

Bag Emporium

Online Bag Shop

Mini Project Report

Submitted by

Pranav P K

Reg. No.: AJC22MCA-2072

In Partial Fulfillment for the Award of the Degree of

MASTER OF COMPUTER APPLICATIONS

(MCA TWO YEAR)

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY



AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2023-2024

DEPARTMENT OF COMPUTER APPLICATIONS
AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY



CERTIFICATE

This is to certify that the Project report, “**Bag Emporium**” is the bonafide work of **PRANAV P K (Regno: AJC22MCA-2072)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Mr. Rony Tom
Internal Guide

Ms. Meera Rose Mathew
Coordinator

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report “**Bag Emporium**” is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

Date:

PRANAV P K

KANJIRAPPALLY

Reg: AJC22MCA-2072

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Mr. Rony Tom** for his inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

PRANAV P K

ABSTRACT

The "Bag Emporium" is an encompassing e-commerce initiative dedicated to delivering a seamless and user-friendly online shopping experience for bag enthusiasts. Designed to be intuitive and visually engaging, the platform allows customers to effortlessly explore, select, and purchase a diverse array of bags, including handbags, backpacks, and travel bags. Admin features encompass a comprehensive dashboard, user status management, seller registration approval, product and customer administration, sales analytics, and database control. Customers can register accounts, browse products, manage shopping carts and wishlists, proceed through secure checkout, and access order history. Sellers have functionalities to log in, update product listings and stocks, and view customer feedback. For sellers, the system streamlines their experience by providing a secure login with seller credentials. Sellers can effortlessly update and add new products, manage product stocks, and view customer feedback for continuous improvement. This structured approach ensures a dynamic and collaborative e-commerce ecosystem, benefiting customers, administrators, and sellers alike.

"Bag Emporium" an all-encompassing e-commerce initiative, redefines online shopping for bag enthusiasts. Seamlessly blending intuitive design with visual appeal, the platform offers a diverse array of bags – from handbags to backpacks and travel bags. Customers enjoy a hassle-free experience with account registration, easy product exploration, secure checkout, and access to order history. Meanwhile, administrators benefit from a comprehensive dashboard, user status management, and sales analytics, fostering a dynamic e-commerce ecosystem. Sellers, equipped with secure login credentials, effortlessly manage product listings, stocks, and customer feedback, ensuring a collaborative and streamlined experience for all stakeholders.

CONTENT

SL. NO	TOPIC	PAGE NO
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	4
2.1	INTRODUCTION	5
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	6
2.4	PROPOSED SYSTEM	6
2.5	ADVANTAGES OF PROPOSED SYSTEM	7
3	REQUIREMENT ANALYSIS	8
3.1	FEASIBILITY STUDY	9
3.1.1	ECONOMICAL FEASIBILITY	9
3.1.2	TECHNICAL FEASIBILITY	9
3.1.3	BEHAVIORAL FEASIBILITY	10
3.1.4	FEASIBILITY STUDY QUESTIONNAIRE	10
3.2	SYSTEM SPECIFICATION	13
3.2.1	HARDWARE SPECIFICATION	13
3.2.2	SOFTWARE SPECIFICATION	13
3.3	SOFTWARE DESCRIPTION	13
3.3.1	DJANGO FRAMEWORK	13
3.3.2	SQLITE	13
4	SYSTEM DESIGN	15
4.1	INTRODUCTION	16
4.2	UML DIAGRAM	16
4.2.1	USE CASE DIAGRAM	16
4.2.2	SEQUENCE DIAGRAM	19
4.2.3	STATE CHART DIAGRAM	21
4.2.4	ACTIVITY DIAGRAM	22
4.2.5	CLASS DIAGRAM	24
4.2.6	OBJECT DIAGRAM	26
4.2.7	COMPONENT DIAGRAM	28

4.2.8	DEPLOYMENT DIAGRAM	30
4.2.9	COLLABORATION DIAGRAM	31
4.3	USER INTERFACE DESIGN USING FIGMA	32
4.4	DATABASE DESIGN	35
4.5	TABLE DESIGN	38
5	SYSTEM TESTING	44
5.1	INTRODUCTION	45
5.2	TEST PLAN	45
5.2.1	UNIT TESTING	46
5.2.2	INTEGRATION TESTING	46
5.2.3	VALIDATION TESTING	47
5.2.4	USER ACCEPTANCE TESTING	47
5.2.5	AUTOMATION TESTING	47
5.2.6	SELENIUM TESTING	47
6	IMPLEMENTATION	62
6.1	INTRODUCTION	63
6.2	IMPLEMENTATION PROCEDURE	63
6.2.1	USER TRAINING	64
6.2.2	TRAINING ON APPLICATION SOFTWARE	64
6.2.3	SYSTEM MAINTENANCE	64
7	CONCLUSION & FUTURE SCOPE	65
7.1	CONCLUSION	66
7.2	FUTURE SCOPE	66
8	BIBLIOGRAPHY	67
9	APPENDIX	69
9.1	SAMPLE CODE	70
9.2	SCREEN SHOTS	107

List of Abbreviation

- IDE - Integrated Development Environment
- HTML - Hyper Text Markup Language
- CSS - Cascading Style Sheet
- JS - JavaScript
- PY - Python
- 1NF - First Normal Form
- 2NF - Second Normal Form
- 3NF - Third Normal Form
- PK - Primary Key
- FK - Foreign Key

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The "Bag Emporium" online bag shop project is a sophisticated e-commerce platform designed to provide users with a seamless and visually appealing shopping experience for a diverse range of bags. This comprehensive platform offers customers the ability to register accounts, explore an extensive catalog, manage shopping carts and wishlists, securely check out, and view order history. Admin functionalities include a detailed dashboard, user and seller management, product catalog oversight, sales analytics, and database management. Sellers can log in, update product listings and stocks, and view customer feedback. With a focus on user satisfaction, efficient administration, and secure transactions, Bag Emporium aims to create a user-friendly environment for bag enthusiasts to effortlessly browse, select, and purchase their preferred products online.

1.2 PROJECT SPECIFICATION

Functionalities:

1. Customer

- Can register an account.
- Browse and search for products.
- View product details.
- Add product to the shopping cart.
- View and modify cart.
- Add product to wishlist.
- Manage wishlist items.
- Proceed to check out and make payment.
- Download payment details.
- Can add reviews and ratings.

2. Admin

- Dashboard and overview.
- Manage user status.
- View and approve seller registration.
- Customer management.
- Product management.
- Sales analytics.
- Can view and make changes in the database.

3. Seller

- Login using seller credentials.
- Can update and add new products.
- Can update the product stocks.
- Can view reviews and feedback.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

The "Bag Emporium" project presents a comprehensive and innovative e-commerce platform focused on providing customers with a seamless and user-friendly online shopping experience for a diverse range of bags. This system encompasses the entire e-commerce process, offering features such as extensive product management, customer engagement, sales analytics, and advanced functionalities like recommendation systems and personalized pricing. In a dynamic landscape related to the e-commerce industry, the project stands out with its commitment to delivering a wide variety of bags, an intuitive user interface, secure payment options, wishlist features, and efficient order management. The ownership lies with the admin module, ensuring centralized control over the system. With the involvement of customers, sellers, administrators, and delivery personnel, the project aims to cater to the unique needs and preferences of each user category. The administrator's dashboard and overview section are equipped with specific functionalities and metrics to monitor and manage the platform effectively. In essence, the "Online Bag Shop" project emerges as a distinctive and customer-centric solution in the competitive realm of online bag retail, emphasizing variety, usability, and security.

.

2.2 EXISTING SYSTEM

In the existing system, the online bag shopping platform is designed with core functionalities to cater to the needs of customers, administrators, and sellers. Customers have the ability to create accounts, browse, and search for products effortlessly, while also enjoying features such as a cart for easy order management and a wishlist for saving preferred items. The checkout process is streamlined, allowing secure payments through various methods.

2.2.1 NATURAL SYSTEM STUDIED

A natural system study for the described online bag shopping platform involves an analysis of the system's interactions and processes in relation to its environment. It explores how the platform functions as an interconnected ecosystem, with customers, administrators, and sellers acting as integral components. The study delves into the natural flow of information, transactions, and feedback, considering the user experience and the seamless integration of features. Additionally, it assesses the adaptability and responsiveness of the system to changing demands within the e-commerce industry. The natural system study seeks to identify potential synergies, dependencies, and areas for enhancement within the platform, ensuring that it aligns harmoniously with the dynamics of the online retail landscape and continues to thrive as a cohesive and efficient digital marketplace for bags.

2.2.2 DESIGNED SYSTEM STUDIED

The designed system study for the online bag shopping platform focuses on the intentional architecture and functionalities crafted to meet specific objectives. It delves into the user-centric design principles applied to enhance the customer experience, such as the seamless registration process, intuitive product browsing, and secure payment options. The study also examines the deliberate choices made in the administrator and seller interfaces to streamline management tasks, optimize order processing, and facilitate effective communication. Furthermore, it explores the strategic inclusion of features like wishlists and reviews to foster user engagement.

2.3 DRAWBACKS OF EXISTING SYSTEM

- **Limited Seller Interaction:** In the existing system, the interaction capabilities for sellers are relatively basic. While sellers can log in, update products, and view feedback, the system lacks more sophisticated tools for seller management, analytics, or direct communication channels with administrators.
- **Lack of Mobile Responsiveness:** The abstract does not explicitly mention the system's compatibility with mobile devices.
- **Incomplete Support:** The existing system might face challenges related to incomplete support, which could manifest in the form of limited compatibility with various browsers, operating systems, or devices. Incomplete support may lead to inconsistencies in user experience, causing functionalities to behave differently across different environments.
- **Accessibility Issues:** The existing system may face accessibility challenges due to the absence of information regarding mobile responsiveness. If the platform is not optimized for mobile devices, users may encounter difficulties navigating and interacting with the online bag shop on smartphones or tablets.
- **Limited Scalability:** The scalability of the existing system might be a concern, particularly if there is an absence of information on the system's ability to handle increased traffic, products, and transactions as the user base grows. A lack of scalability could result in performance issues, slower response times, or even system crashes during peak periods or business expansions.

2.4 PROPOSED SYSTEM

The proposed system for the online bag shopping platform seeks to address the identified drawbacks of the existing system while introducing enhancements to elevate the overall user experience. Notable improvements include an enriched set of analytical tools for administrators,

empowering them with in-depth insights into customer behavior, advanced product performance metrics, and detailed sales forecasts.

2.5 ADVANTAGES OF PROPOSED SYSTEM

- **Comprehensive Support:** The proposed system prioritizes comprehensive support, ensuring a well-rounded user experience by addressing potential issues related to browser compatibility, operating system variations, and device diversity. Through thorough testing and optimization, the system aims to provide consistent and reliable performance across various platforms, enhancing accessibility for users and fostering a seamless interaction irrespective of their chosen browser or device.
- **Dynamic product catalog:** The proposed system supports a dynamic product catalog, allowing for easy addition, editing, and removal of products. It accommodates a diverse range of bags, enabling the platform to adapt to changing market trends and customer preferences swiftly. This ensures that the catalog remains fresh, relevant, and engaging for users.
- **Scalability and Future-Proofing:** The proposed system is designed with scalability in mind, allowing for seamless expansion to accommodate a growing user base and evolving business needs. This ensures that the online bag shopping platform remains technologically current and adaptable to future advancements in e-commerce, providing a sustainable solution for long-term success.
- **Enhanced Communication Channels:** To foster improved communication between users and administrators, the proposed system introduces enhanced messaging and notification systems. This facilitates prompt order updates, feedback responses, and seamless collaboration between customers, sellers, and administrators, contributing to a more transparent and responsive online shopping experience.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

The "Bag Emporium" project exhibits strong feasibility, primarily driven by its well-defined objectives and robust feature set. Economically, the project aligns with market demands for an intuitive and visually appealing online shopping experience, which is critical for attracting and retaining customers in the competitive e-commerce landscape. The platform's wide range of bag categories, coupled with detailed product information, addresses customer expectations and positions the project for market success.

3.1.1 ECONOMICAL FEASIBILITY

The "Bag Emporium" project showcases strong economic feasibility through a well-structured plan that aligns investment with potential returns. The initial costs associated with platform development, system maintenance, and database management are offset by the anticipated revenue streams generated through increased sales and customer retention. The project's focus on creating a visually appealing and intuitive online shopping environment responds directly to market demands, positioning it for profitability. Additionally, the inclusion of features like sales analytics enables informed decision-making for revenue optimization. The ongoing cost of system maintenance and updates is justified by the long-term benefits of sustained customer engagement and business growth, making the project economically viable.

3.1.2 TECHNICAL FEASIBILITY

The technical feasibility of the "Bag Emporium" project is evident in its comprehensive feature set and seamless integration of technology. The platform's ability to accommodate diverse bag categories, provide detailed product information, and support functionalities for different user roles demonstrates its adaptability to market trends. The inclusion of a robust database management system ensures data accuracy and accessibility. The project's technological foundation is further strengthened by features like location addition, order tracking, and feedback mechanisms, enhancing the overall user experience. The project's capability to handle system maintenance and updates signifies its readiness to evolve with emerging technologies, ensuring long-term technical sustainability.

3.1.3 BEHAVIORAL FEASIBILITY

The project exhibits strong behavioral feasibility by aligning with user expectations and behaviors in the online shopping domain. For customers, the user-friendly interface, detailed product information, and order history tracking cater to their preferences, enhancing satisfaction and fostering loyalty. The platform encourages active user engagement through features like feedback submission, contributing to a positive user experience. Seller benefit from a simplified login process and streamlined product management capabilities, aligning with their behavioral patterns in e-commerce environments. The project's emphasis on behavioral factors positions it as a user-centric platform, likely to be embraced by both customers and sellers, ensuring successful adoption and sustained usage.

3.1.4 FEASIBILITY STUDY QUESTIONNAIRE

1. Project Overview?

The "Bag Emporium" project is an e-commerce platform designed to offer a seamless and user-friendly online shopping experience for purchasing bags. It provides a wide range of bag categories, allows customers to browse products, view product details, and make secure online payments.

