Table of Contents

E-commerce Cart System: Technical Test Learning Process Documentation

Cheah Kate-Lynn

cheahkatelynn@gmail.com

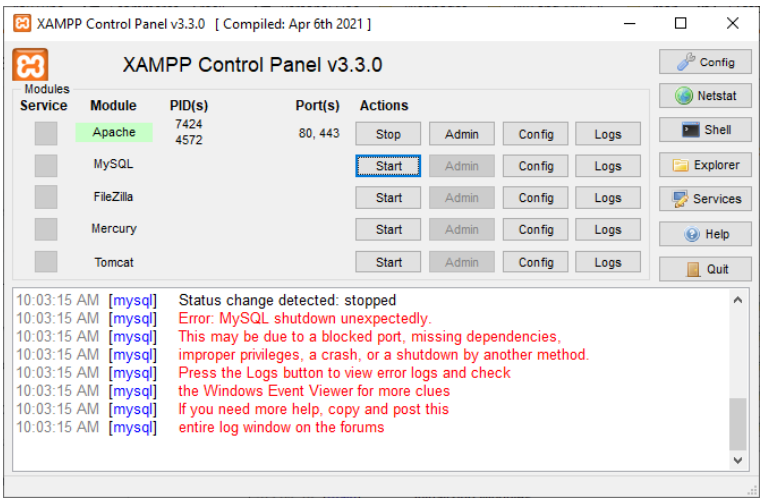
1.0 Introduction

The purpose of this project is to create a simple e-commerce cart system using Laravel, ideally using the TALL stack. This system enables users to add items to a shopping cart, remove items, and calculate the total price based on the cart items added. Additionally, the system is designed to incorporate a basic discount mechanism, applying discounts based on specific total price thresholds. The goal of this project is to showcase efficient use of object-oriented programming principles, facilitate seamless data interaction with MySQL, and demonstrate a user-friendly interface. This documentation will cover the approach taken, design decisions, challenges encountered, and lessons learned throughout the development process.

2.0 Initial Setup & Learning Process

2.1 Installing Laravel

To begin, I followed a YouTube tutorial to install Laravel, which required setting up and installing XAMPP, Composer, and Laravel in Visual Studio Code's terminal. This setup aimed to create a local development environment where I could use Laravel features. However, I encountered several issues related to MySQL configuration in XAMPP, particularly connecting the database in Laravel. A recurring error was:



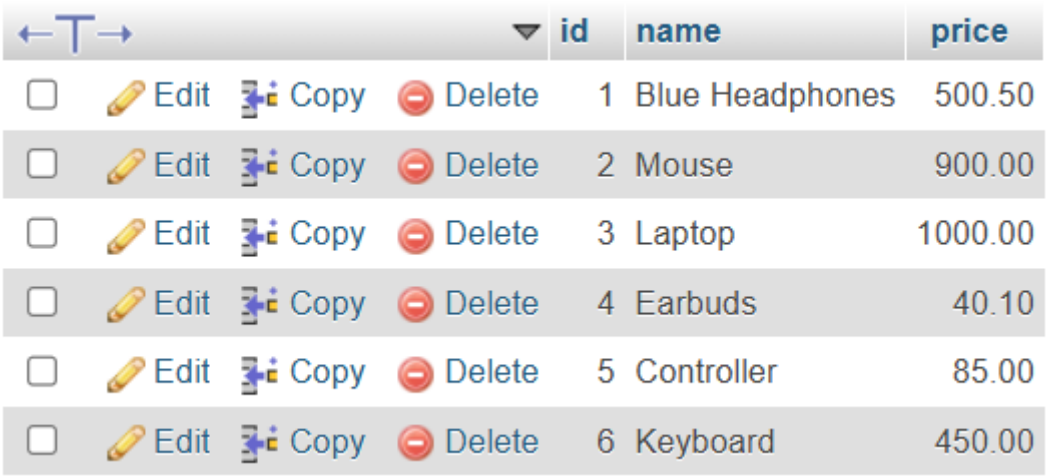
Additionally, I faced an access issue:

SQLSTATE[HY000] [1045] Access denied for user 'root'@'localhost' (using password: NO). DB_HOST set to localhost

To resolve these, I referred to forums like Stack Overflow and found possible solutions. The key to resolving these errors were to:

- Disconnect and close any other MySQL instances that might be interfering.
- Restart the MySQL service in XAMPP.
- Execute the PHP artisan migrate command after resolving the MySQL connection.

