

# E-Commerce Dashboard Web Application

## Project Report

**Student Name:** James Baah

**Course:** Web-Database Application CSC 600 50

**Instructor:** Dr. Wiseborn Danquah

**Date:** February 28, 2026

## 1. Introduction

This project focuses on the design, development, and deployment of a modern E-Commerce Dashboard Web Application. The main idea behind the project was to simulate a simplified online shopping system where users can browse products, add items to a shopping cart, adjust quantities, and view total price calculations in real time.

The application was built using Next.js and React, along with TypeScript and Tailwind CSS for styling. The project demonstrates practical frontend development skills and the ability to deploy a working application to a live hosting platform.

The final result is a responsive web application that includes a product listing page, shopping cart functionality, dynamic total price updates, and a clean dashboard-style interface. The project was successfully deployed using Vercel and is publicly accessible online.

## 2. Project Objectives

The main objectives of this project were:

- To build a functional product listing page
- To implement a fully working shopping cart
- To allow users to add and remove products
- To dynamically calculate total prices
- To design a responsive and clean user interface
- To deploy the application using a free hosting platform

The goal was not just to create a static UI, but to build an interactive application that behaves like a real-world e-commerce system.

## 3. Technologies Used

Several modern web technologies were used in this project.

### 3.1 Next.js

Next.js was used as the main framework for building the application. It provides built-in routing, optimized performance, and a clean project structure. Using Next.js helped organize the application into different pages such as the dashboard, products page, and cart page.

### 3.2 React

React was used to build reusable components such as product cards, cart rows, and layout elements. React's state management system made it easier to handle dynamic updates in the shopping cart.

### 3.3 TypeScript

TypeScript was used to improve code reliability. It helped prevent common errors by enforcing proper data types, especially when managing the cart state and product data.

### 3.4 Tailwind CSS

Tailwind CSS was used for styling the application. It allowed fast UI development using utility classes. It also made it easier to create a responsive layout that works well on different screen sizes.

### 3.5 Git and GitHub

Git was used for version control, and GitHub was used to host the repository. This ensured proper tracking of changes and integration with deployment tools.

### 3.6 Vercel

Vercel was used to deploy the application. It connects directly to GitHub and automatically builds and deploys the project whenever updates are pushed.

## 4. Application Structure

The project follows a component-based structure using Next.js App Router.

The main sections include:

- **Dashboard Page** – Displays summary cards and navigation
- **Products Page** – Shows available products with “Add to Cart” functionality
- **Cart Page** – Displays selected items and total price

Reusable components were stored in a separate folder, and cart logic was handled using React Context to allow global state access across different pages.

This structure keeps the code organized and easier to maintain.

## 5. Core Features

### 5.1 Product Listing

The product page displays items with:

- Image
- Product title
- Category
- Price
- Rating
- Add to Cart button

When the “Add to Cart” button is clicked, the selected item is added to the shopping cart immediately.

### 5.2 Shopping Cart Functionality

The cart page allows users to:

- View added products
- Increase item quantity
- Decrease item quantity
- Remove individual items
- Clear the entire cart

All updates happen instantly without reloading the page. This was achieved using React Context API for global state management.

### 5.3 Dynamic Price Calculation

One of the important features of the project is automatic price calculation.

Each product subtotal is calculated using:

Price × Quantity

The total price is then calculated by summing all product subtotals.

Whenever a user updates quantity or removes an item, the total updates automatically.

## 5.4 Responsive Design

The interface was designed to be responsive. The layout adapts to different screen sizes using Tailwind CSS responsive classes.

On larger screens, the sidebar remains visible. On smaller screens, layout adjustments ensure readability and usability.

The goal was to maintain a clean and professional dashboard look while ensuring accessibility across devices.

## 6. Deployment

After development was completed, the project was deployed using Vercel.

The deployment process involved:

1. Initializing Git repository
2. Pushing project files to GitHub
3. Connecting GitHub repository to Vercel
4. Automatic build and live deployment

The application is now publicly accessible via a live Vercel URL.

This demonstrates understanding of real-world deployment workflow and cloud hosting.

## 7. Challenges Faced

During development, a few challenges were encountered.

One challenge was managing cart state across different pages. Initially, some functions were not updating properly due to incorrect context implementation. This was resolved by refining the Cart Context structure.

Another challenge involved TypeScript errors when defining cart types and state. These were solved by properly defining interfaces and ensuring consistent data structures.

There were also minor UI adjustments needed to ensure layout consistency and responsiveness.

These challenges improved debugging and problem-solving skills.

## 8. Learning Outcomes

This project helped strengthen practical knowledge in:

- React state management
- Context API usage
- TypeScript typing and debugging
- Responsive UI design
- Git version control
- Cloud deployment using Vercel

It also improved understanding of how modern web applications are structured and deployed.

## 9. Conclusion

The E-Commerce Dashboard Web Application successfully demonstrates the implementation of core online shopping features using modern web technologies.

The system allows users to browse products, manage a shopping cart, and view real-time total price updates in a responsive interface.

The project not only shows frontend development skills but also demonstrates the ability to deploy a live production-ready application.

Overall, this project reflects practical application of web development concepts and real-world deployment processes.