2. To what extend the system is proposed for?

The proposed "Bag Emporium" system is envisioned to be a comprehensive e-commerce platform, offering an end-to-end solution for seamless and user-friendly online bag shopping. With a wide range of bag categories, detailed product information, and features tailored to different user roles including Admin, Customer, and Seller, the system aims to cover the entirety of the e-commerce process. From intuitive browsing and product selection to streamlined order management and sales analytics, the platform appears to be designed to meet the diverse needs of customers and sellers alike. The emphasis on database control, system maintenance, and continuous updates suggests a commitment to adaptability and long-term sustainability in the dynamic landscape of online bag retail.

3. Specify the Viewers/Public which is to be involved in the System?

The "Bag Emporium" involves administrators overseeing the platform, managing product listings, customer interactions, sales analytics, and database control. Customers engage in browsing, purchasing, and providing feedback, while sellers, with a dedicated login, actively update and add products to maintain a diverse inventory.

4. List the Modules included in your System?

- **Customer:**

Register, browse, shop, add review.

- **Seller:**

Manage products, Update stocks.

- **Admin:**

Oversee all aspects of the system, manage products, orders, customers, and analytics.

5. Identify the users in your project?

- Admin
- Customer
- Seller

6. Who owns the system?

This is a sole idea of Pranav P K, young talented student of Amal Jyothi College of Engineering. He has the ownership of the idea and its web application.

7. System is related to which firm/industry/organization?

The "Bag Emporium" system is related to the e-commerce industry, specifically designed for online retailing of bags.

8. Details of person that you have contacted for data collection?

If the information for the "Bag Emporium" project was collected by friends, and bag shop owners it suggests that the details about the system's features and modules were obtained through informal means from individuals familiar with the project.

1. ChristoVarghese

9. Questionnaire to collect details about the project? (Min 10 questions, include descriptive answers, attach additional docs (e.g., Bill receipts), if any?)

- **What is the primary goal or objective of the "Bag Emporium" project?**

Answer: The primary goal or objective of the "Bag Emporium" project is to provide an online platform for customers to purchase a wide variety of bags conveniently. The project aims to offer a seamless shopping experience for customers by allowing them to browse, select, and purchase bags of their choice, make secure payments, and track their orders. For administrators, the objective is to efficiently manage products, orders, and customer information while monitoring sales performance.

- **What makes this online bag shop unique or different from competitors?**

Answer: The project offers a diverse range of bags, catering to different styles, sizes, and purposes, which sets it apart from competitors with limited options. The user interface is designed to be intuitive and easy to navigate, enhancing the overall shopping experience for customers.

- **How does the system stay competitive in the e-commerce industry?**

Answer: The system distinguishes itself through a diverse range of bags, a user-friendly interface, secure payment options, wishlist features, and efficient order management.

- **Who has ownership of the system, and how is system maintenance handled?**

Answer: The system is owned by the admin module, and system maintenance, including updates, is overseen by administrators.

- **How does the system enhance customer engagement and seller involvement?**

Answer: Customer engagement is fostered through features like feedback submission and order tracking, while sellers actively contribute by updating and adding new products.

- **What are the key requirements for customers using the platform?**

Answer: Customer requirements include account registration, browsing, product details, shopping cart management, order history tracking, and feedback submission.

- **How well does the system integrate with other organizational systems?**

Answer: The system seamlessly integrates with finance (payment processing), marketing, and inventory systems, ensuring smooth operations.

- **How does the system ensure a positive user experience, and what design principles are implemented to make browsing and purchasing intuitive for customers?**

Answer: The user experience is enhanced through a user-friendly interface, detailed product information, and features like wishlist functionality. Intuitive design principles guide customers through easy navigation, product selection, and a streamlined checkout process.

- **How does the system integrate with various organizational systems, and what benefits does this integration bring to overall business efficiency?**

Answer: The system seamlessly integrates with finance for payment processing, marketing for promotional activities, and inventory systems to ensure accurate product availability. This integration enhances overall business efficiency by centralizing data and streamlining cross-functional processes.

- **What specific functionalities should the admin have access to within the dashboard and overview section?**

The administrator should have access to various specific functionalities and key metrics to efficiently manage and monitor the online bag shop. These functionalities and metrics may include: Seller management, Seller approvals, Product management, Customer Management, Sales Analytics, Order Analytics.

3.2 SYSTEM SPECIFICATION

3.2.1 HARDWARE SPECIFICATION

Processor - Intel I3 or Above

RAM - 4 G B

Hard disk - 256 G B

3.2.2 SOFTWARE SPECIFICATION

Front End - HTML5, Bootstrap, CSS

Back End - Django Framework, Python

Database - SQLite

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, Python, CSS, Django, SQLite

3.3 SOFTWARE DESCRIPTION

3.3.1 DJANGO FRAMEWORK:

The Django Framework is a popular and robust web framework for Python developers. It is revered for its simplicity, clean code, and rapid development capabilities. Django is built on the Model-Template-Views (MTV) architectural pattern, which shares similarities with the Model-View-Controller (MVC) pattern used in other frameworks. One of its standout features is the Object-Relational Mapping (ORM) system, simplifying database interactions by representing database tables as Python objects. This abstraction eliminates the need for writing raw SQL queries, making database operations more straightforward.

Django also offers a built-in administrative interface, making content management a breeze. Its URL routing system allows developers to define clean and user-friendly URLs for their web applications. Furthermore, Django provides comprehensive support for form handling, data validation, and user authentication, reducing the complexity of common web development tasks. Security is a top priority in Django, with built-in protections against common web vulnerabilities like Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF). With a thriving community and an array of reusable packages, Django is a versatile choice for web development Projects of varying sizes and complexities.

3.1.1 SQLite:

SQLite is a lightweight, self-contained, and serverless relational database management system. Unlike traditional databases, it does not require a separate server but is embedded directly within the application. One of its standout features is the self-contained nature of SQLite databases; the

entire database is stored in a single file, simplifying management, backups, and transfers. Despite its lightweight design, SQLite is ACID-compliant, meaning it ensures data integrity by supporting transactions, making it suitable for multi-user environments.

SQLite is cross-platform and compatible with various operating systems, making it a versatile choice for applications targeting different platforms. Its small code size and minimal resource usage make it suitable for resource-constrained environments, such as mobile devices. SQLite is known for its speed and efficiency, especially in read-heavy workloads. It is commonly used in mobile applications, desktop applications, and embedded systems, where a full-fledged database server might be excessive, and portability is essential.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Any designed system or product's development starts with the design phase. An efficient system depends on well-executed design, which is a creative process. It entails utilizing a variety of approaches and concepts to define a process or system in enough depth to allow for its actual execution. Regardless of the development model chosen, the design phase is critical in software engineering. It strives to produce the architectural detail needed to build a system or product and serves as the technical backbone of the software engineering process. This program has through a thorough design phase that optimizes every aspect of effectiveness, performance, and accuracy. A user-oriented document is converted into a document for programmers or database employees during the design process.

4.2 UML DIAGRAM

A standardized dialect called Unified Modelling Language (UML) is utilized to conceptualize, characterize, plan, and depict program frameworks. The Question Administration Gather (OMG) was dependable for creating UML, and the primary draft of the UML 1.0 definition was discharged in January 1997. Programming dialects like Java, C++, and COBOL is not the same as UML. It could be a nonexclusive visual demonstrating dialect utilized for computer program frameworks and a pictorial dialect utilized for program outlines. UML may be utilized for non- software frameworks, such as fabricating forms, indeed though it is generally utilized to speak to program frameworks.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

A use case diagram may be a graphical delineation that appears how clients and other outside onscreen characters associated with a system's inside components. A use case diagram's essential work is to perceive, layout, and orchestrate a system's utilitarian needs as seen through the eyes of its clients. The Unified Modelling Language (UML), a standard language for modelling actual

things and systems, is frequently used to construct use case diagrams.

Use cases can be utilized to achieve an assortment of framework objectives, counting setting fundamental prerequisites, confirming equipment plans, testing, and investigating program, creating online offer assistance references, or performing client bolster obligations. Customer support, product obtaining, catalogue overhauling, and payment processing are as it were a couple of illustrations of use cases within the setting of item deals. The system boundaries, actors, use cases, and their connections together make up a use case diagram. The system boundary establishes the system's boundaries in reference to its surroundings. Actors are often defined depending on the roles they play and reflect the people or systems that interact with the system. The precise activities or behaviors that actors carry out within or close to the system are known as use cases. Finally, the graphic shows the connections between actors and use cases as well as the use cases themselves.

Use case diagrams are graphical representations used to capture the functional requirements of a system. When drawing a use case diagram, it is important to follow these guidelines to ensure an efficient and effective diagram:

- Choose descriptive names for use cases that accurately reflect the functionalities they perform.
- Assign appropriate names to actors to help identify their roles in the system.
- Ensure that relationships and dependencies are clearly depicted in the diagram.
- Avoid including every possible relationship, as the main goal is to identify the essential requirements.
- Use notes when necessary to clarify important points.

By following these guidelines, we can create a clear and concise use case diagram that accurately represents the functional requirements of the system.

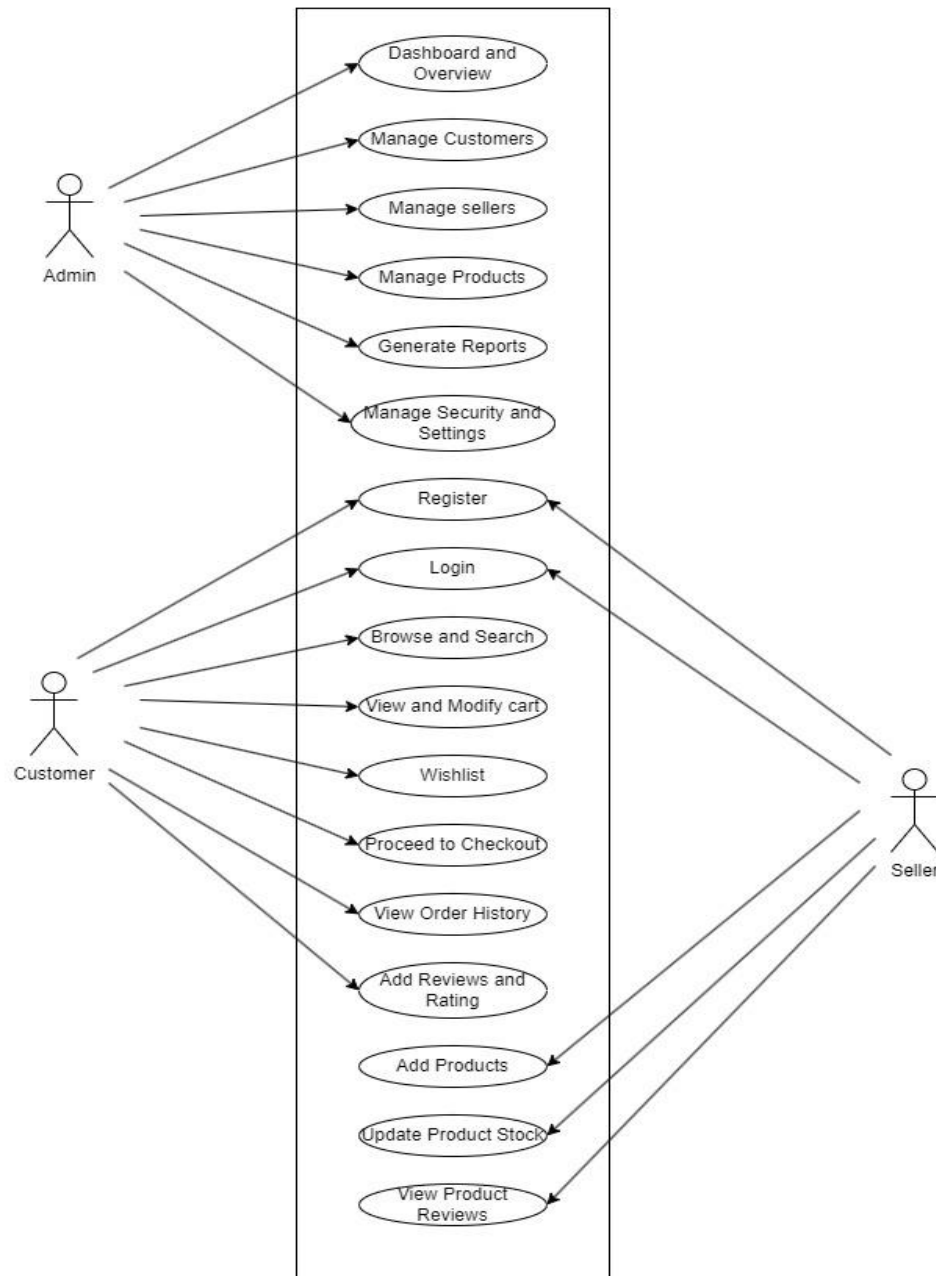


Fig 1: Use case diagram for Bag Emporium

4.2.2 SEQUENCE DIAGRAM

The chronological order of interactions between various system components is shown in a sequence diagram, a form of interaction diagram. It demonstrates how several things communicate with one another over the course of a series of messages. These images are sometimes referred to as event scenarios or event scenarios diagrams. In software engineering, sequence diagrams are frequently used to describe and comprehend the needs of both new and old systems. They support the visualization of object control relationships and the detection of systemic issues.

Sequence Diagram Notations –

- i. **Actors** - In UML, a role that interacts with the system and its objects is represented by an actor. Actors frequently exist outside of the system that the UML diagram is intended to portray. Actors can play a variety of roles, including those of external topics or human users. A stick person notation is used in UML diagrams to represent actors. Depending on the situation that is being modelled, a sequence diagram may have more than one actor.
- ii. **Lifelines** - A lifeline in a sequence diagram is a vertical dashed line that represents the lifespan of an object participating in the interaction. Each lifeline represents an individual participant in the sequence of events and is labeled with the name of the participant. The lifeline shows the timeline of events for the participant and is drawn as a vertical line extending from the participant's activation point to its deactivation point.
- iii. **Messages** - Messages are a key component of sequence diagrams, representing the interactions and communication between objects or components in a system. They can be categorized into synchronous and asynchronous messages, create, and delete messages, self-messages, reply messages, found messages, and lost messages. Guards are also used to model conditions and restrictions on message flow.
- iv. **Guards** - Guards in UML are used to model conditions and are employed to restrict the flow of messages when a certain condition is met. This feature is essential for letting software developers know about any constraints or limitations associated with a system or a particular process.

