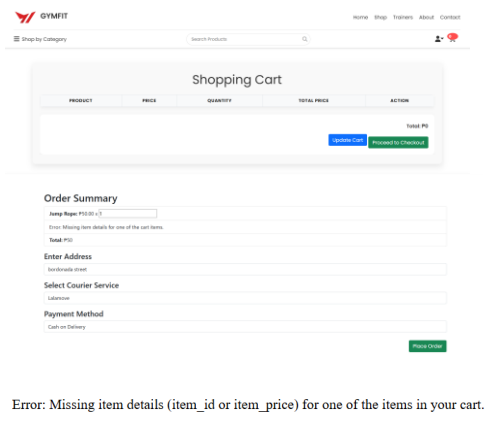
Describe the functionality added or Changed

The updated shopping cart has a better interface and now lets you handle multiple items at once, which makes it more convenient. You can't edit the quantity directly in the cart anymore, but we added some useful features like a "Select All" option to choose all items at once, buttons to "Delete Selected" or "Delete All" items quickly, and a "Continue Shopping" button to go back and add more items. These changes make the cart easier and more practical to use.

- Redesigned interface for a more organized look.
- Added support for handling multiple items in the cart.
- Removed the ability to edit quantities directly in the cart.
- Introduced a "Select All" option for

		<ul style="list-style-type: none"> bulk item selection. Added "Delete Selected" and "Delete All" buttons for easier item removal. Included a "Continue Shopping" button for seamless navigation.
--	--	---

Problem Number: 2
Problem Statement
Unable to Checkout. Need to make the Checkout Work.

CODE CHANGES	INTERFACE CHANGES	Describe the functionality added or Changed
<p>First we started with changes in cart.php we started using</p> <pre>\$user_id = \$_SESSION['user_id'];</pre> <p>And <code><form action="checkout-system.php" method="post" id="cart-form"></code></p> <p>in order to passed the items in cart to checkout-system.php</p>	<p>Old non-functional checkout interface</p> 	<p>The cart view is way simpler now, focusing on the essentials like product names, prices, quantities, and totals, which makes it easy to see what you're buying without any extra clutter. There's also a new error message that pops up if any item details are missing, helping users catch mistakes before they finish their orders. The action buttons, like "Place Order" and "Proceed to Checkout," are much clearer and stand out more, so it's easy to figure out what to do</p>

Next we created a new checkout-system.php

This is the link see what is the checkout-system.php:

<https://github.com/BOBIERJOHNELMER/WEB-SYSTEM-BSIS2A/tree/main/documentation>

Also the last added is check_orderdetails.php
Which gives you a receipt & send you an Email of that receipt.

Error fixed interface.

The screenshot displays a web application interface for a shopping cart and checkout process. At the top, there's a navigation bar with a logo and links like 'Home', 'Shop', 'Contact', 'About', and 'Cart'. Below the navigation bar, a 'Shopping Cart' section is visible, showing a table with columns: 'Item ID', 'Image', 'Name', 'Price', 'Quantity', 'Total Price', and 'Action'. The table contains two items: 'Earpods pro+ (1)' and 'Grand Total: Php 1095'. Below the cart, a 'Complete Your Order' section is shown, featuring a form with fields for 'Your Name', 'Your Number', 'Payment Method', 'Courier Service', 'Island', 'Province', 'City', 'Borough', and 'Street'. A red 'Order' button is at the bottom of the form. Below the form, a 'THANK YOU FOR SHOPPING!' message is displayed, along with a receipt summary: 'Earpods pro+ (1) Total: Php1095/-'. The receipt also includes the user's name, number, address, payment mode, and tracking number. A 'CONTINUE SHOPPING' button is at the bottom of the receipt.

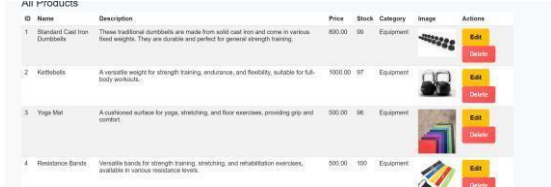
next. Plus, the overall design has taken on a minimalist vibe, with fewer visual distractions, which really improves usability.