After solving this issue, a product table was created in MySQL with an id, name and price which allows each product's data to be stored and retrieved in the database without hardcoding it.

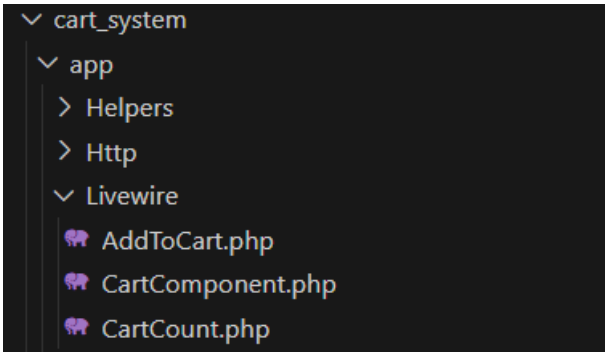


	id	name	price
Edit Copy Delete	1	Blue Headphones	500.50
Edit Copy Delete	2	Mouse	900.00
Edit Copy Delete	3	Laptop	1000.00
Edit Copy Delete	4	Earbuds	40.10
Edit Copy Delete	5	Controller	85.00
Edit Copy Delete	6	Keyboard	450.00

2.2 Implementing Livewire

After setting up the foundational classes and functions, I moved on to implementing Livewire, which allowed users to add items, adjust quantities, and remove items from the cart without needing to refresh the page each time.

Firstly, I installed Livewire by following its documentation, adding it to the project with composer require livewire/livewire. Once installed, I created several Livewire components which were CartComponent, CartCount, and AddToCart. Each of these are designed to manage specific tasks within the cart.



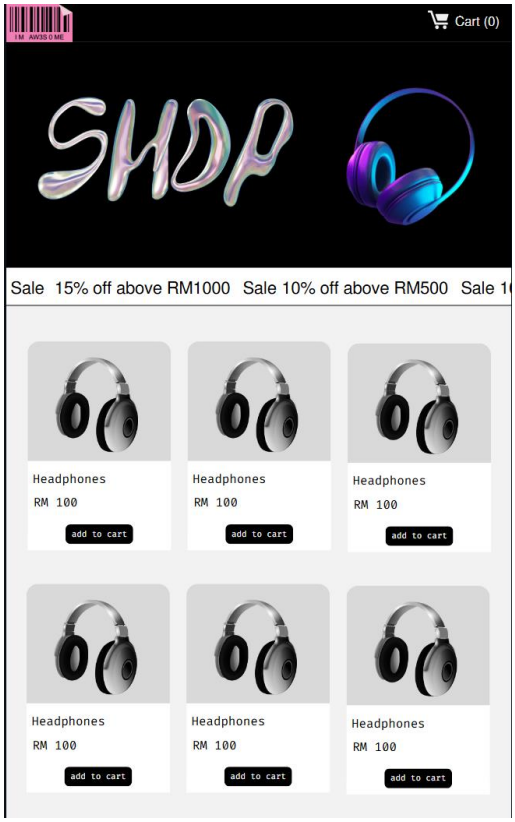
The CartComponent handled displaying cart items and performing core actions, like updating quantities and calculating totals, while CartCount managed the display of the cart's item count, ensuring users see real-time updates. Meanwhile, the AddToCart component made adding items a seamless process, therefore reflecting changes in both the item count and cart contents.