Uses of sequence diagram –

- Modeling and visualizing the logic of complex functions, operations, or procedures.
- Showing details of UML use case diagrams.
- Understanding the detailed functionality of current or future systems.
- Visualizing how messages and tasks move between objects or components in a system.

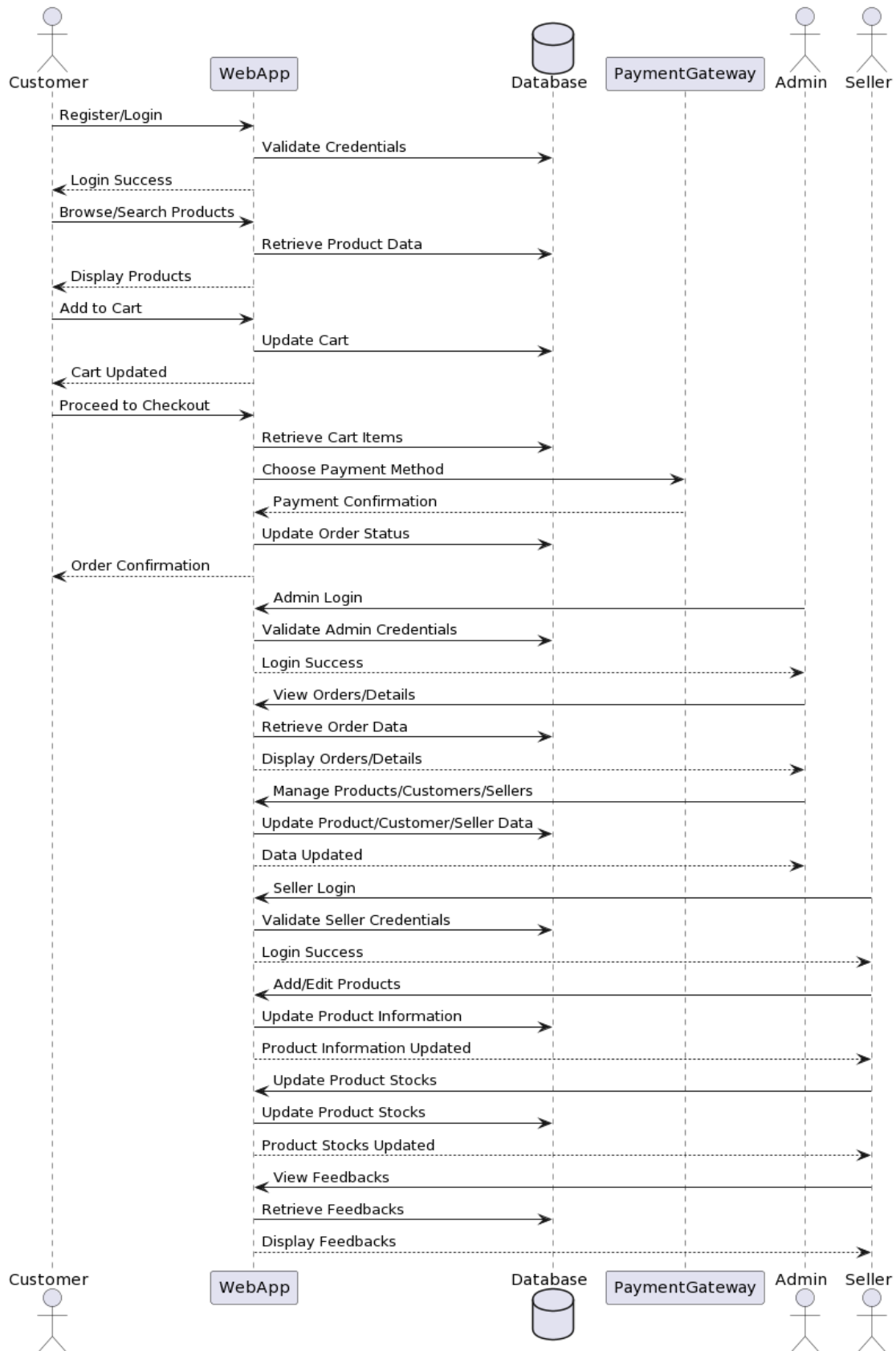


Fig 2: Sequence diagram for Bag Emporium

4.2.3 STATE CHART DIAGRAM

A state diagram is a visual representation, often created using the Unified Modeling Language (UML), that shows the different states that an object can exist in and how it can transition between those states. It is also referred to as a state machine diagram or state chart diagram.

The State Chart Diagram is a behavioral diagram in UML that describes the behavior of a system or object over time. It includes various elements such as:

- Initial State - This state represents the starting point of the system or object and is denoted by a solid black circle.
- State - This element describes the current state of the system or object at a specific point in time and is represented by a rectangle with rounded corners.
- Transition - This element shows the movement of the system or object from one state to another and is represented by an arrow.
- Event and Action - An event is a trigger that causes a transition to occur, and an action is the behavior or effect of the transition.
- Signal - A message or trigger caused by an event that is sent to a state, causing a transition to occur.
- Final State - The State Chart Diagram ends with a Final State element, which is represented by a solid black circle with a dot inside. It indicates that the behavior of the system or object has completed.

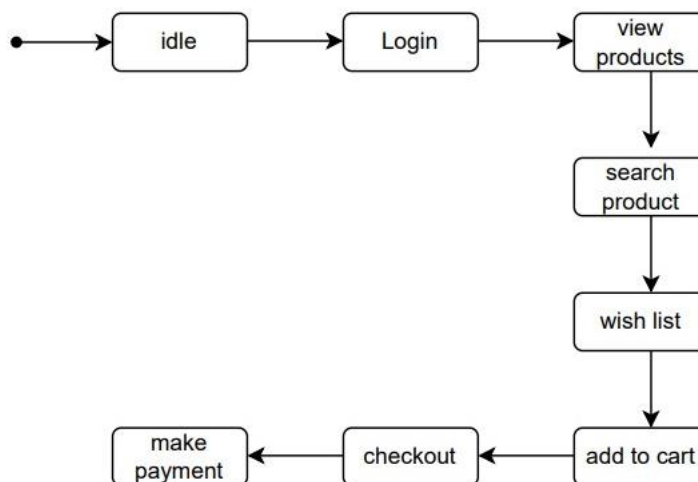


Fig 3: State Chart diagram for BagEmporium

4.2.4 ACTIVITY DIAGRAM

An activity diagram is a visual representation of a workflow that shows how one activity leads to another. An activity is referred to as a system operation, and one operation leads to another in the control flow. A flow can be parallel, concurrent, or branched, and activity diagrams use various functions such as branching, joining, etc., to manage all types of flow control. Activity diagrams are a type of behavior diagram that shows the behavior of a system. They show the flow of control from the start point to the end point and show the different decision paths that exist during the execution of the activity.

The key components of an activity diagram are:

- Initial node - A starting point of the activity diagram, denoted by a black circle.
- Activity - A task or action performed by the system or entity, represented by a rectangle with rounded corners.
- Control flow - It represents the sequence of activities or actions performed by the system or entity, represented by an arrow.
- Decision node - A decision or branching point in the activity flow, denoted by a diamond shape.
- Merge node - Used to merge multiple branches of the activity flow into a single flow, represented by a diamond shape with a plus sign inside.
- Fork node - Used to split the activity flow into multiple parallel flows, represented by a solid black circle with multiple arrows.
- Join node - Used to join multiple parallel flows back into a single flow, represented by a solid black circle with multiple arrows pointing towards it.
- Final node - The end point of the activity diagram, denoted by a black circle with a dot inside.
- Object flow - Represents the flow of objects or data between activities, represented by a dashed arrow.

Activity diagrams are useful in clarifying complex processes, identifying potential issues, and communicating process flows to stakeholders and project team members.

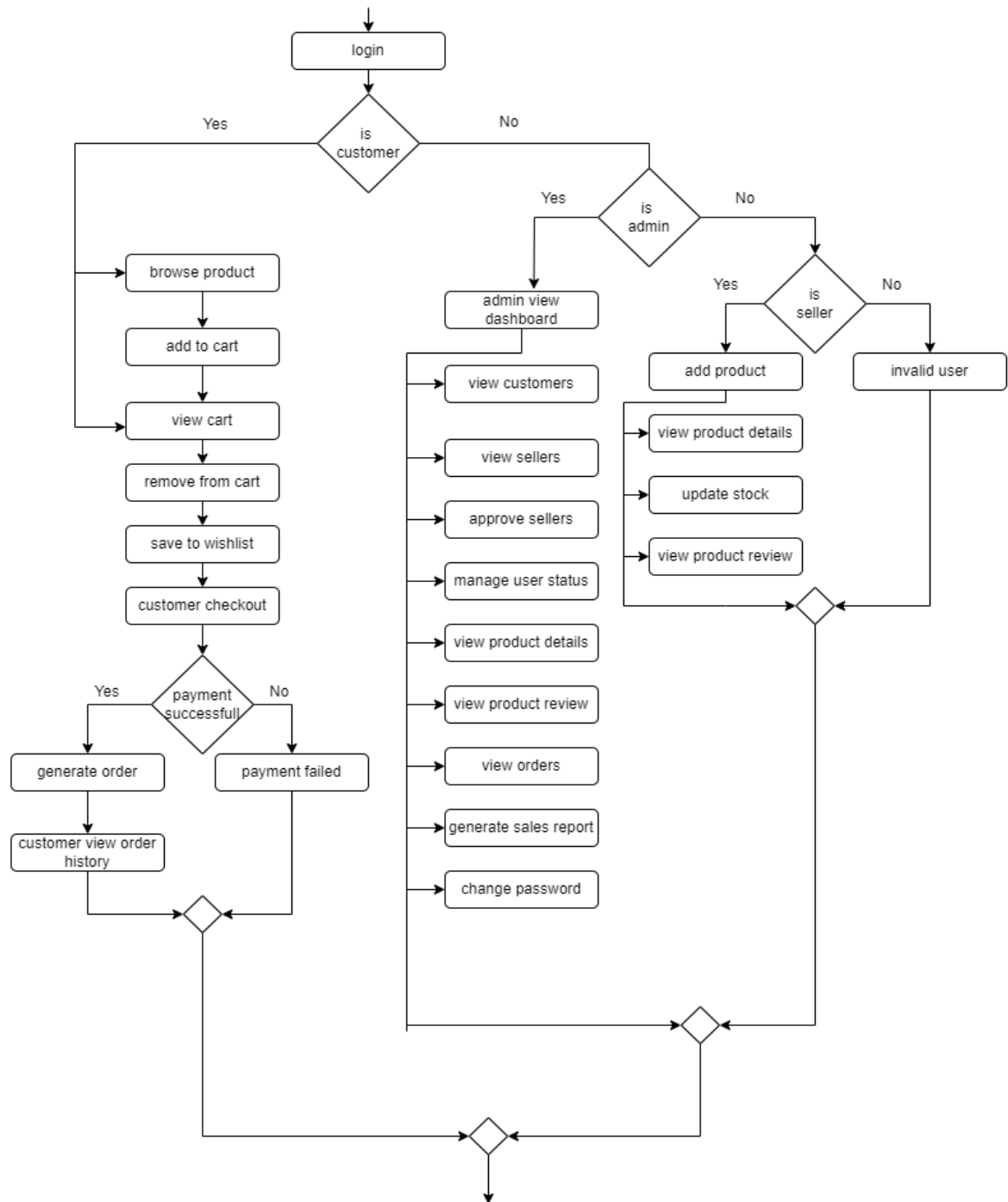


Fig 4: Activity diagram for Bag Emporium

4.2.5 CLASS DIAGRAM

The class diagram is a fundamental component of object-oriented modeling and serves as the primary means of conceptual modeling for the structure of an application. Additionally, class diagrams can be used for detailed modeling that can be translated into programming code. They can also be employed for data modeling purposes. Class diagrams are a crucial component of UML used to represent classes, objects, interfaces, and their relationships and attributes in a system.

Some important components of a class diagram are:

- **Class:** It is a blueprint or template for creating objects and is represented as a rectangle with the class name, attributes, and methods.
- **Interface:** It is a collection of abstract methods that specify a contract between a class and the outside world. It is represented as a circle with the interface name inside.
- **Object:** It is an instance of a class with state and behavior. It is represented as a rectangle with the object name inside.
- **Association:** It is a relationship between two classes that represents a connection or link and is represented as a line with optional directionality, multiplicity, and role names.
- **Aggregation:** It is a part-whole relationship where the whole (aggregator) is composed of parts (aggregates) and is represented as a diamond shape on the aggregator side.
- **Composition:** It is a stronger form of aggregation where the parts cannot exist without the whole and is represented as a filled diamond shape on the aggregator side.
- **Inheritance:** It is a relationship between a superclass and its subclasses that represents an "is-a" relationship and is represented as a line with an open arrowhead pointing from the subclass to the superclass.
- **Dependency:** It is a relationship where a change in one class may affect the other class and is represented as a dashed line with an arrowhead pointing from the dependent class to the independent class.
- **Multiplicity:** It represents the number of instances of a class that can be associated with another class and is represented as a range of values near the association or aggregation line.

Class diagrams are essential in designing and modeling object-oriented software systems as they provide a visual representation of the system's structure, its functionality, and the relationships between its objects. They facilitate software development, maintenance, and improve communication among team members.

