An Undergraduate Internship/Project on Clothing Website: Design & Development

By

Joy Tarafder

Student ID: **2022042**

Spring, 2025

Supervisor:

Sarwar Shahidi

Research & Development Officer

Department of Computer Science & Engineering

Independent University, Bangladesh

June 26, 2025

Dissertation submitted in partial fulfillment for the degree of Bachelor of Science in Computer Science

Department of Computer Science & Engineering

Independent University, Bangladesh

Attestation

This is to certify that I, Joy Tarafder, have completed the report titled "Clothing Website: Design & Development" and submitted it in partial fulfilment of the requirement for the Degree of Computer Science and Engineering from Independent University, Bangladesh. It has been completed under the guidance of my university supervisor Sarwar Shahidi and company supervisor Mr. N M RAFSAN JANY SAMIR. This work has not been submitted as a project to this University previously, neither has it been submitted to any other institution. All the sources of information used in this Project Report has been duly acknowledged in it.

Signature	Date
Joy Tarafder	
Name	

Acknowledgement

First and foremost, I would like to express my deepest sense of gratitude to Almighty Allah, it is because of His mercy and blessing that gave me the motivation and strength to work hard during my internship.

I would like to extend my sincere thanks to Cloud Coder for providing me with the invaluable opportunity to undertake my internship. My deepest gratitude goes to the Managing Director and CEO of Cloud Coder, Mr. N M RAFSAN JANY SAMIR, for his exceptional guidance, unwavering support, and constant motivation throughout my internship period. His insights and trust in my abilities were instrumental in my learning and growth, and I will be forever grateful for the opportunity to work under his mentorship.

My internship at Cloud Coder provided me with a practical understanding of software development processes and the professional environment. I am also thankful to the entire team of software engineers at Cloud Coder who welcomed me, shared their knowledge, and guided me through the intricacies of the project. Their collaborative spirit and willingness to help significantly contributed to my learning experience.

Last but not the least, I would like to thank my parents, other family members, and friends for their constant support and encouragement.

Joy Tarafder
June, 2025
Dhaka, Bangladesh

Letter of Transmittal

June 26, 2025

Sarwar Shahid
Research & Development Officer
School of Computer Science and Engineering
Independent University Bangladesh

Subject: Submission of Internship Report

Dear Sir,

It is with great pleasure that I am presenting my internship report on the project **"Clothing Website: Design & Development"**. This project was undertaken as part of my internship at Cloud Coder for a period of 12 weeks, under the esteemed supervision of Mr. N M RAFSAN JANY SAMIR, Managing Director and CEO of Cloud Coder.

This experience provided me with an invaluable opportunity to apply the theoretical knowledge gained during my undergraduate studies to real-world scenarios in software development. I have diligently worked to ensure that this report accurately reflects my contributions, learning, and the insights gained throughout this period.

I am hoping that this report will be informative, comprehensive, and meet your expectations. I have made every effort to ensure its accuracy and completeness.

Thank you once again for your continuous guidance and the opportunity to submit this report.

Sincerely,
Joy Tarafder
ID: 2022042

Evaluation Committee

Supervision Panel

.....	.....
Academic Supervisor	Industry Supervisor

Panel Members

.....	.....
Panel Member 1	Panel Member 2
.....	.....
Panel Member 3	Panel Member 4

Office Use

.....	.....
Internship Coordinator	Head of the Department

.....

Industry Coordinator of the Department

Abstract

This internship report details the design and development of a web-based Clothing Website, a comprehensive e-commerce platform. The primary objective was to create a robust and user-friendly online store, facilitating seamless browsing, selection, and purchase of clothing items for customers, while providing an efficient management system for the business. This initiative addresses the growing demand for accessible online retail experiences and supports the digital presence of clothing businesses.

The system is a full-stack web application. The frontend user interface is crafted using React and JavaScript, ensuring a dynamic and intuitive browsing experience. For the backend, Node.js provides a powerful server-side environment, handling business logic and API interactions. All data is securely managed within a MongoDB NoSQL database. Key functionalities include user registration, product browsing with filtering, a shopping cart, secure checkout, and order tracking. Administrative features for product and inventory management are also integrated.

This Abstract serves as a critical communication instrument, conveying the project's core value proposition, the market need it addresses, the technical approach, and the significant professional growth experienced. This ensures various stakeholders can quickly grasp the essence and impact of the work.

During the internship at Cloud Coder, substantial personal and professional development occurred. This included gaining hands-on proficiency in full-stack web development with React, Node.js, and MongoDB, providing a deep understanding of modern web architecture. The immersion in a corporate software development environment offered invaluable insights into real-world software engineering processes, particularly the complexities of designing and implementing a complete e-commerce solution.

Keywords— Clothing Website, E-commerce, React, Node.js, MongoDB, Web Development, Full-Stack, Online Store, Cloud Coder

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview/Background of the Work

In today's rapidly evolving digital landscape, the e-commerce sector has emerged as a dominant force, fundamentally transforming consumer purchasing habits and business operations globally. The convenience and accessibility of online shopping have led to a significant surge in online transactions, making it imperative for businesses, including those in the clothing industry, to establish a robust digital presence. This internship project, titled *"Clothing Website: Design & Development,"* addresses this growing demand by focusing on creating a comprehensive e-commerce platform tailored for the clothing retail market.

The project's core involves the design and development of a full-stack web application that provides a seamless online shopping experience for customers and an efficient management system for the business. This initiative aims to overcome the limitations of traditional brick-and-mortar stores by offering a platform that is accessible anytime, anywhere, to a global customer base [**?**, **?**]. The development leverages modern web technologies to ensure scalability, security, and high performance. The frontend is built using React and JavaScript, enabling a dynamic, responsive, and intuitive user interface. The backend is powered by Node.js, providing a robust server-side environment for handling business logic and API interactions. All critical data, including product catalogs, user profiles, and transaction records, are securely managed within a MongoDB NoSQL database.

This project was undertaken as part of an internship at Cloud Coder, a company dedicated to delivering innovative software solutions. The work involved understanding the intricacies of e-commerce platforms, designing user-centric interfaces, and implementing scalable backend services to support a modern online retail operation.

1.2 Objectives

The primary objectives of the *Clothing Website: Design & Development* project are to:

- **Provide a Seamless Online Shopping Experience:** To create an intuitive and user-friendly platform that allows customers to effortlessly browse, select, and purchase clothing items from anywhere with an internet connection .
- **Facilitate Efficient Business Operations:** To develop a robust backend system that enables the business to effectively manage product inventory, process orders, track sales, and handle customer data.
- **Enhance Market Reach and Revenue:** To establish a digital storefront that can reach a wider customer base beyond geographical limitations, thereby contributing to increased sales and business growth.
- **Ensure Data Security and Reliability:** To implement secure authentication, authorization, and data storage mechanisms to protect sensitive customer and business information, ensuring the reliability and integrity of the platform.
- **Leverage Modern Technologies:** To gain practical experience in designing and developing a full-stack application using React, JavaScript, Node.js, and MongoDB, applying theoretical knowledge to a real-world e-commerce solution.
- **Improve Customer Satisfaction:** To offer features such as easy navigation, clear product information, and efficient order processing to enhance overall customer satisfaction and foster loyalty.

1.3 Scopes

The scope of the *Clothing Website: Design & Development* project encompasses the following key functionalities and areas:

- **User Management:**
  - User registration and authentication (login/logout).
  - User profile management (viewing and editing personal details).
  - Password reset functionality.
- **Product Management:**
  - Displaying a catalog of clothing products with details (images, descriptions, prices, sizes, colors).

  - Product categorization and filtering options.
  - Search functionality for products.
  - Admin interface for adding, editing, and deleting products.

- **Shopping Cart and Checkout:**
  - Adding and removing items from a shopping cart.
  - Updating quantities in the cart.
  - Secure checkout process.
  - Integration with a payment gateway (conceptual, not necessarily live implementation).

- **Order Management:**
  - Order placement and confirmation.
  - Order history for users.
  - Admin interface for viewing and managing orders (status updates, fulfillment).

- **Database Management:**
  - Maintaining a database for users, products, and orders using MongoDB.
  - Ensuring data integrity and relationships.

- **Frontend Development:**
  - Responsive user interface design using React and JavaScript.
  - Client-side routing and state management.

- **Backend Development:**
  - API development using Node.js to handle requests from the frontend.
  - Business logic implementation for e-commerce operations.
  - Interaction with the MongoDB database.

- **Security:**
  - Basic user authentication and authorization.
  - Data encryption for sensitive information (e.g., passwords).

- **Reporting (Conceptual):**
  - Basic reporting capabilities for sales and inventory (e.g., number of orders, popular products).

Chapter 2

Literature Review

2.1 Relationship with Undergraduate Studies

The development of the *Clothing Website* project during this internship provided a practical application of the theoretical knowledge and skills acquired throughout my undergraduate studies in Computer Science and Engineering at Independent University, Bangladesh. Several core courses were particularly instrumental in preparing me for the challenges and complexities of this full-stack web development endeavor:

Object-Oriented Programming (CSE213): This course was fundamental in shaping my approach to structuring the frontend with React components and organizing backend logic in Node.js. The principles of encapsulation, inheritance, and polymorphism were crucial for designing modular, reusable, and maintainable code, which is essential for a scalable e-commerce platform.

Database Management (CSE303): Understanding database concepts was paramount for working with MongoDB. This course provided insights into data modeling, schema design (even for NoSQL databases like MongoDB), data storage, retrieval, and manipulation. Knowledge of query languages and database optimization techniques directly applied to managing product catalogs, user profiles, and order information efficiently.

Web Applications and Internet (CSE309): This course equipped me with the foundational knowledge of web application architectures, client-server communication, and core web technologies. Proficiency in HTML, CSS, and JavaScript, gained from this course, was directly applied in building the interactive and responsive user interface with React. Understanding HTTP protocols and web security concerns was also vital for developing a secure e-commerce site.