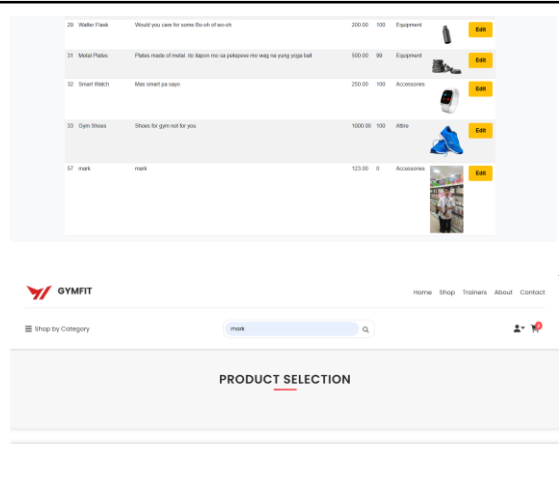
- Simplified Cart View
- Error Message
- Prominent Action Buttons
- Minimalist Design

Problem Number: 3

Problem Statement

Troubleshoot the delete button in Admin, though you need to make the delete, "Not to delete the entire item record as it will have an issue retrieving the past orders for that item.

CODE CHANGES	INTERFACE CHANGES	Describe the functionality added or Changed
<p>We remove the entire delete functionality</p> <pre>// Delete Product Logic if (isset(\$_GET['delete_id'])) { \$delete_id = \$_GET['delete_id']; \$sql_delete = "DELETE FROM items WHERE item_id = \$delete_id"; if (\$conn->query(\$sql_delete) === TRUE) { \$manage_product_message = "Product deleted successfully."; } else { \$manage_product_message = "Error deleting product: " . \$conn->error; } }</pre>	<p>Old non-functional checkout interface</p>  <p>Error fixed interface.</p>	<p>The previous version of the system included a button designed to delete products, but it caused errors during the process. To resolve this issue, we adopted a different approach based on product stock levels. When the stock of a product reaches zero, the product no longer appears on the index or client-facing page. Importantly, the system does not delete the product from the records; rather, it simply hides it from view when unavailable. This approach ensures that client purchase records are preserved while also preventing out-of-stock products from being displayed.</p>

		
--	--	--

Problem Number: 4
Problem Statement
System must land in the Index as a Landing page.

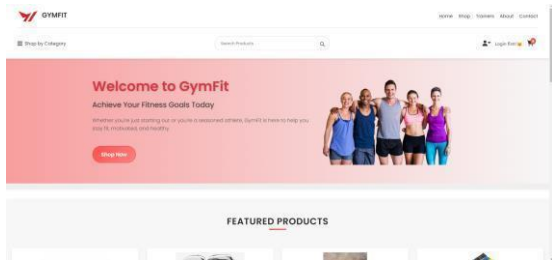
CODE CHANGES	INTERFACE CHANGES	Describe the functionality added or Changed
http://localhost/Gym-fit%20website/GYMFIT-INTERFACE2/php/	Old non-functional checkout interface	We tried adding “.htaccess” that contains “DirectoryIndex index.php” but the basic fix to this problem is to search the folder name along with the php folder to display the index or landing page of our website.

Index of /Gym-fit

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 admin/	2024-12-15 14:28	-	
 css/	2024-12-15 14:28	-	
 js/	2024-12-15 14:28	-	
 php/	2024-12-10 17:28	-	
 product_img/	2024-12-15 14:28	-	
 res_img/	2024-12-15 14:28	-	
 sign-in_sign-up/	2024-12-15 14:28	-	
 trainer_img/	2024-12-15 14:28	-	

Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12 Server at localhost Port 80

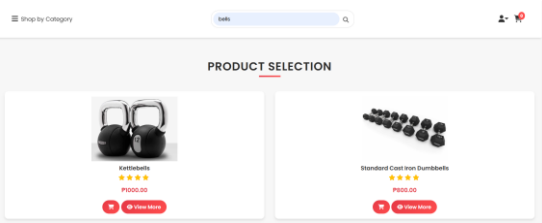
Error fixed interface.



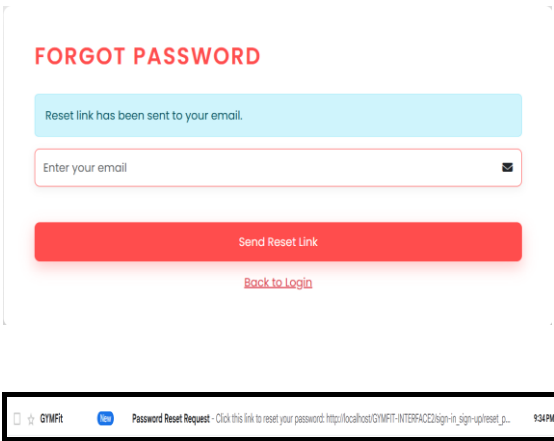
Problem Number: 5

Problem Statement

Troubleshoot the delete button in Admin, though you need to make the delete, "Not to delete the entire item record as it will have an issue retrieving the past orders for that item.