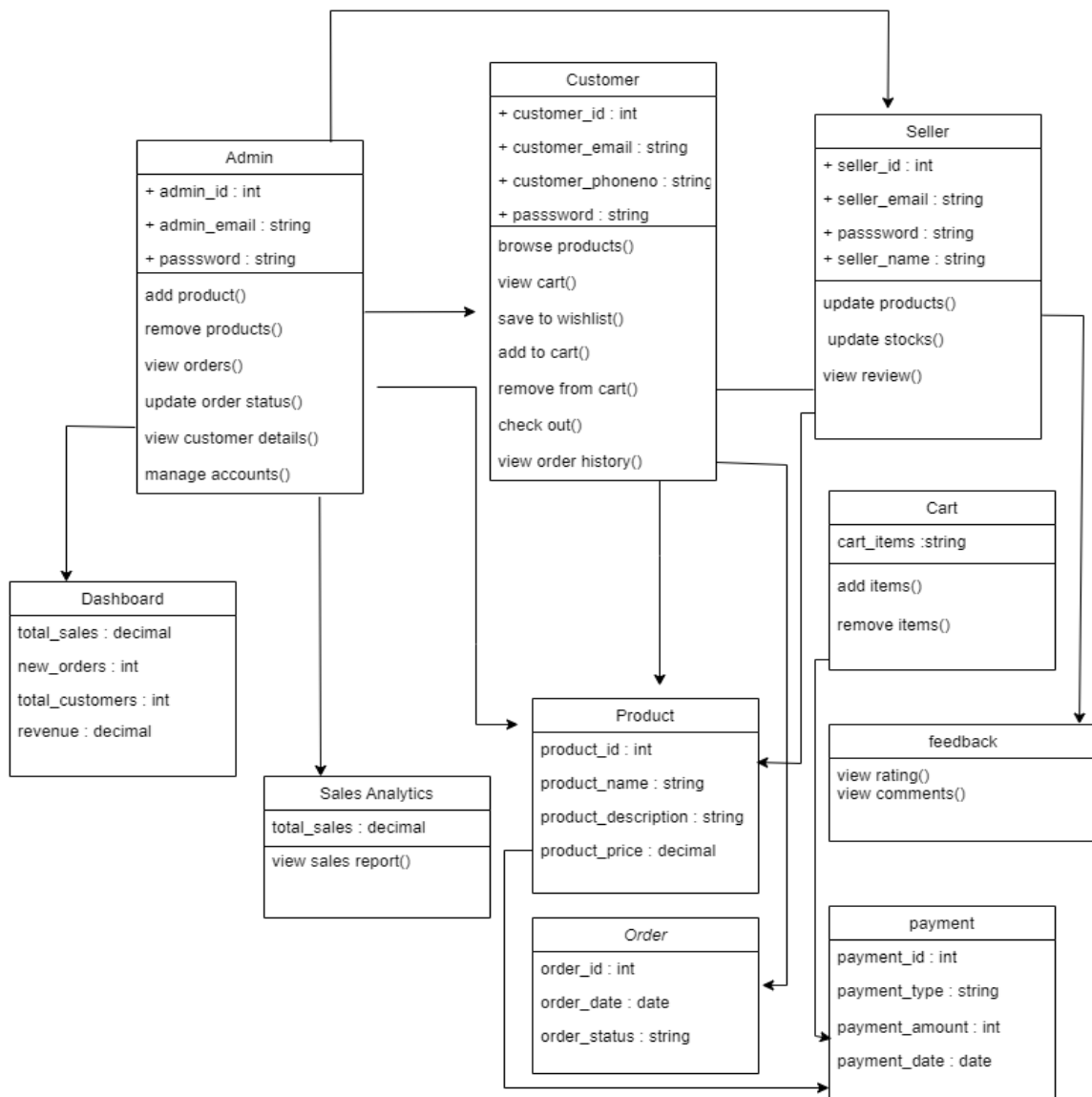


Fig 5: Class diagram for Bag Emporium

4.2.6 OBJECT DIAGRAM

Class diagrams and object diagrams are closely related in object-oriented modeling. Object diagrams are instances of class diagrams, which represent a snapshot of the system at a given moment in time. Both types of diagrams use the same concepts and notation to represent the structure of a system. While class diagrams are used to model the structure of the system, including its classes, attributes, and methods, object diagrams represent a group of objects and their connections at a specific point in time.

An object diagram is a type of structural diagram in UML that shows instances of classes and their relationships. The main components of an object diagram include:

- **Object:** An object is an instance of a class that represents a specific entity in the system. It is represented as a rectangle with the object name inside.
- **Class:** A class is a blueprint or template for creating objects that defines its attributes and methods. It is represented as a rectangle with three compartments for the class name, attributes, and methods.
- **Link:** A link is a relationship between two objects that represents a connection or association. It is represented as a line connecting two objects with optional labels.
- **Attribute:** An attribute is a property or characteristic of an object that describes its state. It is represented as a name-value pair inside the object rectangle.
- **Value:** A value is a specific instance or setting of an attribute. It is represented as a value inside the attribute name-value pair.
- **Operation:** An operation is a behavior or action that an object can perform. It is represented as a method name inside the class rectangle.
- **Multiplicity:** Multiplicity represents the number of instances of a class that can be associated with another class.

Object diagrams help to visualize the relationships between objects and their attributes in a system. They are useful for understanding the behavior of a system at a specific point in time and for identifying potential issues or inefficiencies in the system.

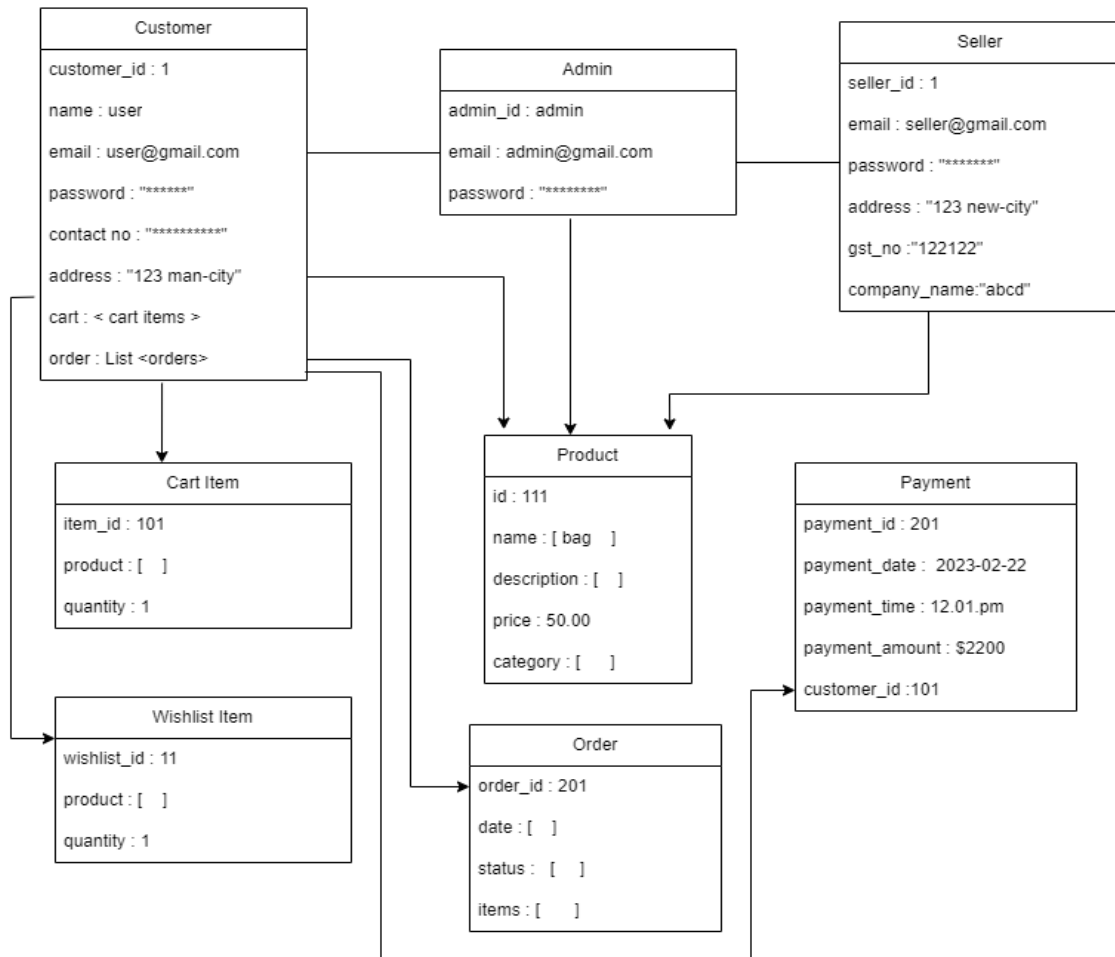


Fig 6: Object diagram for Bag Emporium

4.2.7 COMPONENT DIAGRAM

A component diagram in UML illustrates how various components are interconnected to create larger components or software systems. It is an effective tool for representing the structure of complex systems with multiple components. By using component diagrams, developers can easily visualize the internal structure of a software system and understand how different components work together to accomplish a specific task.

Its key components include:

- **Component:** A modular and encapsulated unit of functionality in a system that offers interfaces to interact with other components. It is represented as a rectangle with the component name inside.
- **Interface:** A contract between a component and its environment or other components, specifying a set of methods that can be used by other components. It is represented as a circle with the interface name inside.
- **Port:** A point of interaction between a component and its environment or other components. It is represented as a small square on the boundary of a component.
- **Connector:** A link between two components that enables communication or data exchange. It is represented as a line with optional adornments and labels.
- **Dependency:** A relationship between two components where one component depends on another for its implementation or functionality. It is represented as a dashed line with an arrowhead pointing from the dependent component to the independent component.
- **Association:** A relationship between two components that represents a connection or link. It is represented as a line connecting two components with optional directionality, multiplicity, and role names.
- **Provided/Required Interface:** A provided interface is an interface that a component offers to other components, while a required interface is an interface that a component needs from other components to function properly. These are represented by lollipops and half-circles respectively.

Component diagrams are useful for modeling the architecture of a software system, and can help identify potential issues and improvements in the design. They can also be used to communicate the structure and behavior of a system to stakeholders, such as developers and project managers.

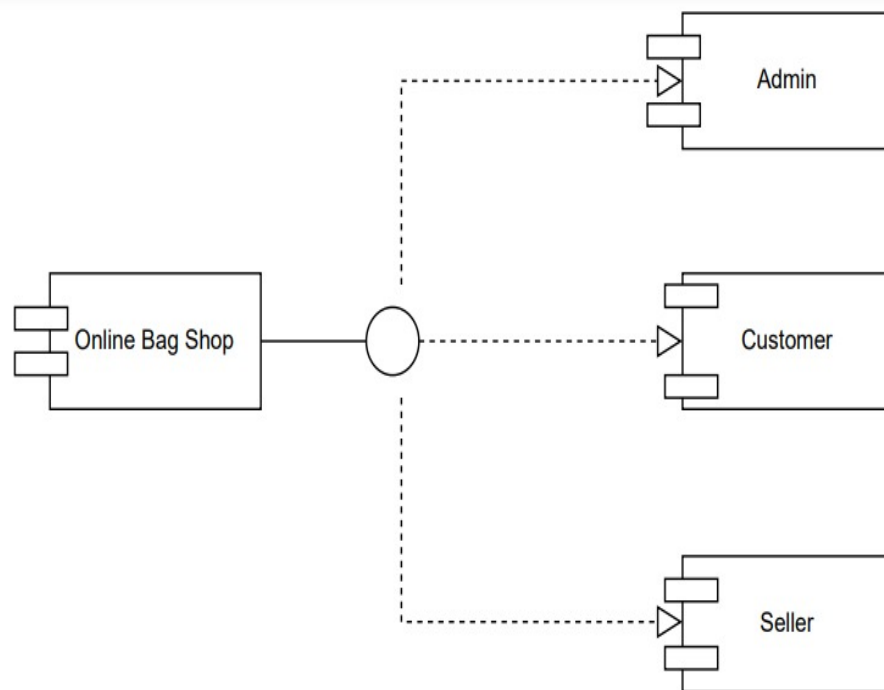


Fig 7: Component diagram for Bag Emporium

4.2.8 DEPLOYMENT DIAGRAM

A deployment diagram is a type of UML diagram that focuses on the physical hardware used to deploy software. It provides a static view of a system's deployment and involves nodes and their relationships. The deployment diagram maps the software architecture to the physical system architecture, showing how the software will be executed on nodes. Communication paths are used to illustrate the relationships between the nodes. Unlike other UML diagram types, which focus on the logical components of a system, the deployment diagram emphasizes the hardware topology.

The key components of a deployment diagram are:

- **Node** - A node is a physical or virtual machine on which a component or artifact is deployed. It is represented by a box with the node's name inside.
- **Component** - A component is a software element that performs a specific function or provides a specific service. It is represented by a rectangle with the component's name inside.
- **Artifact** - An artifact is a physical piece of data that is used or produced by a component. It is represented by a rectangle with the artifact's name inside.
- **Deployment Specification** - A deployment specification describes how a component or artifact is deployed on a node. It includes information about the location, version, and configuration parameters of the component or artifact.
- **Association** - An association is a relationship between a node and a component or artifact that represents a deployment dependency. It is represented by a line connecting the two components with optional directionality, multiplicity, and role names.
- **Communication Path** - A communication path represents the connection between nodes, such as a network connection or communication channel. It is represented by a line with optional labels and adornments.

Deployment diagrams help in visualizing the physical architecture of a system and identifying any potential issues or bottlenecks in the deployment process. They also aid in planning the deployment strategy and optimizing the use of hardware resources.