System Analysis and Design (CSE307): This course provided a systematic framework for

approaching software development. Techniques such as requirements gathering (functional and non-functional), system analysis, and design methodologies (e.g., UML diagrams for conceptualizing system flow) were invaluable in planning the Clothing Website, defining its features, and structuring its components before implementation.

Software Engineering (CSE451): This course introduced industry best practices for software development lifecycles (SDLC), project management, and quality assurance. Concepts like agile methodologies, version control, testing strategies (unit, integration, user acceptance testing), and deployment considerations were directly applicable to managing the project's lifecycle, ensuring code quality, and preparing for deployment.

These courses collectively provided a robust academic foundation, enabling me to effectively contribute to the design, development, and implementation of a complex e-commerce solution in a professional environment.

2.2 Related works

The concept of online retail and e-commerce platforms has fundamentally transformed the global marketplace, offering consumers unprecedented convenience and businesses expanded reach. A clothing website, as an e-commerce platform, optimizes online sales and operational procedures, enabling virtual storefronts, product display, and seamless transaction management.

E-commerce applications provide numerous benefits, including global reach, allowing sales worldwide and transcending geographical limitations. They offer significant customer convenience, enabling purchases anytime, anywhere, and often lead to cost-effectiveness by reducing physical infrastructure needs. Furthermore, these platforms enhance productivity through automation of inventory control, order processing, and payment handling, and improve transparency with detailed product and shipping information.

The development of e-commerce platforms using modern technology stacks, such as the MERN (MongoDB, Express.js, React.js, Node.js) stack, has gained significant traction. This is due to its full-stack capabilities and suitability for building scalable, secure, and high-performance web applications. Research indicates that MERN stack-based e-commerce platforms deliver seamless user experiences, fast loading times, and efficient management of products, categories, and orders through distinct user and admin interfaces. Key features commonly implemented include shopping carts, wish lists, and integrated payment gateways, all contributing to a comprehensive online retail solution. This project aligns with these advancements, aiming to leverage the strengths of the MERN stack to deliver a truly robust and competitive online clothing store.

Chapter 3

Project Management & Financing

Effective project management and meticulous financial planning are crucial for the successful execution of any software development endeavor. This section outlines the structured approach taken for the "Clothing Website: Design & Development" project, ensuring clarity in tasks, timelines, resource allocation, and financial considerations.

3.1 Work Breakdown Structure

A Work Breakdown Structure (WBS) is a hierarchical decomposition of the total scope of work to be carried out by the project team to accomplish the project objectives and create the required deliverables. It breaks down complex projects into smaller, more manageable components, facilitating better planning, execution, and control. For the *Clothing Website: Design & Development* project, the WBS is structured as follows:

1. **Project Initiation**
    - Define Project Scope & Objectives
    - Stakeholder Identification
    - Team Formation
    - Initial Planning & Kick-off Meeting
2. **Requirement Analysis**
    - Gather Functional Requirements (User Stories, Features)
    - Gather Non-Functional Requirements (Performance, Security, Usability)
    - User Persona Development
    - Create Use Case Diagrams

- Finalize Requirement Specification Document

3. **Design Phase**
    - Database Design (Schema, Relationships for MongoDB)
    - UI/UX Design
        - Wireframing & Mockups
        - Prototype Development
        - User Flow Design
        - Visual Design (Color Palette, Typography, Iconography)
    - System Architecture Design (Frontend-Backend Interaction)
    - API Design & Documentation
4. **Development**
    - Frontend Development (React.js, JavaScript, TailwindCSS)
        - Component Development (Product Listings, Cart, Checkout)
        - State Management Implementation
        - Responsive Design Implementation
        - API Integration
    - Backend Development (Node.js, Express.js, MongoDB)
        - Database Setup & Configuration
        - API Endpoint Creation (e.g., Authentication, Product, Order)
        - Business Logic Implementation
        - Authentication & Authorization Implementation
    - Admin Dashboard Development
5. **Testing**
    - Unit Testing (Individual Components/Functions)
    - Integration Testing (Frontend-Backend API Calls)
    - System Testing (End-to-End Functionality)
    - User Acceptance Testing (UAT)
    - Bug Fixing & Regression Testing
6. **Deployment & Launch**
    - Server Setup & Configuration
    - Database Migration (if applicable)
    - Application Deployment

- Domain Configuration
- Post-Deployment Monitoring

7. **Maintenance & Support**
   - Bug Fixing (Post-Launch)
   - Performance Monitoring
   - Feature Enhancements (Future Iterations)

3.2 Process/Activity wise Time Distribution

The **Critical Path Method (CPM)** is a project management technique that identifies the critical path, which is the longest sequence of activities that must be completed on time for the entire project to be completed by its deadline. Any delay in these critical activities will directly delay the entire project. For this project, assuming a typical internship duration of approximately 12 weeks (60 working days), the time distribution for key activities is estimated as follows:

Activity Phase	**Estimated Duration (Working Days)**
1. Project Initiation	3
2. Requirement Analysis	10
3. Design Phase	12
4. Development	25
5. Testing	8
6. Deployment & Launch	2
Total Estimated Working Days	**60**

Table 3.1: Estimated Duration of Each Project Phase

This distribution ensures that adequate time is allocated to each phase, with development consuming the largest portion, followed by design and testing. The critical path would generally run through:

as each subsequent phase is dependent on the completion of the previous one. Monitoring this path is essential for ensuring the timely delivery of the project within the internship period.

3.3 Gantt Chart

A Gantt chart is a visual representation of a project schedule, showing the start and end dates of various activities, along with their dependencies. It helps in visualizing the project timeline,

Figure 3.1: Critical Path of the Project Proces

tracking progress, and managing tasks effectively. Below is a simplified Gantt chart illustrating the timeline for the *Clothing Website: Design & Development* project over a 12-week period.

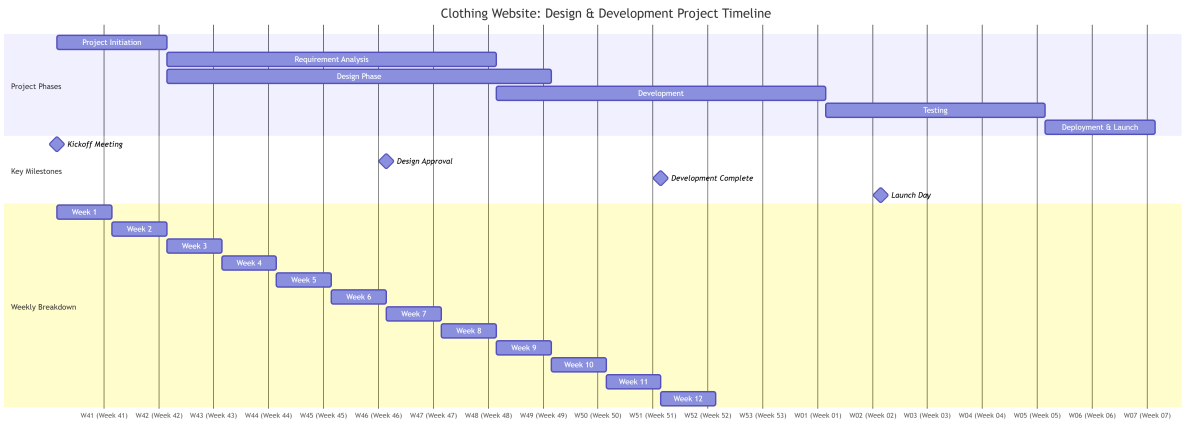


Figure 3.2: Gantt Chart representing the project timeline

3.4 Process/Activity wise Resource Allocation

Resource allocation involves assigning the appropriate human resources, tools, and time to various phases of the project to ensure timely and efficient completion. Since this internship project was executed individually, the primary focus of resource allocation revolves around human effort and technical tools.

Human Resource Allocation (Based on Roles)

Table 3.2: Project Phases with Corresponding Roles and Estimated Effort

Project Phase	**Role/Responsibility**	**Estimated Effort (%)**
Project Initiation	Planning, defining scope, and setting objectives	5%
Requirement Analysis	Gathering, analyzing, and documenting requirements	10%
Design Phase	UI/UX design, database schema, architecture	15%
Development (Frontend)	React components, styling, API integration	25%
Development (Backend)	API development, business logic, database	25%
Testing	Writing test cases, fixing bugs	10%
Deployment & Launch	Hosting setup, deployment	5%
Maintenance (Post-Launch)	Minor bug fixes, enhancements	5%

Note: All development and testing activities were performed solely by the intern under the guidance of supervisors.

Technical Resources Used per Activity

Table 3.3: Development Activities and Associated Tools/Technologies

Activity	**Tools/Technologies Involved**
Design	Figma (UI mockups), Lucidchart (UML diagrams)
Frontend Development	React.js, TailwindCSS, JavaScript
Backend Development	Node.js, Express.js
Database	MongoDB Atlas (for local testing), Mongoose ORM
API Testing	Postman
Version Control	Git & GitHub
Deployment (Conceptual)	Render / Vercel / Netlify (Frontend), MongoDB Atlas
Documentation	Microsoft Word / Overleaf (for report writing)

3.5 Estimated Costing

Although the *Clothing Website: Design & Development* project was conducted as part of an academic internship, estimating the development cost helps understand the financial aspects of launching a similar real-world application. The following chart outlines the estimated costing based on typical software development expenses.