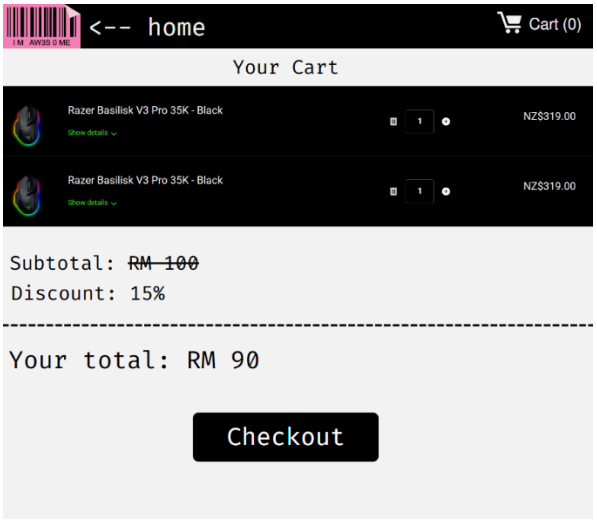
However, the integration presented a few challenges, such as data consistency across components and session management. Resolving these issues required referring to Livewire's documentation and finding ways to solve these issues on community forums. The solution was to clear session data in the SQL with each restart.

2.3 Interface Design

After establishing the base of the program with all necessary functions, I recognized the need for a design to enhance user experience and improve overall usability.

To visualize this, I created a storyboard for the shop page using Canva, which led to the final prototype design.





Implementing this design involved utilizing CSS, starting with the header design for both the cart and shop pages. The header utilizes Blade UI Kit Icons for the cart label, ensuring users can navigate it.

Moving on to the shop page, I designed it to utilize containers and images that correspond to the product names stored in the SQL database. This approach allows each image to update automatically based on the corresponding product name, increasing efficiency and reducing the need for manual adjustments. The layout of each container is also automatically positioned without hard coding, which simplifies the design process and improves device responsiveness.

Lastly, the cart page is designed to display items with their assigned images. It includes a functionality to calculate discounts. For example, a 10% discount for orders over RM500 and a 15% discount for over RM1000. Users can easily remove or update items using the provided buttons, all without needing to refresh the page, with the integration of Livewire.

However, this design process faced challenges. I encountered issues related to CSS alignment and had trouble implementing the code that allowed images to displayed correctly with their respective product names. Additionally, I had to troubleshoot various layout inconsistencies across different window sizes. By iterating on these designs and extensive testing, I was able to create a simple and user-friendly interface.