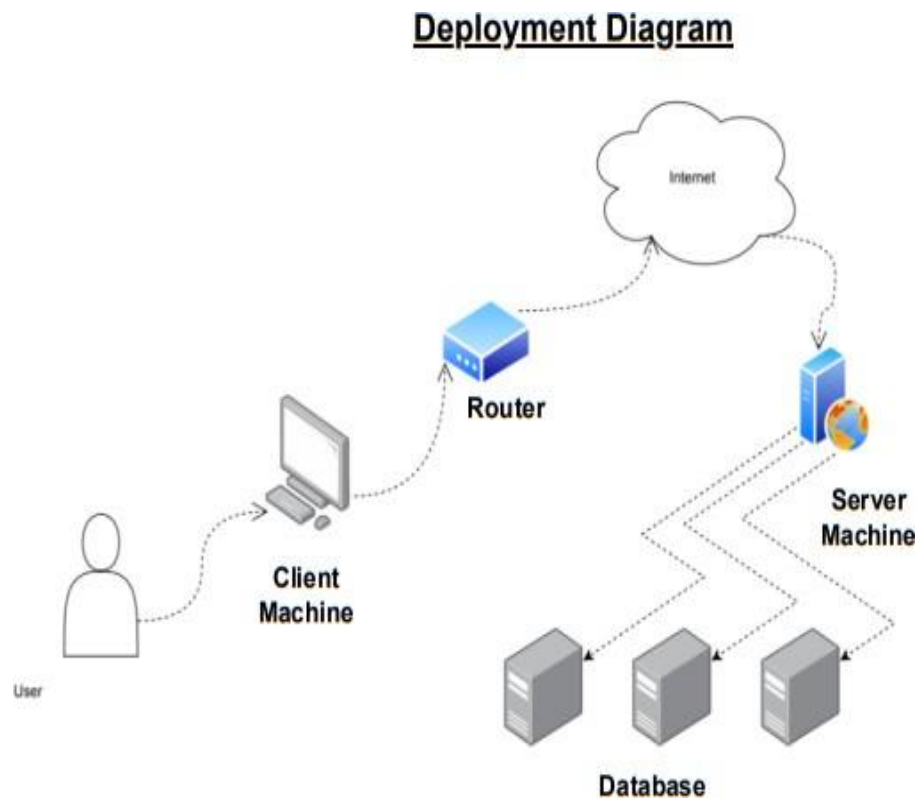


Fig 8: Deployment diagram for Bag Emporium

4.3 USER INTERFACE DESIGN USING FIGMA

Form Name: Login Form

Home Explore About Us Register


Login Here

Email

Password

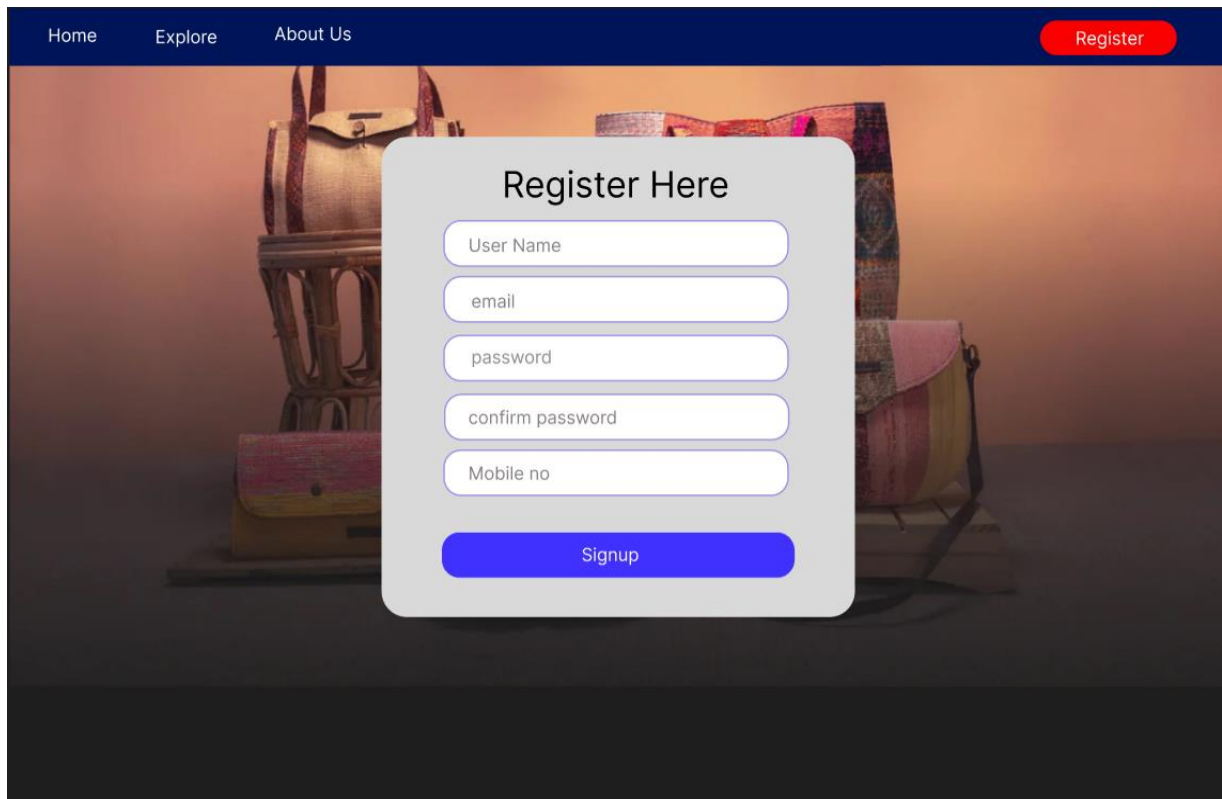
login

--OR--

Continue with 

Don't have account

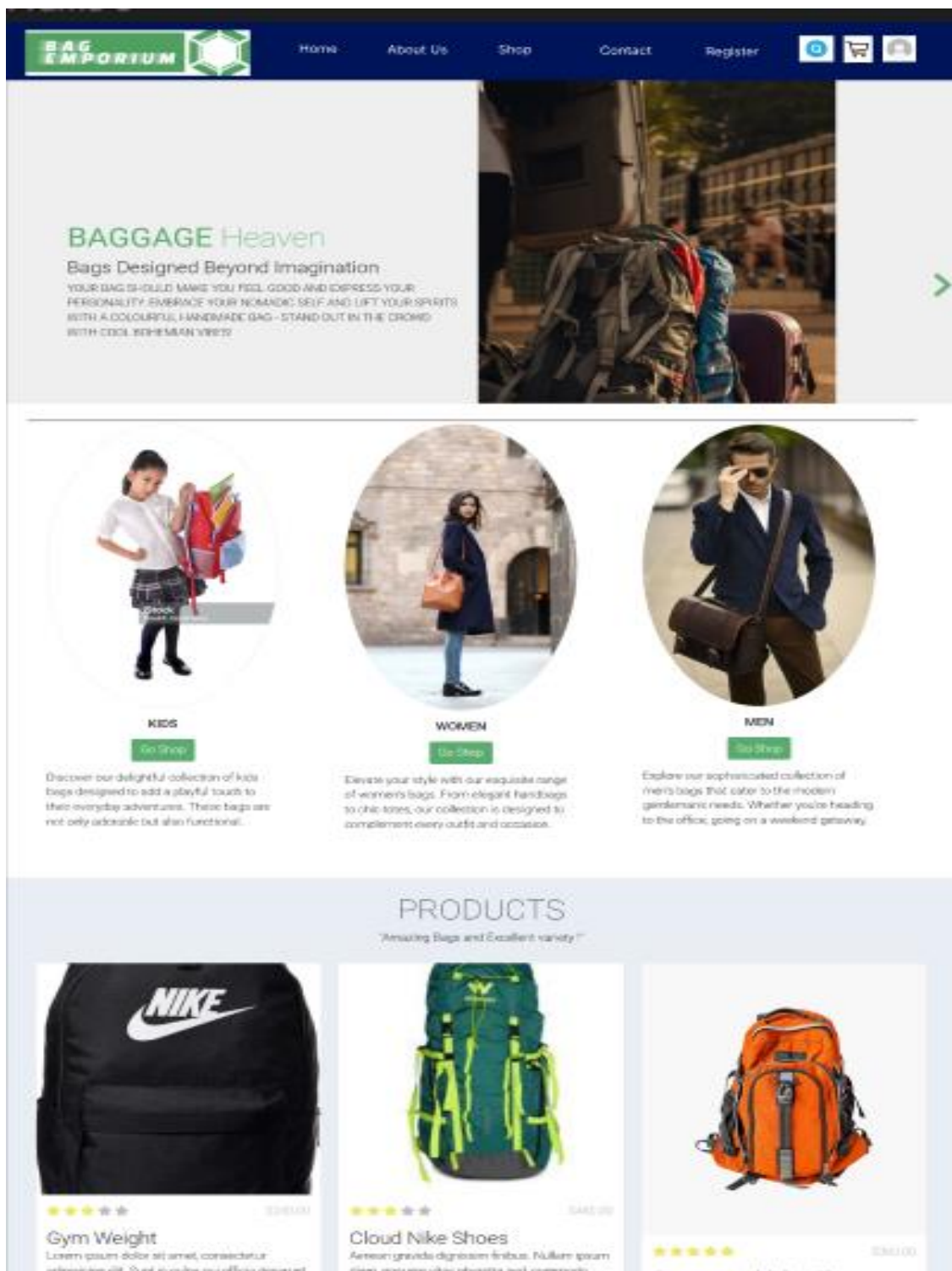
Signup

Form Name: Customer Signup

The image shows a web application interface for a customer signup form. The background is a dark blue header with navigation links: Home, Explore, and About Us. A red 'Register' button is in the top right corner. The main content area features a large, semi-transparent modal box titled 'Register Here'. Inside the modal, there are five input fields: 'User Name', 'email', 'password', 'confirm password', and 'Mobile no'. Below these fields is a blue 'Signup' button. The background of the page shows a collection of colorful, patterned bags and backpacks.

Form Name: Admin Dashboard

Form Name: Home page



4.4 DATABASE DESIGN

A database is an organized collection of information that's organized to enable easy accessibility, administration, and overhauls. The security of information could be an essential objective of any database. The database design process comprises of two stages. In the first stage, user requirements are gathered to create a database that meets those requirements as clearly as possible. This is known as information-level design and is carried out independently of any DBMS. In the second stage, the design is converted from an information-level design to a specific DBMS design that will be used to construct the system. This stage is known as physical-level design, where the characteristics of the specific DBMS are considered. Alongside system design, there is also database design, which aims to achieve two main goals: data integrity and data independence.

4.4.1 RELATIONAL DATABASE MANAGEMENT SYSTEM

A relational database management system (RDBMS) is a popular type of database that organizes data into tables to facilitate relationships with other stored data sets. Tables can contain vast amounts of data, ranging from hundreds to millions of rows, each of which are referred to as records. In formal relational model language, a row is called a tuple, a column heading is an attribute, and the table is a relation. A relational database consists of multiple tables, each with its own name. Each row in a table represents a set of related values.

In a relational database, relationships are already established between tables to ensure the integrity of both referential and entity relationships. A domain D is a group of atomic values, and a common way to define a domain is by choosing a data type from which the domain's data values are derived. It is helpful to give the domain a name to make it easier to understand the values it contains. Each value in a relation is atomic and cannot be further divided.

In a relational database, table relationships are established using keys, with primary key and foreign key being the two most important ones. Entity integrity and referential integrity relationships can be established with these keys. Entity integrity ensures that no primary key can have null values, while referential integrity ensures that each distinct foreign key value must have a matching primary key value in the same domain. Additionally, there are other types of keys such as super keys and candidate keys.

4.4.2 NORMALIZATION

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique,

superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are two different kinds of keys. Primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized.

Normalization is a process in database design that aims to organize data into proper tables and columns, making it easily correlated to the data by the user. This process eliminates data redundancy that can be a burden on computer resources. The main steps involved in normalization include:

- Normalizing the data
- Choosing appropriate names for tables and columns
- Choosing the correct names for the data

By following these steps, a developer can create a more efficient and organized database that is easier to manage and maintain.

First Normal Form-

The First Normal Form (1NF) requires that each attribute in a table must contain only atomic or indivisible values. It prohibits the use of nested relations or relations within relations as attribute values within tuples. To satisfy 1NF, data must be moved into separate tables where the data is of similar type in each table, and each table should have a primary key or foreign key as required by the project. This process eliminates repeating groups of data and creates new relations for each non-atomic attribute or nested relation. A relation is in 1NF only if it satisfies the constraints that contain the primary key only

Second Normal Form-

Second normal form (2NF) is a rule in database normalization that states that non-key attributes should not be functionally dependent on only the part of the primary key in a relation that has a composite primary key. In other words, each non-key attribute should depend on the entire primary key, not just a part of it. To achieve this, we need to decompose the table and create new relationships for each subkey along with their dependent attributes. It is important to maintain the relationship with the original primary key and all attributes that are fully functionally dependent

on it. A relation is said to be in 2NF only if it satisfies all the 1NF conditions for the primary key and every non-primary key attribute of the relation is fully dependent only on the primary key

Third Normal Form-

Third normal form (3NF) requires that a relation have no non-key attribute that is functionally determined by another non-key attribute or set of non-key attributes. This means that there should be no transitive dependency on the primary key. To achieve 3NF, we decompose the relation and set up a new relation that includes non-key attributes that functionally determine other non-key attributes. This helps eliminate any dependencies that do not just rely on the primary key. A relation is considered a relation in 3NF if it satisfies the conditions of 2NF and, moreover, the non-key attributes of the relation are not dependent on any other non-key attribute

4.4.3 SANITIZATION

Data sanitization is the process of removing any illegal characters or values from data. In web applications, sanitizing user input is a common task to prevent security vulnerabilities. PHP provides a built-in filter extension that can be used to sanitize and validate various types of external input such as email addresses, URLs, IP addresses, and more. These filters are designed to make data sanitization easier and faster. For example, the PHP filter extension has a function that can remove all characters except letters, digits, and certain special characters (!#\$%&'*+ -=?_`{|}~@.[]), as specified by a flag. Web applications often receive external input from various sources, including user input from forms, cookies, web services data, server variables, and database query results. It is important to sanitize all external input to ensure that it is safe and does not contain any malicious code or values.