Table 3.4: Estimated Project Cost Breakdown

Cost Component	**Description**	**Estimated Cost (BDT)**
Human Resources	Developer's effort over 12 weeks (stipend/honorarium)	30,000
Software Tools	All tools used were open-source (React, Node.js, MongoDB, etc.)	0
Development Equipment	Laptop, internet, peripherals (amortized for project)	5,000
Domain & Hosting	Conceptual (1 year shared hosting & .com domain)	3,000
Miscellaneous	Electricity, travel (if applicable), maintenance	2,000
Total Estimated Cost	—	**40,000 BDT**

Explanation

- **Human Resources:** Though unpaid as an intern, the value of the work is estimated based on an entry-level developer rate ($\tilde{2}$,500 BDT/week).
- **Software Tools:** The MERN stack is open-source, which eliminates licensing costs.
- **Development Equipment:** Usage of personal devices and resources for work is partially estimated.
- **Hosting:** Cost is assumed based on shared hosting providers in Bangladesh (e.g., ExonHost, Hostever).
- **Miscellaneous:** Covers any minor unexpected expenses.

Chapter 4

Methodology

The successful development of any software project hinges on the adoption of a structured and systematic approach, commonly referred to as the **System/Software Development Life Cycle (SDLC)**. The SDLC provides a framework that guides software engineers through various phases of software creation, from initial conceptualization to deployment and ongoing maintenance. Its purpose is to ensure the development of high-quality applications efficiently, by maximizing output, minimizing costs, and adhering to project timelines. Key phases within an SDLC typically include *Requirement Analysis*, *Planning*, *Design*, *Development*, *Testing*, and *Deployment*.

For the *Clothing Website: Design & Development* project, the **Waterfall Model** was adopted as the primary SDLC methodology. The Waterfall Model is a sequential and linear approach, where each phase must be completed and reviewed before the next phase can begin. This methodology is characterized by distinct stages, each with specific objectives and deliverables, ensuring a structured progression through the project lifecycle.

The phases of the Waterfall Model, as applied to this project, are detailed below:

Requirement Analysis

This initial phase involved a comprehensive understanding of the client's needs and the market demands for an online clothing store. Activities included gathering functional requirements (e.g., user registration, product browsing, shopping cart, checkout, admin functionalities) and non-functional requirements (e.g., performance, security, usability). User personas and use case diagrams were developed to capture the system's intended behavior and user interactions.

Design

Following a clear understanding of the requirements, the design phase focused on creating a detailed blueprint for the Clothing Website. This encompassed:

- **Database Design:** Structuring the MongoDB schema for products, users, and orders.
- **UI/UX Design:** Wireframing, mockups, prototypes, and visual design using tools like Figma to ensure an intuitive and aesthetically pleasing interface.
- **System Architecture Design:** Defining the interaction between the React frontend, Node.js/Express.js backend, and MongoDB database. API designs were also documented.

Implementation

This is the core development phase where the designs were translated into functional code.

- **Frontend Development:** Focused on building responsive user interfaces and components using React.js and JavaScript, integrating with the backend APIs.
- **Backend Development:** Involved setting up the Node.js server with Express.js, creating API endpoints for various functionalities (e.g., authentication, product management, order processing), and implementing business logic.
- **Database Setup:** Configuring and populating the MongoDB database.

Testing

After the implementation, a rigorous testing phase was conducted to identify and rectify any defects. This included:

- **Unit Testing:** Testing individual functions and components of both frontend and backend.
- **Integration Testing:** Verifying the seamless interaction between different modules, especially frontend-backend API communication.
- **System Testing:** Comprehensive end-to-end testing of the entire application's functionality.
- **User Acceptance Testing (UAT):** Involving stakeholders to validate if the system meets the defined business requirements.

Deployment

Once the system was thoroughly tested and deemed stable, it was prepared for deployment. This phase involved setting up the server environment, deploying the frontend and backend applications, configuring the database, and ensuring the website is accessible to users.

Maintenance

Post-deployment, the project enters a maintenance phase. This includes ongoing bug fixes, performance monitoring, security updates, and potential future enhancements based on user feedback or evolving business needs.

The Waterfall Model was chosen for this project due to its well-defined stages and clear documentation requirements, which are beneficial for projects with well-understood requirements and a need for a structured progression. Its linear nature helps in managing dependencies and ensures that each phase is thoroughly completed before moving to the next, providing a robust framework for delivering a stable and reliable e-commerce solution.

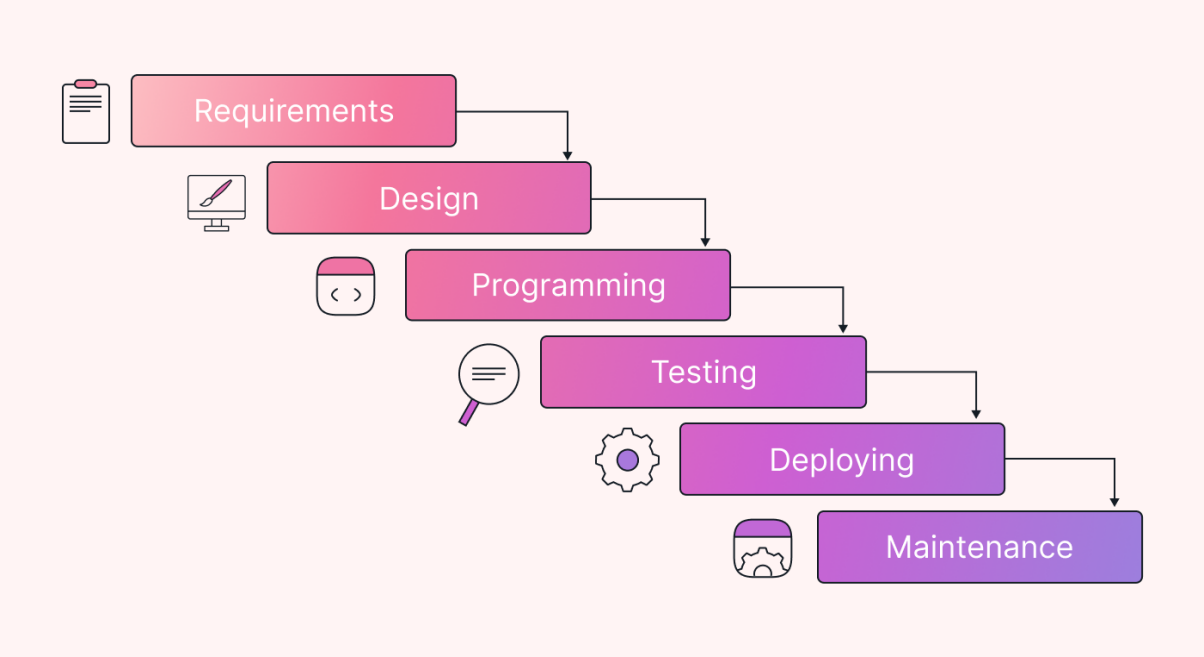


Figure 4.1: Waterfall Model used in the Clothing Website Project

Chapter 5

Body of the Project

The **Clothing Website: Design & Development** project aimed to build a dynamic, responsive, and secure e-commerce platform dedicated to selling clothing items online. This chapter outlines the work carried out throughout the internship, including requirement analysis, system analysis, design, implementation, and testing. The development was approached using full-stack technologies — specifically the MERN stack — allowing seamless interaction between the client-side and server-side components.

5.1 Work Description

The project was developed from scratch, starting with idea conceptualization to delivering a functional prototype. I was responsible for all stages of development, including UI/UX design, frontend and backend development, database integration, and administrative functionalities.

The frontend was built using React.js to provide a modern user interface with dynamic routing and reusable components. TailwindCSS was used to style the website for a clean and responsive design. The backend was developed using Node.js and Express.js, handling business logic, user authentication, and secure API endpoints. MongoDB was used for storing user data, product information, and orders. Throughout the project, version control was maintained via GitHub, and API testing was performed using Postman.

5.2 Requirement Analysis

The project was initiated with detailed requirement analysis to understand user expectations and technical needs.

Functional Requirements & Non-functional Requirements:

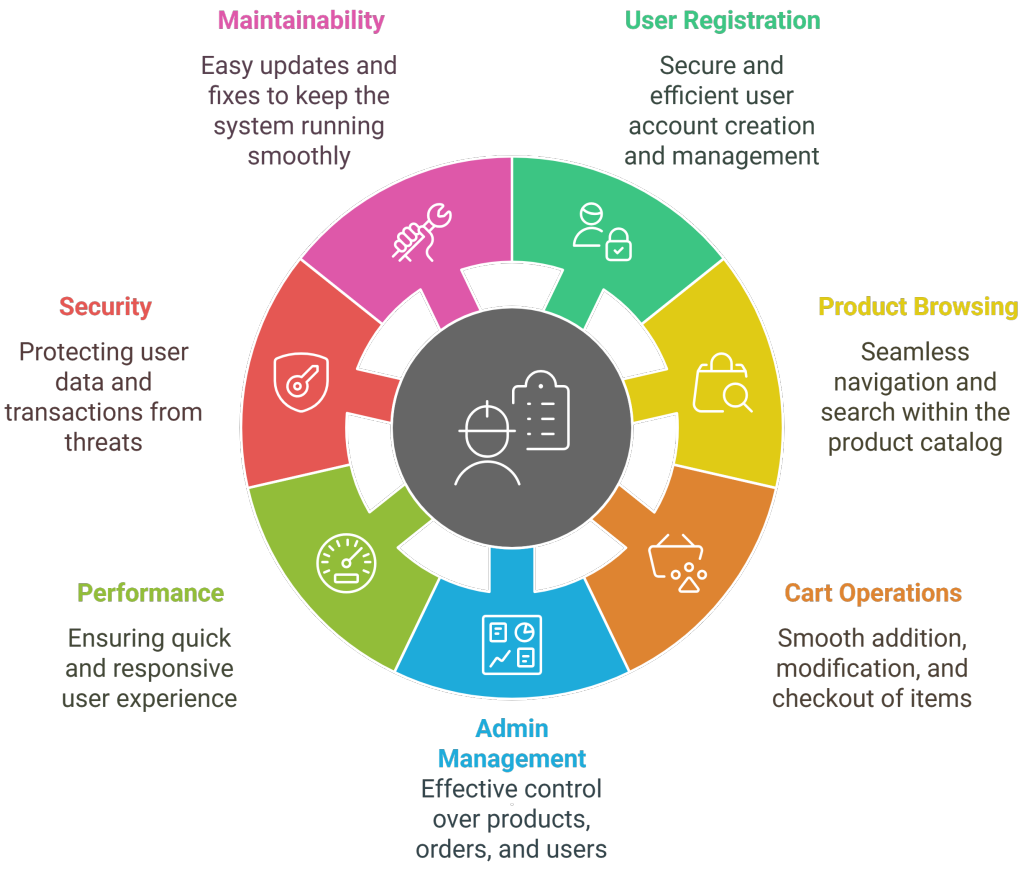


Figure 5.1: Functional and Non-functional Requirements Overview

A rich picture was created to visually express stakeholders and interactions. Requirements were categorized into user stories and acceptance criteria to guide development.