CODE CHANGES	INTERFACE CHANGES	Describe the functionality added or Changed
<pre> if (isset(\$_GET['search']) && !empty(\$_GET['search'])) { \$search_term = mysqli_real_escape_string(\$conn, \$_GET['search']); // SQL query to search for products by name or description \$sql = "SELECT * FROM items WHERE item_name LIKE '%\$search_term%' OR item_desc LIKE '%\$search_term%' ORDER BY item_name ASC"; } else if (isset(\$_GET['category_id'])) { // Display products by category \$category_id = \$_GET['category_id']; \$sql = "SELECT * FROM items WHERE category_id = \$category_id"; } else { // Default to showing all products \$sql = "SELECT * FROM items ORDER BY rand() "; } \$all_products = \$conn- >query(\$sql); ?> </pre>	<p>Error fixed interface.</p> 	<p>The added functionality introduces a dynamic filtering mechanism for product display based on user input or category selection. It prioritizes user-driven search by checking if a search term is provided (<code>\$_GET['search']</code>), sanitizing the input to prevent SQL injection, and querying products whose names or descriptions match the search term. If no search term is entered but a category ID is provided (<code>\$_GET['category_id']</code>), it retrieves products specific to that category. In the absence of both filters, the code defaults to displaying all products in a randomized order. This enhancement ensures a tailored user experience, allowing flexible browsing and refined product discovery.</p>

Problem Number: 6
Problem Statement
Troubleshoot the delete button in Admin, though you need to make the delete, "Not to delete the entire item record as it will have an issue retrieving the past orders for that item.

CODE CHANGES	INTERFACE CHANGES	Describe the functionality added or Changed
<pre> if (\$_SERVER['REQUEST_METHOD'] == 'POST') { \$email = \$_POST['email']; // Check if email exists \$sql = "SELECT * FROM user_account WHERE email='\$email'"; \$result = mysqli_query(\$conn, \$sql); if (mysqli_num_rows(\$result) == 1) { // Generate a unique reset token \$token = bin2hex(random_bytes(50)); // Save token in the database \$sql = "UPDATE user_account SET reset_token='\$token' WHERE email='\$email'"; mysqli_query(\$conn, \$sql); // Create the reset link </pre>	<p>Error fixed interface.</p> 	<p>The provided code handles the initial steps of the password reset process for a user requesting to reset their password. When the user submits their email through a POST request, the system checks if the email exists in the database by querying the <code>user_account</code> table. If the email is found, a unique reset token is generated using <code>random_bytes()</code> and stored in the database. A reset link is then created by appending the token as a query parameter to the URL. Finally, the code initializes PHPMailer, which will be used to send the reset link to the user's email address (though the email sending process is not fully implemented in the provided snippet). This functionality ensures that users can securely reset their passwords through a token-based verification system.</p>

<pre><i>\$reset_link =</i> <i>"http://localhost/GYMFIT</i> <i>INTERFACE2/sign-in_sign-</i> <i>up/reset_password.php?token=" . \$token;</i> <i>// Initialize PHPMailer</i> <i>\$mail = new PHPMailer(true);</i></pre>		
---	--	--

Group Name: 404 NOT FOUND

Name: John Elmer B. Bobier Jr

Role: Front-end Developer

Frontend Code Contributions:

- Cart Interface
- Checkout Interface
- Designer Interface
- Admin Orders Interface
- Admin Landing Page Interface
- Tester

Name: Charls Emil C. Barquin

Role: Front-end Developer

Frontend Code Contributions:

- Login Interface
- Signup Interface
- Landing Page Interface
- Shop Section Interface
- About Page Interface
- Contact Page Interface
- Navbar Section
- Footer Section
- Checkout Section
- Tester

Name: Rod B. Ranola

Role: Back-end Developer

Back-end Code Contributions:

- Landing Page functions
- Shopping cart functions
- Shopping Page functions
- Trainers Page functions
- Checkout Page functions
- Contacts Page
- Navbar Section functions
- Footer Section functions
- About Page
- Product Descriptions
- Mostly Customer Side features
- Tester

Name: Christian Jeric Non

Role: Back-end Developer

Back-end Code Contributions:

- Admin Function Developer
- Checkout Page
- Adding Products, Trainers, Contents
- Login/Logout
- Remember Password
- Registrations
- Accounts
- Databases
- Dashboard
- Receipt
- Mostly Admin Side features
- Tester

Name: Vincent Macauyam

Role: Front-end Developer

Frontend Code Contributions:

- Checkout Interface
- Designer Interface
- Admin Orders Interface
- Tester