4.4.4 INDEXING

An index is a database structure that enhances the speed of table operations. Indexes can be created on one or more columns to facilitate quick lookups and efficient ordering of records. When creating an index, it is important to consider which columns will be used in SQL queries and to create one or more indexes on those columns. In practice, indexes are a type of table that store a primary key or index field and a pointer to each record in the actual table. Indexes are invisible to users and are only used by the database search engine to quickly locate records. The CREATE INDEX statement is used to create indexes in tables. When tables have indexes, the INSERT and UPDATE statements take longer because the database needs to insert or update the index values as well. However, the SELECT statements become faster on those tables because the index allows the database to locate records more quickly

4.5 TABLE DESIGN

1. User Table (Extended From AbstractUser):

Primary key: **User_ID**

No:	Field Name	Data Type	Key Constraints	Description
1	User_id	Int	Primary Key	User Id
2	User_First_name	Varchar (30)		User FirstName
3	User_Last_name	Varchar (30)		User LastName
4	User_Email	Varchar (30)		Email
5	User_Password	Varchar (30)		Password
6	User_role	Varchar (30)		User Role

2. Customer (Update profile Table)

Primary key: **Customer_id**

Foreign key: **User_id** references table **User**

No:	Field Name	Data Type	Key Constraints	Description
1	Customer_Id	Int	Primary key	Id
2	User_id	Int	Foreign key	User id
3	Customer_Profile_picture	Image		Profile picture
4	Customer_Phone	Varchar (30)		Phone number
5	Customer_Alt_phone	Varchar (30)		Alternate phone_number
6	Customer_Pincode	Varchar (30)		Pincode
7	Customer_State	Varchar (30)		State
8	Customer_City	Varchar (30)		City
9	Customer_Building_name	Varchar (30)		Building name
10	Customer_Road_area	Varchar (30)		Road/area

3. Seller (Update profile Table)

Primary key: **Seller_ID**

Foreign key: **User_id** references table **User**

No:	Field Name	Data Type	Key Constraints	Description
1	Seller_Id	Int	Primary key	Id
2	User_id	Int	Foreign key	User Id
3	Seller_Phone	Varchar (20)		Phone number
4	Seller_Alt_phone	Varchar (20)		Alternate phone number
5	Seller_Profile_picture	Image		Profile picture
6	Seller_Gst	Varchar (20)		Gst number
7	Seller_Company_name	Varchar (20)		Company name
8	Seller_Country	Varchar (20)		Country
9	Seller_Pincode	Varchar (20)		Pincode
10	Seller_State	Varchar (20)		State
11	Seller_City	Varchar (20)		City
12	Seller_Address	Varchar (20)		Address
13	Is_approved	Boolean		Approval status
14	Incorporation_certificate	File		Incorporation certificate

4. Category

Primary key: **Category_ID**

No:	Field Name	Data Type	Key Constraints	Description
1	Category Id	Int	Primary Key	Category Id
2	Category_name	Varchar (330)		Category Name
3	Category_description	Varchar (330)		Category Description
4	Category_picture	Image		Category Picture
5	Category_verify	Boolean		status

5. Sub_category

Primary key: **Sub_category_ID**

Foreign key: **Category_ID** references table **Category**

No:	Field Name	Data Type	Key Constraints	Description
1	Sub_Category_id	Int	Primary Key	Sub_Category_id
2	Category_id	Int	Foreign key	Category Id
3	Sub_Category_name	Varchar (330)		Sub Category Name
4	Sub_Category_description	Varchar (330)		Sub Category Description
5	Sub_Category_picture	Image		Sub Category Picture
6	Sub_Category_verify	Boolean		status

6. Product_Review

Primary key: **Review_ID**

Foreign key: **User_id** references **User**

Foreign key: **Product_id** references **Product**

No:	Field Name	Data Type	Key Constraints	Description
1	Product_Review Id	Int	Primary Key	Review Id
2	User_id	Int	Foreign key	User Id
3	Product_id	Int	Foreign key	Product Id
4	Product_Review_Created at	Varchar (30)		Creation Date and Time
5	Product_rating	Int		Product Rating
6	Product_review_Description	Varchar (30)		Review Description

7. Image

Primary key: **Image_ID**

Foreign key: **Product_id** references **Product**

No:	Field Name	Data Type	Key Constraints	Description
1	Image_Id	Int	Primary Key	Product Image Id
2	Product_id	Int	Foreign key	Product Id
3	Image_data	Image		Product Image Data

8. Cart

Primary key: **Cart_ID**

Foreign key: **User_id** references **User**

No:	Field Name	Data Type	Key Constraints	Description
1	Cart_Id	Int	Primary key	Cart Id
2	User_id	Int	Foreign key	User Id

9.Cart_Item

Primary key: **Cart_item_ID**

Foreign key: **Cart_id** references **Cart**

Foreign key: **User_id** references **User**

Foreign key: **Product_id** references **Product**

No:	Field Name	Data Type	Key Constraints	Description
1	Cart Item Id	Int	Primary Key	Cart Item Id
2	User_id	Int	Foreign key	User Id
3	Product_id	Int	Foreign key	Product Id
4	Cart_id	Int	Foreign key	Cart Id
5	Cart_item_Quantity	Int		Quantity of Products
6	Cart_item_price	int		Price of Product
7	Cart_item_verify	Boolean		Cart Verify

10.Payment

Primary key: **Payment_ID**

Foreign key: **User_id** references **User**

Foreign key: **Cart_id** references **Cart**

No:	Field Name	Data Type	Key Constraints	Description
1	Payment Id	Int	Primary Key	Payment Id
2	User_id	Int	Foreign key	User_id
3	Payment_Order_data	Int		Order_id_data
4	Payment_Amount	Int		Payment Amount
5	Cart_id	Int	Foreign key	Cart_id
6	Payment_Date_Time	DateTime		Date_Time
7	Payment_id_data	int		Payment_id_data

11.Wishlist

Primary key: **Wishlist_ID**

Foreign key: **User_id** references **User**

Foreign key: **Product_id** references **Product**

No:	Field Name	Data Type	Key Constraints	Description
1	Wishlist Id	Int	Primary Key	Wishlist Id
2	User_id	Int	Foreign key	User Id
3	Product_id	Int	Foreign key	Product Id

12.Product

Primary key: **Product_ID**

Foreign key: **Seller_id** references table **Seller**

Foreign key: **category_id** references table **category**

Foreign key: **Sub_category_id** references table **Seub_Category**

No:	Field Name	Data Type	Key Constraints	Description
1	Product_Id	Int	Primary key	Product_Id
2	Seller_id	Int	Foreign key	Seller Id
3	Product_Brand_name	Varchar (40)		Brand Name
4	Product_name	Varchar (40)		Product name
5	Product_number	Varchar (20)	Unique	Product Number
6	Product_Stock	Int		Stock Quantity
7	Product_Description	Varchar (200)		Product Description
8	Product_Current_price	Decimal		Current price
9	Category_id	Int	Foreign key	Category Id
10	Sub_category_id	Int	Foreign key	Sub Category id
11	Product_Capacity	Varchar (60)		Product Capacity
12	Product_Color	Varchar (20)		Product Color
13	Product_Material	Varchar (50)		Product Material
14	Product_status	Boolean		Product Status
15	Product_img_1	Image		Product Main Image

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software testing involves executing a software program in a controlled manner to determine if it behaves as intended, often using verification and validation methods. Validation involves evaluating a product to ensure it complies with specifications, while verification can involve reviews, analyses, inspections, and walkthroughs. Static analysis examines the software's source code to identify issues, while dynamic analysis examines its behavior during runtime to gather information like execution traces, timing profiles, and test coverage details.

Testing involves a series of planned and systematic activities that start with individual modules and progress to the integration of the entire computer-based system. The objectives of testing include identifying errors and bugs in the software, ensuring that the software functions according to its specifications, and verifying that it meets performance requirements. Testing can be performed to assess correctness, implementation efficiency, and computational complexity.

A successful test is one that detects an undiscovered error, and a good test case has a high probability of uncovering such errors. Testing is crucial to achieving system testing objectives and can involve various techniques such as functional testing, performance testing, and security testing.

5.2 TEST PLAN

A test plan is a document that outlines the required steps to complete various testing methodologies. It provides guidance on the activities that need to be performed during testing. Software developers create computer programs, documentation, and associated data structures. They are responsible for testing each component of the program to ensure it meets the intended purpose. To address issues with self-evaluation, an independent test group (ITG) is often established.

Testing objectives should be stated in quantifiable language, such as mean time to failure, cost to find and fix defects, remaining defect density or frequency of occurrence, and test work-hours per regression test.

The different levels of testing include:

- Unit testing
- Integration testing
- Data validation testing
- Output testing

5.2.1 UNIT TESTING

Unit testing is a software testing technique that focuses on verifying individual components or modules of the software design. The purpose of unit testing is to test the smallest unit of software design and ensure that it performs as intended. Unit testing is typically white-box focused, and multiple components can be tested simultaneously. The component-level design description is used as a guide during testing to identify critical control paths and potential faults within the module's perimeter.

During unit testing, the modular interface is tested to ensure that data enters and exits the software unit under test properly. The local data structure is inspected to ensure that data temporarily stored retains its integrity during each step of an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once, and all error handling paths are tested to ensure that the software can handle errors correctly.

Before any other testing can take place, it is essential to test data flow over a module interface. If data cannot enter and exit the system properly, all other tests are irrelevant. Another crucial duty during unit testing is the selective examination of execution pathways to anticipate potential errors and ensure that error handling paths are set up to reroute or halt work when an error occurs. Finally, boundary testing is conducted to ensure that the software operates correctly at its limits.

In the ShareSphere System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. Any issues with the internal logic of the modules were fixed, and each module was tested and run separately after coding. Unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome.

5.2.2 INTEGRATION TESTING

Integration testing is a systematic approach that involves creating the program structure while simultaneously conducting tests to identify interface issues. The objective is to construct a program structure based on the design that uses unit-tested components. The entire program is then tested. Correcting errors in integration testing can be challenging due to the size of the overall program, which makes it difficult to isolate the causes of the errors. As soon as one set of errors is fixed, new ones may arise, and the process may continue in an apparently endless cycle.

Once unit testing is complete for all modules in the system, they are integrated to check for any interface inconsistencies. Any discrepancies in program structures are resolved, and a unique program structure is developed.

5.2.3 VALIDATION TESTING OR SYSTEM TESTING

The final stage of the testing process involves testing the entire software system as a whole, including all forms, code, modules, and class modules. This is commonly referred to as system testing or black box testing. The focus of black box testing is on testing the functional requirements of the software. A software engineer can use this approach to create input conditions that will fully test each program requirement. The main types of errors targeted by black box testing include incorrect or missing functions, interface errors, errors in data structure or external data access, performance errors, initialization errors, and termination errors.

5.2.4 OUTPUT TESTING OR USER ACCEPTANCE TESTING

User acceptance testing is performed to ensure that the system meets the business requirements and user needs. It is important to involve the end users during the development process to ensure that the software aligns with their needs and expectations. During user acceptance testing, the input and output screen designs are tested with different types of test data. The preparation of testdata is critical to ensure comprehensive testing of the system. Any errors identified during testing are addressed and corrected, and the corrections are noted for future reference.

5.2.5 AUTOMATION TESTING

Automation testing is a software testing approach that employs specialized automated testing software tools to execute a suite of test cases. Its primary purpose is to verify that the software or equipment operates precisely as intended. Automation testing identifies defects, bugs, and other issues that may arise during product development. While some types of testing, such as functional or regression testing, can be performed manually, there are numerous benefits to automating the process. Automation testing can be executed at any time of day and uses scripted sequences to evaluate the software. The results are reported, and this information can be compared to previous test runs. Automation developers typically write code in programming languages such as C#, JavaScript, and Ruby.

5.2.6 SELENIUM TESTING

Selenium is an open-source automated testing framework used to verify web applications across different browsers and platforms. Selenium allows for the creation of test scripts in various programming languages such as Java, C#, and Python. Jason Huggins developed Selenium in 2004 while working on a web application that required frequent testing. He created a JavaScript program called "JavaScriptTestRunner" to automate browser actions and improve testing efficiency. Selenium has since evolved and continues to be developed by a team of contributors.

In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It allows for the creation of executable specifications in a human-readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals.

Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web applications across different browsers and platforms, using a business-readable and maintainable format.

Test Case 1:

Customer- Login

```
package cucumber_java;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.By;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.When;

import io.cucumber.java.en.Then;

public class Loginsteps {

    WebDriver driver;

    @Given("browser is open")

    public void browserIsOpen() {

        System.setProperty("webdriver.gecko.driver",
```



```
"C:\\Users\\Lenovo\\Desktop\\geckodriver.exe");

    driver = new FirefoxDriver();
    driver.manage().window().maximize();
}
@Given("user is on login page")

public void userIsOnLoginPage() {

    System.setProperty("webdriver.gecko.driver",
"C:\\Users\\Lenovo\\Desktop\\geckodriver.exe");

    driver = new FirefoxDriver();

    driver.manage().window().maximize();

    String loginPageURL = "http://127.0.0.1:8000/myauth/login/";

    driver.get(loginPageURL);

}

@When("user enters email and password")

public void userEntersUsernameAndPassword() {

    // Locate the username and password fields and input data

    WebElement usernameField = driver.findElement(By.id("email")); // Replace "username"
with the actual ID of the username field

    WebElement passwordField = driver.findElement(By.id("password")); // Replace
"password" with the actual ID of the password field

    usernameField.sendKeys("pranavpk2024b@mca.ajce.in"); // Replace "your_username"
with the actual username

    passwordField.sendKeys("pRANAV@2002"); // Replace "your_password" with the actual
password

}

@When("User clicks on login")

public void userClicksOnLogin() {

    // Locate and click the login button

    WebElement loginButton = driver.findElement(By.id("submit")); // Replace "login-button"
```

with the actual ID of the login button

```
loginButton.click();
```

```
@Then("user is navigated to the home page")
```

```
public void userIsNavigatedToHomePage() {
```

```
    // Verify navigation to the home page by checking the URL, title, or elements on the home page
```

```
    String homePageURL = driver.getCurrentUrl();
```

```
String expectedHomePageURL = "http://127.0.0.1:8000/myauth/home/"; // Replace
```

```
    if (homePageURL.equals(expectedHomePageURL)) {  
with the expected URL of the home page
```

```
        // Navigation to the home page is successful
```

```
        System.out.println("Successfully navigated to the home page");
```

```
    } else {
```

```
        // Navigation to the home page failed
```

```
        System.out.println("Failed to navigate to the home page");
```

```
    }
```

```
    // Remove the 'PendingException' or handle the logic here based on the verification
```

```
    // throw new io.cucumber.java.PendingException();
```

```
    }
```

```
}
```

Eg. Screenshot

```
Scenario: Check login is succesfull with valid credentials # src/test/resources/signup.feature:3  
  Given browser is open # cucumber_java.Loginsteps.browserIsOpen()  
  And user is on login page # cucumber_java.Loginsteps.userIsOnLoginPage()  
  When user enters email and password # cucumber_java.Loginsteps.userEntersUsernameAndPassword()  
  And User clicks on login # cucumber_java.Loginsteps.userClicksOnLogin()  
  Successfully navigated to the home page  
  Then user is navigated to the home page # cucumber_java.Loginsteps.userIsNavigatedToHomePage()  
  
1 Scenarios (1 passed)  
5 Steps (5 passed)  
0m7.715s
```