5.3 System Analysis

5.3.1 Six Element Analysis

- **People:** End-users (customers), admin users, developers.
- **Process:** Browsing, cart operations, order placements, admin management.
- **Product:** The web-based clothing store application.

- **Data:** Users, product details, orders, authentication tokens.
- **Tools:** React.js, Node.js, MongoDB, Git, Postman.
- **Environment:** Web browsers, local server, and code editor (VS Code).

5.3.2 Feasibility Analysis

- **Technical Feasibility:** Feasible using open-source, modern technologies.
- **Operational Feasibility:** The system serves real business needs.
- **Economic Feasibility:** No major cost as development tools were free.
- **Schedule Feasibility:** The project was planned and completed within a 12-week internship period.

5.3.3 Problem-Solution Analysis

The main problem addressed was the limited reach of traditional clothing stores. Many small retailers do not have an online presence. This project solves that by offering a low-cost, scalable, and customizable e-commerce solution to digitize their operations and reach a broader audience.

5.3.4 Effect and Constraints Analysis

Positive Effects:

- Increases business reach.
- Automates inventory and order management.

Constraints:

- Limited server capacity for deployment.
- No real-time payment gateway integration due to internship scope.

5.4 System Design

System design plays a critical role in transforming the project's requirements into a technical blueprint that guides development. In this project, both conceptual and technical designs

were developed to ensure clarity, consistency, and scalability. The system design included user interaction mapping through a rich picture, the definition of functional and non-functional requirements, and a high-level architectural approach to building a modular and maintainable web application.

5.4.1 Rich Picture

A rich picture was conceptualized to visually represent the stakeholders, processes, data flow, and system interactions involved in the Clothing Website project. This informal diagram outlined how users (customers and admins) interact with the system and how data flows between the frontend, backend, and database layers. It depicted the major components such as the customer interface (for product browsing and ordering), the admin panel (for managing products and orders), the authentication system (for login/signup), and the MongoDB database (storing users, products, and orders). This rich picture helped clarify the broader context of the system, including internal and external users, and served as an initial design artifact before proceeding to formal modeling like use case diagrams and ER models.

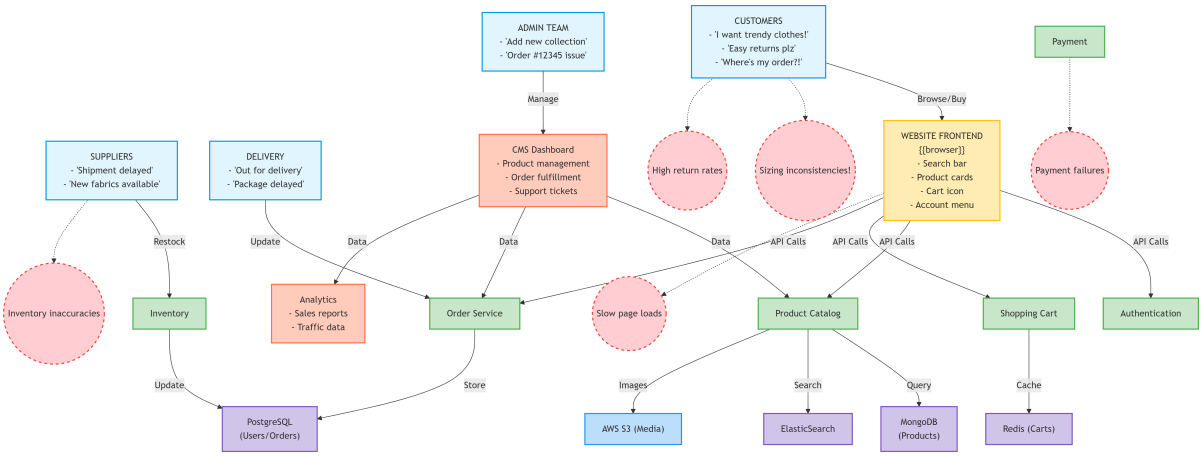


Figure 5.2: Rich Picture diagram illustrating user-system interaction

5.4.2 Use Case Diagram

The Use Case Diagram defines how various actors interact with the system. Two primary actors were identified: **Customer** and **Admin**.

The Customer can register, log in, browse products, search/filter products, manage the shopping cart, and place orders.

The Admin can log in, add/edit/delete products, and view/manage customer orders.

This diagram helped in understanding all high-level functionalities from a user's perspective

and served as the foundation for defining system requirements and permissions.

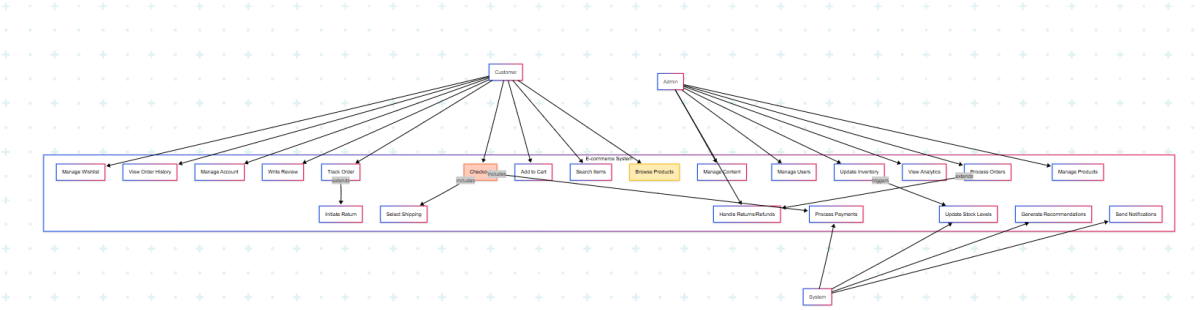


Figure 5.3: Use Case Diagram for the Clothing Website

5.4.3 Class Diagram

The Class Diagram models the static structure of the system, identifying key objects, their attributes, methods, and relationships. The main classes in the system include:

- **User:**
  - *Attributes:* id, name, email, password, role
  - *Methods:* register(), login(), updateProfile()
- **Product:**
  - *Attributes:* id, title, description, price, category, stock, imageURL
  - *Methods:* addProduct(), editProduct(), deleteProduct()
- **Order:**
  - *Attributes:* orderId, userId, products[], status, totalPrice, timestamp
  - *Methods:* placeOrder(), updateStatus(), getOrdersByUser()

These classes represent how entities are modeled in the MongoDB collections and how their behavior is handled in the backend logic.

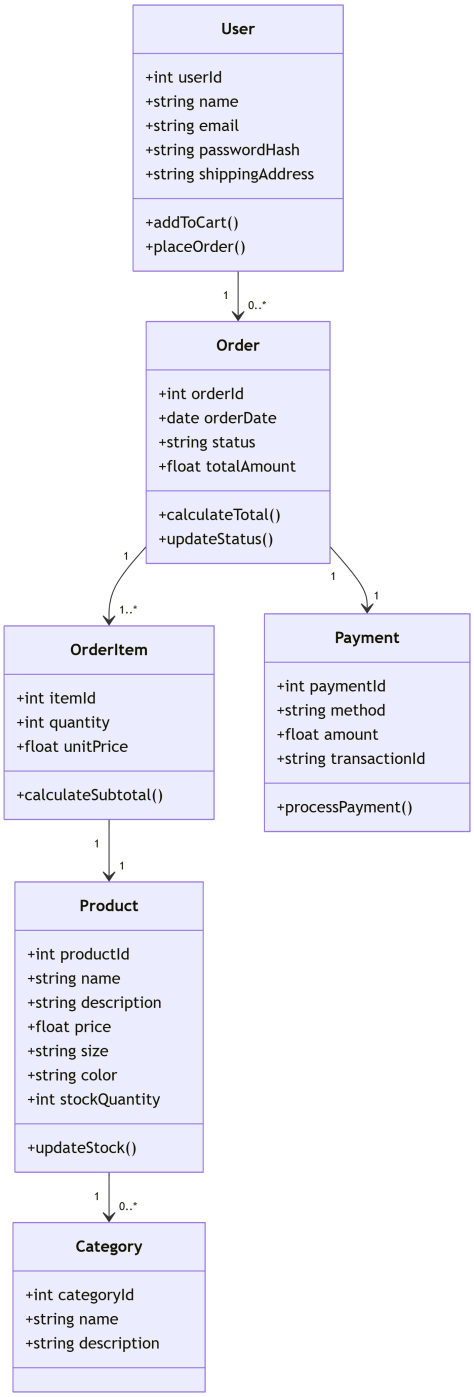


Figure 5.4: Class Diagram for the Clothing Website System

5.4.4 Sequence Diagram

The Sequence Diagram was used to visualize the interaction between frontend and backend components during specific operations.

For example, the “Place Order” flow illustrates:

1. User clicks “Checkout” in the frontend.
2. Frontend sends order data to backend API.
3. Backend validates the order and stores it in MongoDB.
4. Confirmation response is sent back to the frontend.
5. Frontend displays “Order Successful” message.