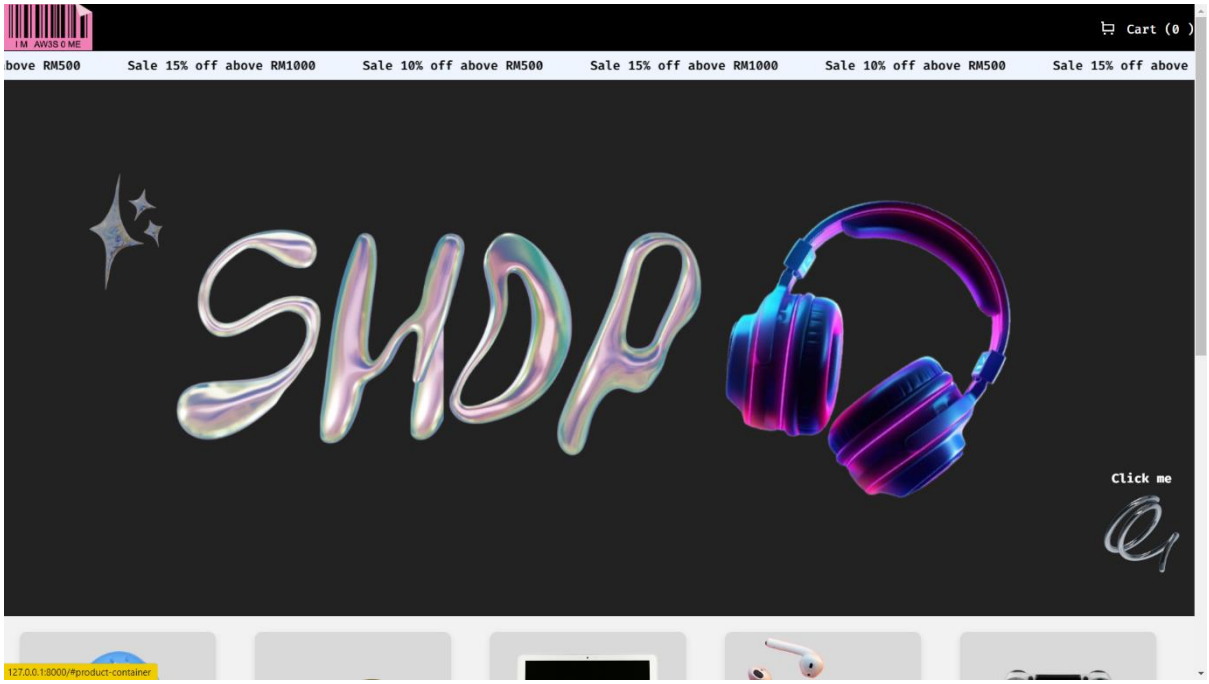
3.0 Final System Overview

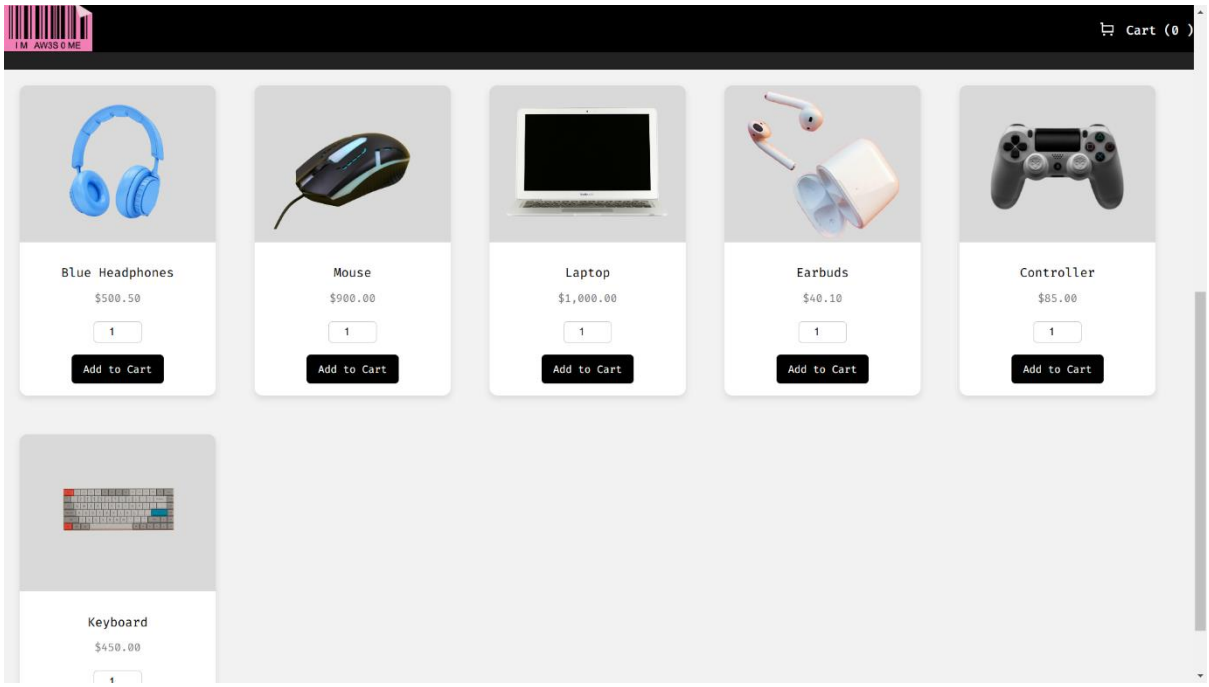
The screenshots below showcase the final version of this e-commerce cart system, highlighting its functionalities, including the shop and cart pages.