Test Report-

Test Case 1					
Project Name: Bag Emporium					
Login Test Case					
Test Case ID: 1			Test Designed By: Pranav P k		
Test Priority(Low/Medium/High):High			Test Designed Date: 03-12-2023		
Module Name: Login Screen			Test Executed By: Mr. Rony Tom		
Test Title: User Login			Test Execution Date: 04-12-2023		
Description: Verify login with valid email and password					
Pre-Condition: User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Dashboard should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: pranavpk2024b@mca.ajce.in	User should be able to login	User Logged in and navigated to User Dashboard	Pass
3	Provide Valid Password	Password: Pranav@2002			
4	Click on Login button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database					

Test Case 2: Search

```

package cucumber_java;

import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.cucumber.java.en.Given;

```

```
import io.cucumber.java.en.When;

import io.cucumber.java.en.Then;

import static org.junit.Assert.assertEquals;

import static org.junit.Assert.assertTrue;

public class search {

    WebDriver driver;

    @Given("the user is on the login page")

    public void userIsOnLoginPage() {

        openBrowser();

        navigateToLoginPage();

    }

    @When("the user logs in with email {string} and password {string}")

    public void userLogsIn(String email, String password) {

        enterCredentials(email, password);

        clickLoginButton();

    }

    @Then("the user is redirected to the homepage")

    public void userIsRedirectedToHomepage() {

        // Verify navigation to the homepage by checking the URL or elements on the page

        String homepageURL = "http://127.0.0.1:8000/myauth/home/";

        assertTrue(driver.getCurrentUrl().startsWith(homepageURL));

    }

    @When("the user clicks on the search icon on the homepage")

    public void userClicksOnSearchIcon() {

        WebElement searchIcon = driver.findElement(By.cssSelector("a.nav-icon.d-none.d-lg-inline"));

    }
```

```
        searchIcon.click();
    }

    @Then("the user is redirected to seach page")
    public void the_user_is_redirected_to_seach_page() {
        String expectedSearchPageURL = "http://127.0.0.1:8000/myauth/search/";
        String actualURL = driver.getCurrentUrl();
        assertEquals(expectedSearchPageURL, actualURL);
    }

    @When("the user enters {string} in the search input and press submit")
    public void userEntersSearchTermAndSubmit(String searchTerm) {
        WebElement searchInput = driver.findElement(By.id("search-input"));
        searchInput.sendKeys(searchTerm);

        // Press submit (Enter key)
        searchInput.sendKeys(Keys.ENTER);
    }

    @When("the user clicks on the search button")
    public void userClicksOnSearchButton() {
        WebElement searchButton = driver.findElement(By.id("search"));
        searchButton.click();
    }

    @Then("the user is redirected to the search result page")
    public void userIsRedirectedToSearchResultPage() {
        // Verify navigation to the search result page by checking the URL
        String searchResultPageURL = "http://127.0.0.1:8000/myauth/search/?search_term=women";
        assertTrue(driver.getCurrentUrl().startsWith(searchResultPageURL));
    }

    private void openBrowser() {
```

```
        System.setProperty("webdriver.gecko.driver",
"C:\\Users\\Lenovo\\Desktop\\geckodriver.exe");

        driver = new FirefoxDriver();

        driver.manage().window().maximize();

    }
    private void navigateToLoginPage() {

        String loginPageURL = "http://127.0.0.1:8000/myauth/login/";

        driver.get(loginPageURL);

    }

    private void enterCredentials(String email, String password) {

        WebElement emailField = driver.findElement(By.id("email"));

        WebElement passwordField = driver.findElement(By.id("password"));

        emailField.sendKeys(email);

        passwordField.sendKeys(password);

    }

    private void clickLoginButton() {

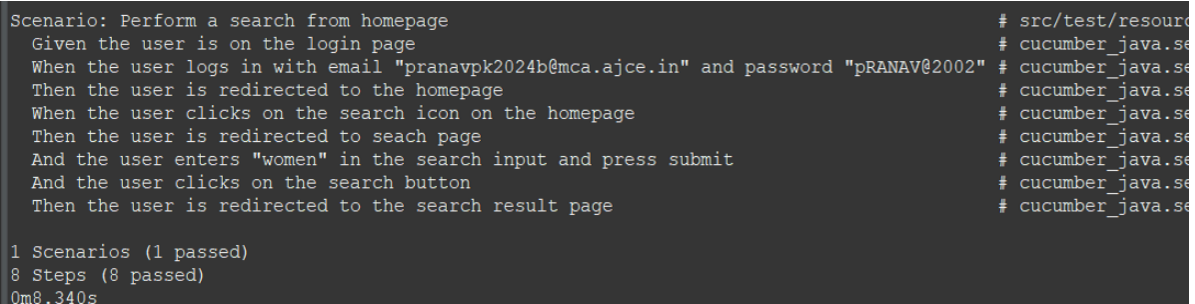
        WebElement loginButton = driver.findElement(By.id("submit"));

        loginButton.click();

    }

}
```

Eg. Screenshot



```
Scenario: Perform a search from homepage # src/test/resourc
  Given the user is on the login page # cucumber_java.se
  When the user logs in with email "pranavpk2024b@mca.ajce.in" and password "pRANAV@2002" # cucumber_java.se
  Then the user is redirected to the homepage # cucumber_java.se
  When the user clicks on the search icon on the homepage # cucumber_java.se
  Then the user is redirected to seach page # cucumber_java.se
  And the user enters "women" in the search input and press submit # cucumber_java.se
  And the user clicks on the search button # cucumber_java.se
  Then the user is redirected to the search result page # cucumber_java.se

1 Scenarios (1 passed)
8 Steps (8 passed)
0m8.340s
```

Test Report-

Test Case 2					
Project Name: Bag Emporium					
Search					
Test Case ID: Test_3			Test Designed By: Pranav P k		
Test Priority (Low/Medium/High): High			Test Designed Date: 03-12-2023		
Module Name: Search			Test Executed By : Mr. Rony Tom		
Test Title : Search provider			Test Execution Date: 04-12-2023		
Description: User Search I					
Pre-Condition :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page displayed	Pass
2	Provide valid Email	Email:pranavpk2024b@mca.a	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: Pranav@2002			
4.	User clicks on search icon		Search bar should be displayed	Search bar is displayed	
5	Provide the search deatails	Search:benzelectrical			
6	User clicks on Submit		Search result should be displayed	Search result should is displayed	Pass
Post-Condition: The search functionality successfully displays relevant results based on the provided search details while ensuring the user's continued logged-in session for further interaction with the system.					

Test Case 3: Update Profile

Code-

```
package cucumber_java;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;
import java.util.concurrent.TimeoutException;
import org.junit.Assert;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.When;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.And;
public class updateprofile {
    WebDriver driver;
    WebDriverWait wait;
    @Given("the browser is open")
    public void browserIsOpen() {
        System.setProperty("webdriver.gecko.driver",
"C:\\\\Users\\\\Lenovo\\\\Desktop\\\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.manage().window().maximize();
        wait = new WebDriverWait(driver, Duration.ofSeconds(10));
    }
    @Given("the user is on the login page")
    public void userIsOnLoginPage() {
        String loginPageURL = "http://127.0.0.1:8000/myauth/login/";
        driver.get(loginPageURL);
    }
    @When("the user enters their username and password")
    public void userEntersUsernameAndPassword() {
        WebElement usernameField = driver.findElement(By.id("email"));
        WebElement passwordField = driver.findElement(By.id("password"));
        usernameField.sendKeys("prnavpk2024b@mca.ajce.in"); // Replace with the correct username
        passwordField.sendKeys("pRANAV@2002"); // Replace with the correct password
    }
    @When("the user clicks on the login button")
    public void userClicksOnLogin() {
        WebElement loginButton = driver.findElement(By.id("submit"));
        loginButton.click();
    }
}
```



```

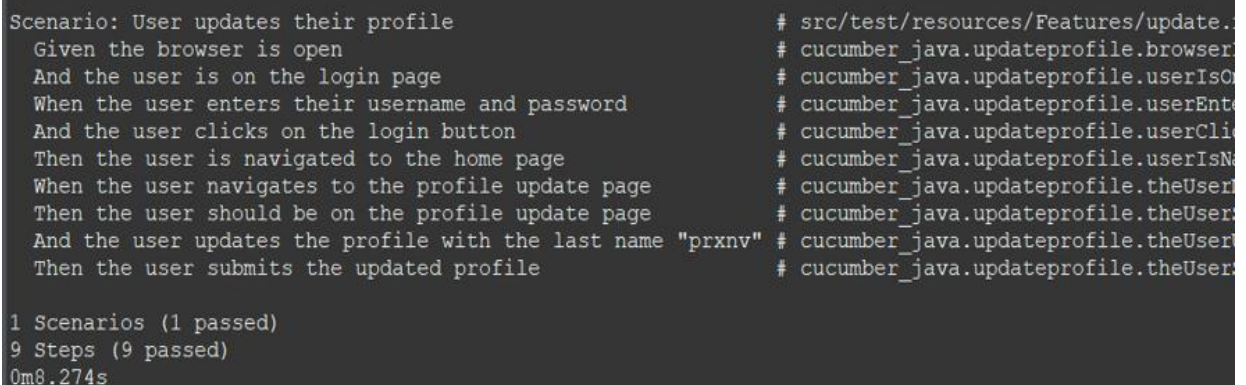
}
@Then("the user is navigated to the home page")
public void userIsNavigatedToHomePage() {
    String homePageURL = driver.getCurrentUrl();
    String expectedHomePageURL = "http://127.0.0.1:8000/myauth/home/";
    Assert.assertEquals(expectedHomePageURL, homePageURL);
}
@When("the user navigates to the profile update page")
public void theUserNavigatesToProfileUpdatePage() {
    WebElement updateProfileLink = driver.findElement(By.id("upd"));
    updateProfileLink.click();
}
@Then("the user should be on the profile update page")
public void theUserShouldBeOnProfileUpdatePage() {
    String expectedUrl = "http://127.0.0.1:8000/myauth/updateprofile/";
    String currentUrl = driver.getCurrentUrl();

    Assert.assertEquals(expectedUrl, currentUrl);
}
@And("the user updates the profile with the last name {string}")
public void theUserUpdatesProfileWithLastName(String newLastName) {
    WebElement lastNameField = driver.findElement(By.id("last_name"));
    lastNameField.clear();
    lastNameField.sendKeys(newLastName);
}
// Existing steps...
@Then("the user submits the updated profile")
public void theUserSubmitsUpdatedProfile() {

    WebElement updateButton = driver.findElement(By.id("submitbutton"));
    updateButton.click();
}
}

```

Screenshot



```

Scenario: User updates their profile                                     # src/test/resources/Features/update.
  Given the browser is open                                           # cucumber_java.updateprofile.browser
  And the user is on the login page                                   # cucumber_java.updateprofile.userIsO
  When the user enters their username and password                   # cucumber_java.updateprofile.userEnt
  And the user clicks on the login button                             # cucumber_java.updateprofile.userCli
  Then the user is navigated to the home page                         # cucumber_java.updateprofile.userIsN
  When the user navigates to the profile update page                 # cucumber_java.updateprofile.theUser
  Then the user should be on the profile update page                 # cucumber_java.updateprofile.theUser
  And the user updates the profile with the last name "prxnv"       # cucumber_java.updateprofile.theUser
  Then the user submits the updated profile                           # cucumber_java.updateprofile.theUser

1 Scenarios (1 passed)
9 Steps (9 passed)
0m8.274s

```

Test Report-**Test Case 3****Project Name:** Bag Emporium**Update Profile****Test Case ID:** Test_3**Test Designed By:** Pranav P k**Test Priority
(Low/Medium/High):** High**Test Designed Date:** 03-12-2023**Module Name:** Update Profile**Test Executed By :** Mr. Rony Tom**Test Title :** Update Profile**Test Execution Date:** 04-12-2023**Description:** User updated user profile details**Pre-Condition :** User has valid username and password

Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page displayed	Pass
2	Provide valid Email	email: pranavpk2024b@mc a.ajce.in	User should be able to login	navigated to corresponding Home Page Profile page should be displayed	Pass
3	Provide Valid Password	Password: pranav@2002			
4.	Navigation to Update profile Page		Profile page should displayed		
5	Provide update details	Last name: pk			
6	User update profile		Update successful message should be displayed	Update successful message is displayed	Pass

Post-Condition: The user's profile data has been successfully updated and that the system acknowledges the update with a relevant message. It also maintains the user's login session, allowing them to continue using the system with the updated details.