This diagram clarifies the timing and order of messages passed between system components and helps detect possible bottlenecks or errors in logic.

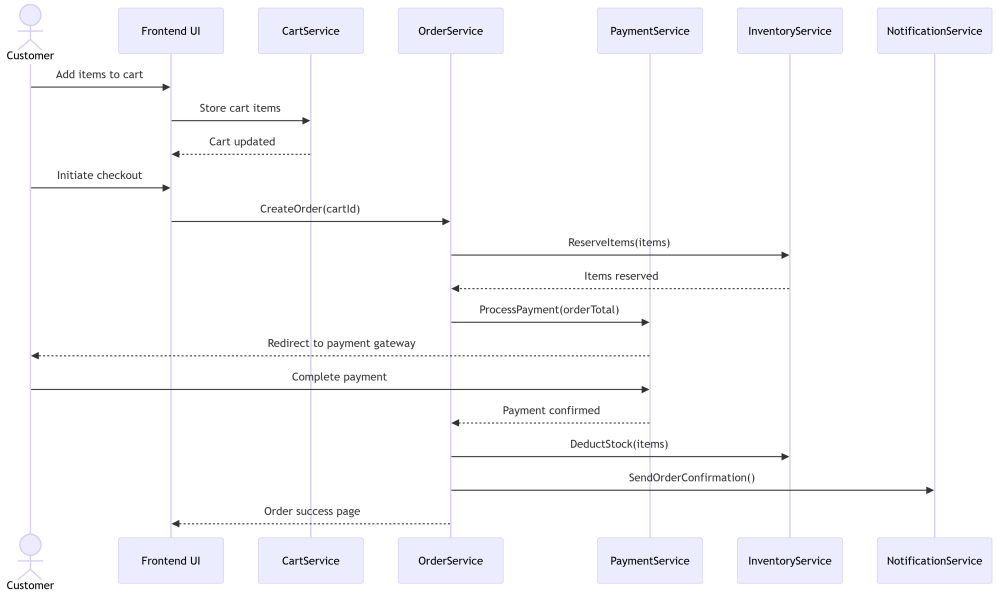


Figure 5.5: Sequence Diagram – Place Order Flow

5.4.5 Functional Requirements

The functional requirements define what the system must do to fulfill its purpose. These requirements were derived from the needs of end-users (customers) and administrative users (store managers). The primary functional requirements of the Clothing Website system include:

- **User Registration and Authentication:** The system should allow users to register, log in, and log out securely.
- **Product Browsing:** Users should be able to view a list of available clothing products, including filtering by category, price, and search keywords.
- **Shopping Cart Management:** Users should be able to add items to their cart, update item quantities, and remove items before checkout.

- **Checkout Process:** The system should support a multi-step checkout process, allowing users to confirm their order and shipping details.
- **Order Management:** Customers should be able to view their order history, and admins should be able to view, update, or fulfill orders.
- **Admin Dashboard:** Admin users should be able to add, update, and delete products, as well as manage customer orders.
- **Profile Management:** Users should be able to view and edit their profile information, including name, email, and password.
- **Access Control:** The system should distinguish between normal users and administrators through role-based access permissions.

5.4.6 Non-Functional Requirements

Non-functional requirements define how the system should perform and support usability, scalability, security, and performance. These aspects ensure the software operates reliably under real-world conditions. The key non-functional requirements of the project include:

- **Usability:** The interface must be intuitive and user-friendly, with a clean layout, readable fonts, and accessible navigation across devices.
- **Responsiveness:** The system must work seamlessly on different screen sizes including desktops, tablets, and mobile devices.
- **Performance:** The website should load pages within 2 seconds and respond to user actions (e.g., adding to cart) without noticeable delay.
- **Scalability:** The architecture must allow the addition of more users, products, and features without major refactoring.
- **Security:** Passwords must be stored securely using hashing (bcrypt), and user sessions must be authenticated using JWT tokens.
- **Maintainability:** The codebase should follow best practices such as modularity, clear naming conventions, and documentation to support future updates.
- **Data Integrity:** The system should ensure that user inputs are validated, and database transactions are correctly handled to avoid corruption or duplication.
- **Availability:** The system must aim for high uptime once deployed, with basic fault tolerance measures in place.

Together, these non-functional requirements contributed to building a system that is not only functional but also reliable and ready for real-world deployment.

5.5 Implementation

Implementation followed a modular and component-based approach.

Frontend:

- React functional components with hooks (`useState`, `useEffect`).
- Pages: Home, Login, Signup, Product Details, Cart, Admin Dashboard.
- React Router for navigation.

Backend:

- Express.js API routes for user authentication, product management, and order processing.
- MongoDB for data storage.
- Mongoose for schema validation.
- JWT-based secure authentication and middleware.
- Asynchronous API integration for cart, checkout, and admin actions.