This includes basic CRUD (Create, Read, Update, Delete) functionalities that allows users to:

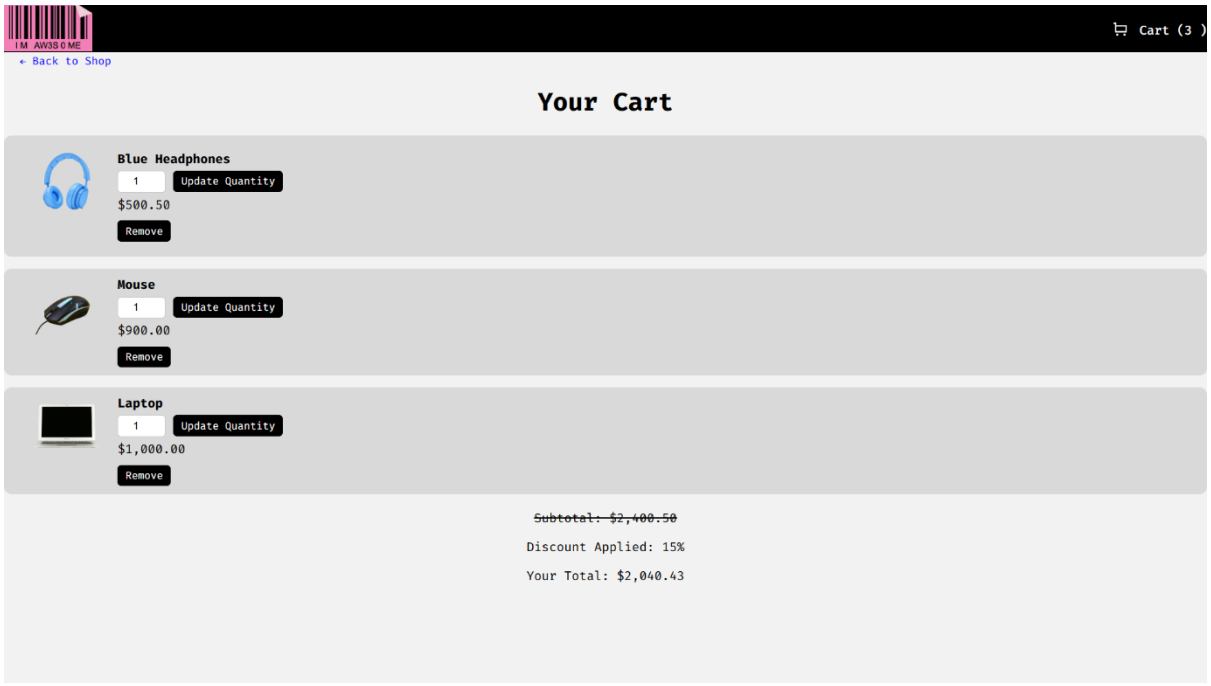
- View the list of available products
- Add products to the cart
- Remove products from the cart
- View the current cart contents and total price

The home page displays the products along with their names, prices, a quantity selector for users to choose the desired amount, and an "Add to Cart" button.





The "Add to Cart" page implements a simple discount system, displays the total costs and discount applied.



4.0 Reflection and Conclusion

This project has been a week-long learning journey, filled with both challenges and rewarding discoveries. Starting from scratch with an entirely new framework was initially daunting, but it allowed me to gain valuable insights and develop a practical skill set in Laravel and TALL stack. Even within this short period, I've learned a lot and have gained a deeper understanding of the concepts. I am grateful for the experience and am committed on building on these foundations. Overall, this has been a fulfilling and insightful process that I'll carry forward in my work.