5.5.1 Development

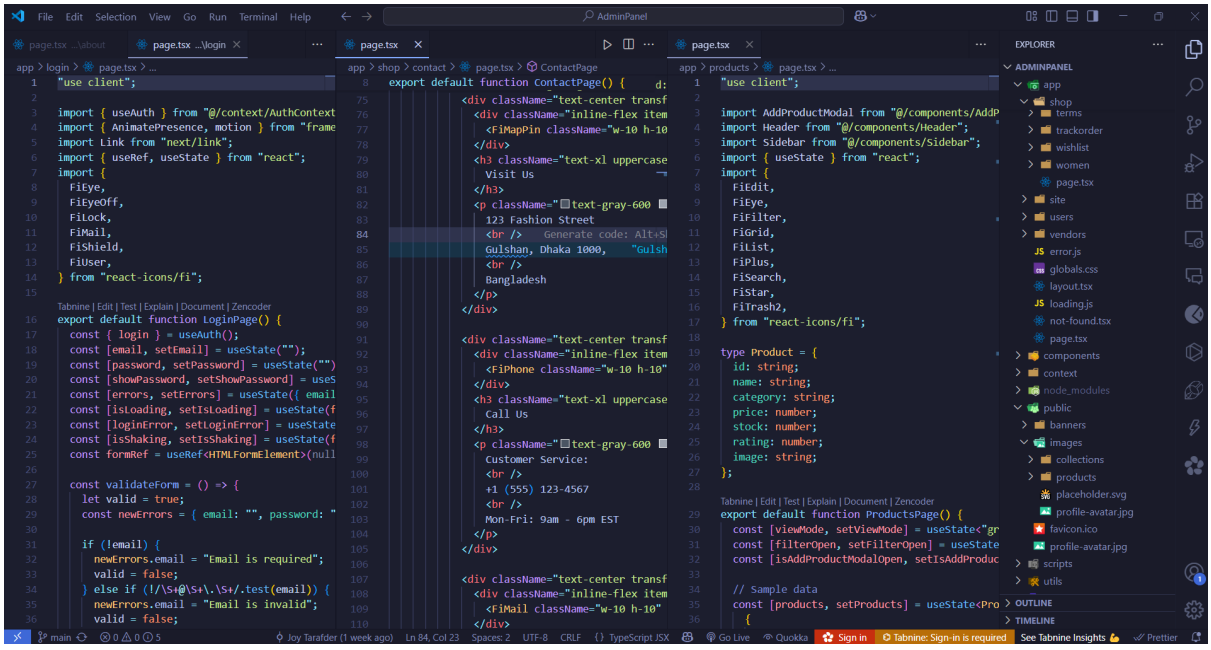


Figure 5.6: User Interface Design for ClothingCo Web App

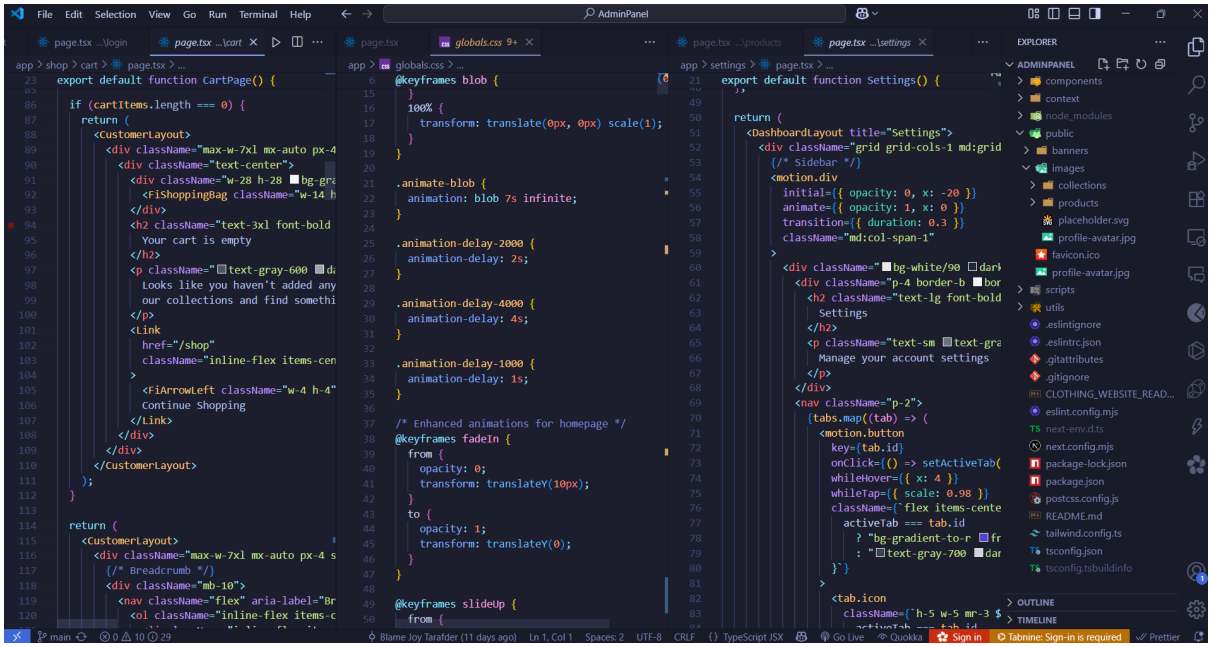


Figure 5.7: Database Schema for Product Management

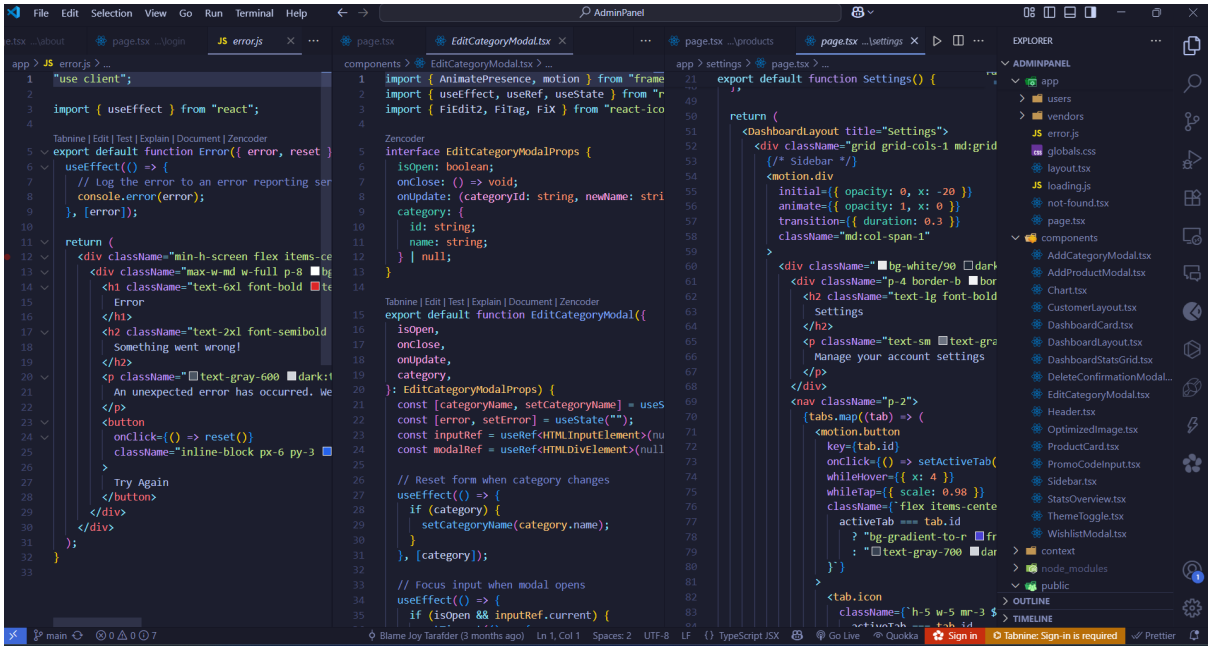


Figure 5.8: Shopping Cart Functionality Implementation

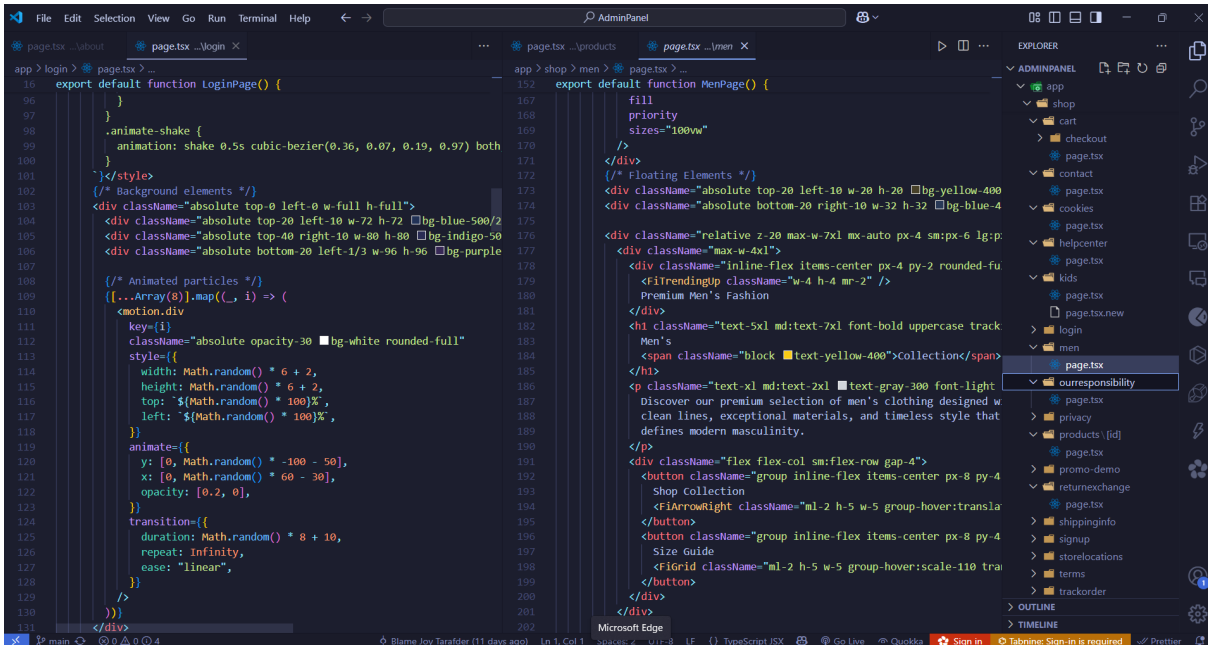


Figure 5.9: Development stages and components of the ClothingCo Internship Project

GitHub Project Link:
https://github.com/JoyTarafder/CothingCo*InternshipProject.git*

Caption: Internship Project Repository for ClothingCo

5.6 Testing

Testing played a crucial role in verifying the correctness, reliability, and robustness of the Clothing Website. A structured approach was adopted to identify and fix errors at both the frontend and backend levels. Several types of testing were conducted, including unit testing, integration testing, system testing, and user acceptance testing (UAT). Postman was used extensively for backend API testing, while manual and functional testing was applied to the frontend UI interactions. This section details the types of input and output processed during testing, the design of test cases, and the results obtained.

Input

Inputs were derived from user interactions with the system through various forms and user actions. These included:

- Registration form inputs: name, email, password, etc.

- Login credentials: email and password.
- Product filters: category, price, and search keyword.
- Cart modifications: product selections, quantity updates, and removal of items.
- Order placements: selected products and shipping details.
- Admin inputs: new product data (name, price, description, images), order status updates.

All input fields were tested for valid and invalid entries to ensure data integrity and proper error handling.

Output

Outputs consisted of system responses to user actions and API requests. Key outputs included:

- Successful or failed login attempts.
- Display of filtered product listings based on search input.
- Real-time cart updates after adding or removing items.
- Order confirmation pages displaying order summary.
- Admin notifications after product creation, update, or deletion.
- Error messages for invalid credentials, empty cart, or unauthorized access.

These outputs were visually validated in the browser and programmatically confirmed through HTTP response codes and JSON payloads during API testing.

Designing Test Cases

Test cases were designed to cover the major functional areas of the system. Each test case included a scenario, input condition, expected output, and actual result. The goal was to simulate real-world user behavior and identify possible edge cases. Test cases were created for both normal and abnormal conditions such as:

- User attempts to register with an already-used email.
- Product search with an empty query.
- Adding out-of-stock items to the cart.

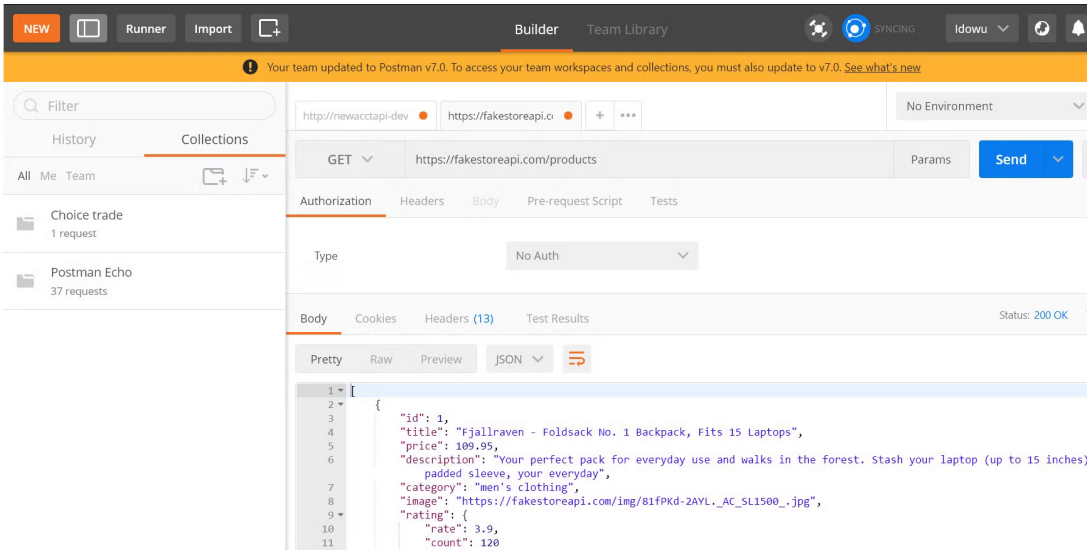


Figure 5.10: API Testing Interface

- Trying to access admin routes as a non-admin user.
- Handling incorrect product ID in the order process.

Each scenario was documented and executed step-by-step to ensure comprehensive coverage of the system functionalities.

Test Results

The testing process produced positive results. Most of the functionalities worked as intended after the first round of testing, while a few minor bugs were identified and promptly resolved. The table below summarizes a few selected test cases and their outcomes:

Table 5.1: System Test Cases with Input, Expected and Actual Results

Test Case Description	Input	Expected Result	Actual Result	Status
Register a new user with valid details	Name, Email, Password	User is registered and redirected to login	Success	Passed
Attempt login with wrong password	Email, Wrong Password	Error message: "Invalid credentials"	Success	Passed
Add product to cart	Click "Add to Cart"	Product added to cart, quantity updated	Success	Passed
Search for product by keyword	Search: "t-shirt"	List of matching products displayed	Success	Passed

Continued on next page

Test Case Description	Input	Expected Result	Actual Result	Status
Place order with empty cart	Click “Checkout”	Error message: “Cart is empty”	Success	Passed
Admin adds a new product	Product form input	Product appears in product list	Success	Passed
Non-admin accessing admin dashboard	Unauthorized token	Error message: “Access denied”	Success	Passed

All test cases passed as expected, and bugs discovered during testing were resolved before the final review.

Chapter 6

Results & Analysis

The Clothing Website: Design & Development project produced a fully functional and interactive e-commerce platform tailored for online clothing retail. The outcome reflects a successful implementation of a full-stack web application using modern web technologies. The project was developed in line with its objectives — ensuring a smooth user experience, providing administrative control, maintaining data integrity, and enhancing user accessibility through responsive design. This chapter discusses the key results achieved and provides an in-depth analysis of their effectiveness and relevance.

6.1 Frontend Results

The frontend of the application was successfully developed using React.js. Key UI components such as the product listing page, product details view, shopping cart, checkout page, and user profile were implemented and functioned as expected. The design was kept minimal and clean using TailwindCSS, which ensured mobile responsiveness and a consistent visual experience across different devices. The navigation between pages was seamless due to React Router, and state management was handled using hooks and context APIs. As a result, users could register, log in, browse clothing products, add them to the cart, and proceed to checkout without reloading the page — achieving a Single Page Application (SPA) experience.

6.2 Backend Results

The backend was built using Node.js and Express.js, and it handled various server-side functionalities including authentication, product management, and order processing. API endpoints were created and tested successfully using Postman, confirming that all core functionalities such

as user creation, login, fetching product data, managing the shopping cart, and order placement worked reliably. The system was designed to differentiate between regular users and admin users using role-based access control (RBAC), ensuring that only authorized users could access administrative features.

6.3 Database Integration

The MongoDB database, accessed using Mongoose, effectively stored and retrieved data in real-time. Collections were created for users, products, and orders. Each order was linked to a user and included relevant details such as item list, price, date, and order status. Product documents stored information such as name, description, price, category, stock, and images. All operations such as adding new products, retrieving user profiles, and viewing orders were successfully performed through properly structured database queries.

6.4 Authentication and Security

User authentication was implemented using JWT (JSON Web Tokens), and password protection was ensured with bcrypt.js hashing. The login system worked as intended, rejecting unauthorized access and preventing sensitive endpoints from being accessed by non-authenticated users. Input validation was applied across forms, and error-handling middleware ensured that users received proper feedback for invalid actions.

6.5 Admin Dashboard Results

The admin dashboard allowed product CRUD (Create, Read, Update, Delete) operations and order management. Admins could add new products with necessary details, update existing product information, and mark orders as fulfilled. This feature demonstrated the project's ability to support internal business operations in addition to external customer engagement.

6.6 System Performance Analysis

Performance was tested on local servers using both simulated user flows and developer tools. The average page load time remained below 2 seconds across key components. Efficient use of asynchronous operations (via Axios and Express middleware) prevented server blocking and ensured responsiveness. No major memory leaks or server crashes were detected during testing.

6.7 User Experience and Feedback

Although formal user testing with external users was not conducted, simulated test scenarios were used to analyze the overall user journey. Navigation was intuitive, interactions were smooth, and feedback messages were clearly displayed. Forms had client-side validation, error messages appeared appropriately, and users were guided through each step (e.g., checkout, login).

6.8 Limitations Identified

While the platform met its primary goals, some limitations were observed:

- The payment gateway was conceptually integrated but not live.
- Deployment was done on a local server; real-world hosting (e.g., Vercel, MongoDB Atlas) was not completed due to internship scope.
- Advanced analytics and reporting (e.g., sales graphs, revenue dashboards) were not included in this version.

6.9 Overall Analysis

The results of this project validate the feasibility and efficiency of building an end-to-end e-commerce solution using the MERN stack. Each component of the application — from the frontend UI to backend APIs and database models — worked cohesively to support a fully operational online clothing store. The hands-on implementation not only fulfilled the initial project objectives but also significantly enhanced my technical proficiency, especially in areas of API design, authentication, and full-stack development practices.

The outcome is a professional-grade prototype that could be extended further into a production-level product with the addition of live payment systems, search engine optimization, and real-time analytics.

Chapter 7

Project as Engineering Problem Analysis

The *Clothing Website: Design & Development* project was not just a software development exercise but also a real-world engineering solution to a common business problem. It tackled the challenges faced by small to medium-sized clothing retailers in establishing an effective online presence. This chapter evaluates the project through the lens of engineering problem analysis, focusing on sustainability, social and environmental impacts, and ethical considerations.

7.1 Sustainability of the Project/Work

Sustainability is a critical aspect of any engineering solution, particularly in the field of software development. This project was designed with sustainability in mind through the use of scalable technologies, efficient coding practices, and modular architecture. The use of open-source tools such as React, Node.js, and MongoDB ensures that the platform can be maintained and updated without recurring license costs. Additionally, the modular component-based architecture supports long-term maintainability, allowing future developers or teams to easily enhance or modify features without disrupting the existing system.

From a business perspective, the application supports digital transformation for local retailers, enabling them to reach broader markets with minimal resource consumption compared to traditional storefronts. The system can also be hosted on cloud platforms, which support auto-scaling and resource optimization, further promoting energy efficiency and cost control.

7.2 Social and Environmental Effects and Analysis

The project contributes positively to social development by promoting the digital inclusion of small clothing businesses. In regions where traditional retail is dominant, transitioning to e-commerce empowers local entrepreneurs to compete in the global marketplace. It also makes clothing more accessible to consumers who may not have access to physical stores, including those in rural or remote areas.

Environmentally, the platform indirectly contributes to sustainability by reducing the need for large-scale physical infrastructure, transportation, and paper-based inventory or billing systems. E-commerce platforms typically produce a smaller carbon footprint per transaction compared to physical retail stores, especially when implemented using eco-friendly cloud services.

However, it is important to recognize that online retail can also contribute to environmental burdens through packaging waste and delivery logistics. While these factors are outside the scope of this project, future integrations — such as eco-friendly packaging options or carbon-offset shipping — can further align the platform with green technology principles.

7.3 Addressing Ethics and Ethical Issues

Ethical responsibility is a fundamental component of engineering practice. In this project, several ethical principles were considered and implemented. Firstly, user data privacy was treated with utmost importance. Passwords were stored using hashing algorithms (bcrypt), and user sessions were managed using secure JWT tokens to prevent unauthorized access. Although no financial data was stored, all conceptual payment features were designed to comply with basic security standards.

Transparency and honesty were maintained in all aspects of the project, including feature limitations. Users were clearly informed of actions such as login errors, product availability, and order status. Role-based access control ensured that administrative features could only be accessed by verified users, protecting the system from misuse.

Additionally, efforts were made to ensure inclusivity by designing a responsive interface that could be used across devices and by users with varying levels of technical proficiency. This supports the broader ethical goal of making technology accessible and usable for all.

Going forward, the platform could benefit from further ethical considerations such as accessibility for disabled users (e.g., screen-reader support) and compliance with international privacy regulations like the General Data Protection Regulation (GDPR).

Chapter 8

Lesson Learned

The completion of the *Clothing Website: Design & Development* project provided me with valuable real-world experience that extended far beyond the theoretical knowledge gained in the classroom. Over the course of the internship, I encountered several technical and professional challenges, each of which contributed to my personal and academic growth. This chapter outlines the key problems faced during the project and how they were resolved.

8.1 Problems Faced During this Period

One of the initial challenges I encountered was the lack of clarity in feature prioritization. Since the project aimed to replicate a real-world e-commerce platform, it was tempting to include a wide range of features such as wishlists, live chat, coupon systems, and full payment integration. However, with limited time and resources, I needed to focus on the core functionalities first. Learning to identify what was essential versus what was desirable was a key lesson in project scoping and time management.

Another significant challenge was in backend API design and database structuring, especially considering the asynchronous nature of Node.js and the flexibility of MongoDB. Initially, I faced difficulties in designing efficient data models and ensuring smooth communication between the frontend and backend. Inconsistencies in API responses and unstructured data returned from MongoDB led to bugs and user interface issues.

Security was also a major area of concern. Implementing user authentication and protecting routes was more complex than expected. It took multiple iterations to get JSON Web Token (JWT) implementation working securely and consistently. Additionally, managing role-based access control for admin features added an extra layer of complexity to the system.

On the frontend, building a fully responsive design with TailwindCSS took significant effort

and repeated testing across different devices. Handling state management in React using the Context API was also initially confusing, especially when coordinating between components like the cart, product list, and checkout page.

Lastly, time constraints toward the end of the internship made it difficult to explore optional features such as payment gateway integration or deployment to live hosting platforms like Vercel or MongoDB Atlas.

8.2 Solution of Those Problems

To address the issue of unclear priorities, I created a feature prioritization matrix, classifying tasks as "Must Have," "Should Have," "Could Have," and "Won't Have" for this version. This approach helped me stay focused on delivering a Minimum Viable Product (MVP) within the internship period.

For the backend API challenges, I studied REST principles more thoroughly and used tools like Postman to manually test API endpoints. I revised my data models using Mongoose schemas to enforce structure and relationships between collections. I also modularized the codebase to improve clarity and reusability.

In terms of security, I followed best practices by hashing passwords with `bcrypt.js` and storing tokens securely. I implemented middleware for token validation and applied conditional logic to restrict admin-only routes. Online tutorials, documentation, and community forums played a significant role in helping me solve these security-related issues.

To improve responsiveness on the frontend, I used mobile-first design principles and the grid system in TailwindCSS. I conducted manual testing on different screen sizes and devices. For managing state in React, I used the Context API for global state (e.g., cart data) and `useState` for local UI behavior. Debugging tools such as React Developer Tools helped in visualizing and tracking state updates.

Time management was improved by setting weekly goals and using simple tools like Trello to track task progress. While payment integration and full deployment were deferred, the architecture was intentionally left modular to allow for future expansion.

Chapter 9

Future Work & Conclusion

The development of the *Clothing Website: Design & Development* project marked the successful completion of a real-world, full-stack web application under the scope of my internship at Cloud Coder. While the core features of the platform were fully implemented and tested, there remains significant potential to extend and enhance the application in future iterations. This chapter outlines the scope for future work and concludes the project with a summary of its achievements and impact.

9.1 Future Works

Although the current version of the website fulfills essential e-commerce functionalities such as user registration, product browsing, cart management, and order placement, several advanced features can be introduced to improve user experience, administrative control, and business scalability.

One of the most critical enhancements would be the integration of a live payment gateway such as Stripe or SSLCommerz to allow real-time transactions. Currently, the checkout process is conceptual, lacking live payment functionality. Implementing this would make the platform production-ready.

Another area of development is inventory and stock management, including automatic alerts when stock runs low and detailed tracking of sales and returns. Introducing email or SMS notifications for order updates would enhance customer engagement and communication.

To provide better insights for store owners, analytics and reporting dashboards can be added. These would show data visualizations of sales trends, user behavior, most viewed or purchased items, and revenue generation over time.

From a user experience perspective, future improvements may include wishlist functionality, user reviews and ratings, coupon or discount systems, and multi-language support to cater to a broader customer base.

In terms of performance and reliability, deploying the application on cloud platforms like Vercel (frontend) and MongoDB Atlas (backend database) would make the platform accessible to real users and support scalability. Additional improvements could include unit test automation, end-to-end testing with tools like Cypress, and integration with CI/CD pipelines for production readiness.

Accessibility improvements such as screen reader support, keyboard navigation, and adherence to WCAG (Web Content Accessibility Guidelines) would ensure that the platform is usable by people with disabilities, aligning the project with inclusive design principles.

9.2 Conclusion

The *Clothing Website* project has been a significant milestone in my academic and professional development. Through the successful design and development of a full-stack e-commerce platform, I was able to apply theoretical concepts learned throughout my undergraduate coursework to a practical and impactful project. From designing intuitive UI components to managing backend APIs and securing user data, every step of the process contributed to a deeper understanding of modern web development practices.

The project has addressed real-world challenges faced by small clothing retailers who lack the resources to build their own digital presence. By offering a scalable, user-friendly, and secure online shopping solution, this platform has the potential to contribute positively to business growth and customer accessibility.

Overall, this internship has helped me grow not just as a developer, but also as a problem-solver, team collaborator, and critical thinker. It has provided a clear picture of professional software engineering workflows and inspired confidence to take on more complex projects in the future.

Bibliography

[1] Journal of Integrated Science and Technology, "Harnessing the mern stack for scalable e-commerce website design: A full-stack approach with mongodb, node.js, express.js, and react.js," *Journal of Integrated Science and Technology*, vol. 13, no. 5, p. 1116, 2025.

[2] International Journal of Research Publication and Review, "Maximizing efficiency: Unveiling the advantages of e-commerce apps for online operations," *International Journal of Research Publication and Review*, vol. 5, no. 3, 2024.

[3] ResearchGate, "(pdf) harnessing the mern stack for scalable e-commerce website design: A full-stack approach with mongodb, node.js, express.js, and react.js." `https://www.researchgate.net/publication/390647948_Harnessing_the_MERN_stack_for_scalable_E-commerce_website_design_A_full-_stack_approach_with_MongoDB_Nodejs_Expressjs_and_Reactjs`, 2025.

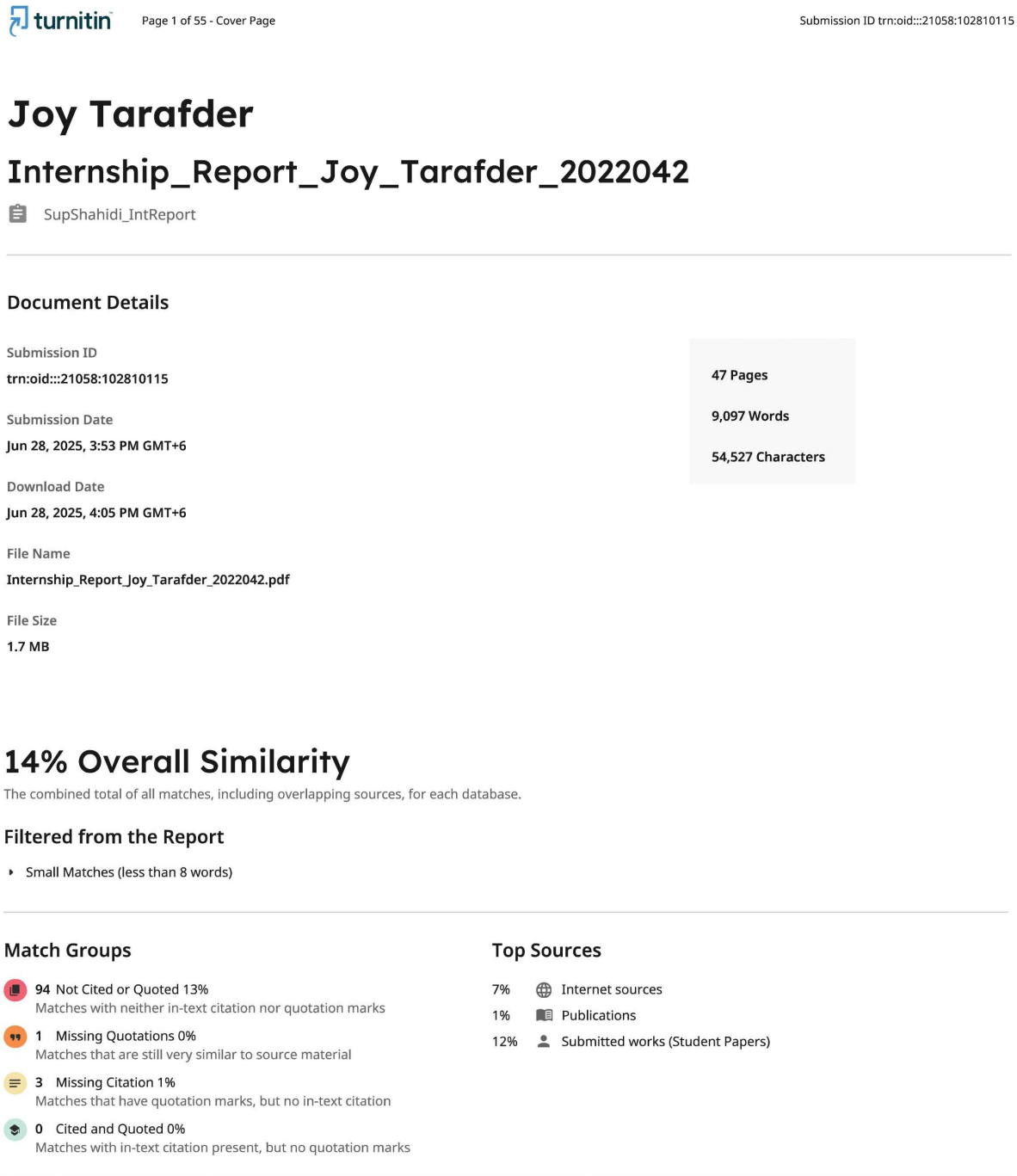
[4] ResearchGate, "(pdf) e-commerce: Advantages and limitations." `https://www.researchgate.net/publication/354062933_E-Commerce_Advantages_and_Limitations`, 2021.

[5] R. S. Pressman, *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 8 ed., 2014.

[6] I. Sommerville, *Software Engineering*. Pearson Education, 10 ed., 2015.

[7] ISO/IEC/IEEE 12207:2017, "Systems and software engineering – software life cycle processes," 2017. International Standard.

[1] [2] [3] [4] [5] [6] [7]


turnitin Page 1 of 55 - Cover Page Submission ID trn:oid:::21058:102810115

Joy Tarafder

Internship_Report_Joy_Tarafder_2022042

SupShahidi_IntReport

Document Details

Submission ID
trn:oid:::21058:102810115

Submission Date
Jun 28, 2025, 3:53 PM GMT+6

Download Date
Jun 28, 2025, 4:05 PM GMT+6

File Name
Internship_Report_Joy_Tarafder_2022042.pdf

File Size
1.7 MB

47 Pages

9,097 Words

54,527 Characters

14% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

Small Matches (less than 8 words)

Match Groups

94 Not Cited or Quoted 13%
Matches with neither in-text citation nor quotation marks

1 Missing Quotations 0%
Matches that are still very similar to source material

3 Missing Citation 1%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

7% Internet sources
1% Publications
12% Submitted works (Student Papers)

Figure 9.1: Figure: Plagiarism Check Result

An Undergraduate Internship/Project on Clothing Website: Design & Development

By

Joy Tarafder

Student ID: **2022042**

Spring, 2025

The student modified the internship final report as per the recommendation made by his or her academic supervisor and/or panel members during final viva, and the department can use this version for achieving.

Signature of the Supervisor

Sarwar Shahidi
Research & Development Officer
Department of Computer Science & Engineering
School of Engineering, Technology & Sciences
Independent University, Bangladesh