Test Case 4: Add To Cart**Code-**

```
package cucumber_java;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

import java.time.Duration;
import org.junit.After;
import org.junit.Assert;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.When;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.And;

public class addtocart {

    WebDriver driver;
    WebDriverWait wait;

    @Given("the browser is open")
    public void browserIsOpen() {
        System.setProperty("webdriver.gecko.driver",
"C:\\Users\\Lenovo\\OneDrive\\Desktop\\geckodriver.exe");
        driver = new FirefoxDriver();
        driver.manage().window().maximize();
        wait = new WebDriverWait(driver, Duration.ofSeconds(10));
    }

    @Given("the user is on the login page")
    public void userIsOnLoginPage() {
        String loginPageURL = "http://127.0.0.1:8000/myauth/login/";
        driver.get(loginPageURL);
    }

    @When("the user enters their username and password")
    public void userEntersUsernameAndPassword() {
        WebElement usernameField = driver.findElement(By.id("email"));
        WebElement passwordField = driver.findElement(By.id("password"));
        usernameField.sendKeys("pranavpk2024b@mca.ajce.in"); // Replace with the correct
username
        passwordField.sendKeys("pRANAV@2002"); // Replace with the correct password
    }
}
```

```
}

@When("the user clicks on the login button")
public void userClicksOnLogin() {
    WebElement loginButton = driver.findElement(By.id("submit"));
    loginButton.click();
}

@Then("the user is navigated to the home page")
public void userIsNavigatedToHomePage() {
    String homePageURL = driver.getCurrentUrl();
    String expectedHomePageURL = "http://127.0.0.1:8000/myauth/home/";

    Assert.assertEquals(expectedHomePageURL, homePageURL);
}

@And("the user clicks on the \"Add to Cart\" button")
public void userClicksOnAddToCartButton() {
    WebElement productLink = driver.findElement(By.id("cart"));

    // Scroll the element into view
    ((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView(true);",
productLink);

    // Click on the element
    productLink.click();
}

@Then("the user is redirected to the shopping cart page")
public void userIsRedirectedToShoppingCartPage() {
    wait.until(ExpectedConditions.urlContains("/view_cart/"));
    String expectedCartPageURL = "http://127.0.0.1:8000/myauth/view_cart/";
    String actualCartPageURL = driver.getCurrentUrl();

    Assert.assertEquals(expectedCartPageURL, actualCartPageURL);

    // Add additional assertions or verifications for the shopping cart page if needed.
}

// Close the browser after the test completes
@After
public void closeBrowser() {
    if (driver != null) {
        driver.quit();
    }
}
}
```

Screenshot

```

Scenario: User adds a product to the cart and is redirected # src/test/resources/Features/addtocart.feature:3
  Given the browser is open                               # cucumber_java.addtocart.browserIsOpen()
  And the user is on the login page                         # cucumber_java.addtocart.userIsOnLoginPage()
  When the user enters their username and password          # cucumber_java.addtocart.userEntersUsernameAndPassword()
  And the user clicks on the login button                  # cucumber_java.addtocart.userClicksOnLogin()
  Then the user is navigated to the home page              # cucumber_java.addtocart.userIsNavigatedToHomePage()
  And the user clicks on the "Add to Cart" button          # cucumber_java.addtocart.userClicksOnAddToCartButton()
  Then the user is redirected to the shopping cart page    # cucumber_java.addtocart.userIsRedirectedToShoppingCartPage()

1 Scenarios (1 passed)
7 Steps (7 passed)
0m5.588s

```

Test Report-

Test Case 4

Project Name: Bag Emporium					
Add To Cart					
Test Case ID: Test_4			Test Designed By: Pranav P k		
Test Priority (Low/Medium/High): High			Test Designed Date: 03-12-2023		
Module Name: Add To Cart			Test Executed By : Mr. Rony Tom		
Test Title : Add To Cart			Test Execution Date: 04-12-2023		
Description: Product Added to the cart.					
Pre-Condition :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page displayed	Pass
2	Provide valid Email	email: pranavpk2024b@mc a.ajce.in	User should be able to login	navigated to corresponding Home Page Profile page should be displayed	Pass
3	Provide Valid Password	Password: pranav@2002			
4.	User click the product to add to cart		Product added to the add to cart page		Pass
Post-Condition: The user's added product to the cart and perform checkout function. It also maintains the user's login session, allowingthem to continue purchasing items.					

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

The implementation phase of a project is where the design is transformed into a functional system. It is a crucial stage in ensuring the success of the new system, as it requires gaining user confidence that the system will work effectively and accurately. User training and documentation are key concerns during this phase. Conversion may occur concurrently with user training or at a later stage. Implementation involves the conversion of a newly revised system design into an operational system.

During this stage, the user department bears the primary workload, experiences the most significant upheaval, and has the most substantial impact on the existing system. Poorly planned or controlled implementation can cause confusion and chaos. Whether the new system is entirely new, replaces an existing manual or automated system, or modifies an existing system, proper implementation is essential to meet the organization's needs. System implementation involves all activities required to convert from the old to the new system. The system can only be implemented after thorough testing is done and found to be working according to specifications. System personnel evaluate the feasibility of the system. Implementation requires extensive effort in three main areas: education and training, system testing, and changeover. The implementation phase involves careful planning, investigating system and constraints, and designing methods to achieve changeover

6.2 IMPLEMENTATION PROCEDURES

Software implementation is the process of installing the software in its actual environment and ensuring that it satisfies the intended use and operates as expected. In some organizations, the software development project may be commissioned by someone who will not be using the software themselves. During the initial stages, there may be doubts about the software, but it's important to ensure that resistance does not build up.

This can be achieved by:

- Ensuring that active users are aware of the benefits of the new system, building their confidence in the software.
- Providing proper guidance to the users so that they are comfortable using the application. Before viewing the system, users should know that the server program must be running on the server. Without the server object up and running, the intended process will not take place.

6.2.1 USER TRAINING

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database, and call up routine that will produce reports and perform other necessary functions.

6.2.2 TRAINING ON THE APPLICATION SOFTWARE

After providing the necessary basic training on computer awareness, it is essential to provide training on the new application software to the user. This training should include the underlying philosophy of using the new system, such as the flow of screens, screen design, the type of help available on the screen, the types of errors that may occur while entering data, and the corresponding validation checks for each entry, and ways to correct the data entered. Additionally, the training should cover information specific to the user or group, which is necessary to use the system or part of the system effectively. It is important to note that this training may differ across different user groups and levels of hierarchy.

6.2.3 SYSTEM MAINTENANCE

The maintenance phase is a crucial aspect of the software development cycle, as it is the time when the software is actually put to use and performs its intended functions. Proper maintenance is essential to ensure that the system remains functional, reliable, and adaptable to changes in the system environment. Maintenance activities go beyond simply identifying and fixing errors or bugs in the system. It may involve updates to the software, modifications to its functionalities, and enhancements to its performance, among other things. In essence, software maintenance is an ongoing process that requires continuous monitoring, evaluation, and improvement of the system to meet changing user needs and requirements.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The "Bag Emporium" project presents a comprehensive e-commerce platform tailored for the sale of bags. With a focus on providing a seamless and user-friendly experience, the platform caters to a diverse range of users, including customers, sellers, and administrators. Noteworthy features such as a wide variety of bags, a user-friendly interface, secure payment options, and innovative functionalities for both customers and administrators distinguish the platform in the competitive market. The integration of advanced technologies like recommendation systems, NLP for search optimization, and AI-powered chatbots enhances the overall shopping experience. The involvement of stakeholders, including customers, sellers, and administrators, is well-defined, ensuring a collaborative and efficient operation of the platform. The project aligns with industry trends and user preferences, offering a unique value proposition through a combination of diverse product offerings, personalized services, and continuous adaptation to market dynamics. Overall, the "Online Bag Shop" project is poised to make a significant impact in the e-commerce industry by addressing user needs, fostering engagement, and staying attuned to market trends.

7.2 FUTURE SCOPE

These future scopes aim to propel the "Bag Emporium" project into new dimensions, embracing innovation, sustainability, and customer-centric strategies for long-term success in the dynamic e-commerce landscape. Some potential areas for development and growth include:

- **Global Expansion:** Explore opportunities for international market expansion, catering to a broader audience and leveraging global trends in the bag and fashion industry.
- **Smart technology Integration:** Incorporate emerging technologies like augmented reality for virtual try-ons, enhancing the online shopping experience and bridging the gap between online and offline retail.
- **Sustainability Initiatives:** Introduce a dedicated section for eco-friendly and sustainable bags, aligning with the growing demand for environmentally conscious products and contributing to corporate social responsibility.
- **Collaborations with Designers:** Collaborate with renowned designers for exclusive bag collections, elevating the platform's brand image and attracting customers seeking unique and premium offerings.
- **User-Generated Content:** Implement features that encourage users to share their bag experiences through photos, reviews, and social media integration, fostering a community-driven platform and enhancing brand visibility.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009.
- Lisa Crispin, Janet Gregory, "Agile Testing: A Practical Guide for Testers and Agile Teams, Addison Wesley Professional, 1st Edition (2008) - Guiding the integration of agile testing. practices, ensuring continuous testing and quality assurance throughout Urban Scout's development lifecycle.
- Google Support – How to search on Google: Tips on effective Google Searches, Including finding quick answer.
- Roger S Pressman, “Software Engineering”.
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.flipkart.com
- www.shoppersstop.com
- www.bootstrap.com

CHAPTER 9

APPENDIX

9.1 SAMPLE CODE

LOGIN

```
{% extends 'nav.html' % }
{% block title % }
AispaceX Login
{% endblock title % }
{% block style % }
<style>
    .btn{
        font-family: "Raleway",sans-serif;
        font-weight: 500;
        font-size: 16px;
        letter-spacing: 1px;
        display: inline-block;
        padding: 10px 30px;
        border-radius: 50px;
        transition: 0.5s;
        color: #3498db;
        border: 2px solid #3498db;
    }
    .btn:hover{
        background-color: rgb(57, 230, 22);
        color: white;
    }
</style>
{% endblock style % }
{% block form % }
<div class="container mt-5">
    <div class="row justify-content-center">
        <div class="col-md-6">
            <div class="card">
                <div class="card-header">
                    <center><h4>Login</h4></center>
                </div>
                <div class="card-body">

                    {% for message in messages % }

                        <div class="alert alert-{{ message.tags }} alert-dismissible fade show"
role="alert">
```

```

        <strong></strong>{ { message } }
        <button type="button" class="btn-close" data-bs-dismiss="alert" aria-
label="Close"></button>
    </div>
    { % endfor % }

    <form action="http://127.0.0.1:8000/myauth/login/" method="POST"
id="Login-form">
        { % csrf_token % }
        <br>

        <!-- Email with spacing -->
        <div class="form-group">
            <input type="email" class="form-control mb-3" placeholder="Email"
name="email" id="email">
            <div class="text-danger" id="email_error"></div>
        </div>
        <!-- Password with spacing -->
        <div class="form-group">
            <input type="password" class="form-control mb-3"
placeholder="Password" name="password" id="password" >
            <div class="text-danger" id="password_error"></div>
        </div>

        <div class="text-center">
            <button type="submit" class="btn" id="submit">Login</button>
        </div>
    </form>
    <div class="mt-3 text-center">
        <hr>
        Not Already a user? <a href="/myauth/signup/">SignUp</a><br>
        <span>Forget Password? <a href="{ % url 'password_reset'
% } ">Reset</a></span>
    </div>
</div>
</div>
</div>
</div>
</div>
{ % endblock form % }

```

Home

```
{ % load static % }
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Customer Home Page</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="apple-touch-icon" href="{ % static 'assets/img/apple-icon.png' % }">
  <link rel="shortcut icon" type="image/x-icon" href="{ % static 'assets/img/favicon.ico' % }">
  <link rel="stylesheet" href="{ % static 'assets/css/bootstrap.min.css' % }">
  <link rel="stylesheet" href="{ % static 'assets/css/templatemo.css' % }">
  <link rel="stylesheet" href="{ % static 'assets/css/custom.css' % }">
  <!-- Load fonts style after rendering the layout styles -->
  <link rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Roboto:wght@100;200;300;400;500;700;900&dis
play=swap">
  <link rel="stylesheet" href="{ % static 'assets/css/fontawesome.min.css' % }">
  <!--
https://templatemo.com/tm-559-zay-shop
-->
</head>

<body>
  <!-- Start Top Nav -->
  <nav class="navbar navbar-expand-lg bg-dark navbar-light d-none d-lg-block"
id="templatemo_nav_top">
    <div class="container text-light">
      <div class="w-100 d-flex justify-content-between">
        <div>
          <i class="fa fa-envelope mx-2"></i>
          <a class="navbar-sm-brand text-light text-decoration-none"
href="mailto:info@company.com">info@company.com</a>
          <i class="fa fa-phone mx-2"></i>
          <a class="navbar-sm-brand text-light text-decoration-none" href="tel:010-020-
0340">010-020-0340</a>
        </div>
        <div>
          <a class="text-light" href="https://fb.com/templatemo" target="_blank"
rel="sponsored"><i class="fab fa-facebook-f fa-sm fa-fw me-2"></i></a>
          <a class="text-light" href="https://www.instagram.com/" target="_blank"><i
class="fab fa-instagram fa-sm fa-fw me-2"></i></a>
          <a class="text-light" href="https://twitter.com/" target="_blank"><i class="fab fa-
twitter fa-sm fa-fw me-2"></i></a>
          <a class="text-light" href="https://www.linkedin.com/" target="_blank"><i class="fab
fa-linkedin fa-sm fa-fw"></i></a>
        </div>
      </div>
    </div>
  </div>
</body>
```



```

</nav>
<!-- Close Top Nav -->
<!-- Header -->
<nav class="navbar navbar-expand-lg navbar-light shadow">
  <div class="container d-flex justify-content-between align-items-center">
    
    <a class="navbar-brand text-success logo h1 align-self-center" href="index.html">
    </a>
    <button class="navbar-toggler border-0" type="button" data-bs-toggle="collapse" data-bs-
target="#templatemo_main_nav" aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="align-self-center collapse navbar-collapse flex-fill d-lg-flex justify-content-lg-
between" id="templatemo_main_nav">
    <div class="flex-fill">
      <ul class="nav navbar-nav d-flex justify-content-between mx-lg-auto">
        <li class="nav-item">
          <a class="nav-link" href="/myauth/home">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#about">About</a>
        </li>
        <li class="nav-item">
          <a
                                                                    class="nav-link"
href="http://127.0.0.1:8000/myauth/productallview/">Shop</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#contact">Contact</a>
        </li>
        <!--<li class="nav-item">
          <a class="nav-link" href="/myauth/logout/">Logout</a>
        </li>-->

        <!-- <li class="nav-item">
          <a class="nav-link" href="contact.html">Welcome{ { user.username} }</a>

        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            Profile
          </a>
          <div class="dropdown-menu" aria-labelledby="navbarDropdown">
            <a class="dropdown-item" href="">Signup</a>
            <a class="dropdown-item" href=""></a>
            <a class="dropdown-item" href="/myauth/logout/">Logout</a>
          </div>
        </li>-->
        <!--<li class="nav-item dropdown">

```

```

        <a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button"
data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            Welcome {{ user.username }}
        </a>
        <div class="dropdown-menu" aria-labelledby="userDropdown">
            <a class="dropdown-item" href="#">Profile</a>
            <a class="dropdown-item" href="/myauth/logout/">Logout</a>
        </div>

    </li>-->
</div>
<div class="navbar align-self-center d-flex">
    <div class="d-lg-none flex-sm-fill mt-3 mb-4 col-7 col-sm-auto pr-3">
        <div class="input-group">
            <input type="text" class="form-control" id="inputMobileSearch"
placeholder="Search ...">
            <div class="input-group-text">
                <i class="fa fa-fw fa-search"></i>
            </div>
        </div>
    </div>
    <div>
        <a class="nav-icon d-none d-lg-inline" href="/myauth/search/" >
            <i class="fa fa-fw fa-search text-dark mr-2"></i>
        </a>
        <a class="nav-icon position-relative text-decoration-none" href="/myauth/view_cart/">
            <i class="fa fa-fw fa-cart-arrow-down text-dark mr-1"></i>
            <span class="position-absolute top-0 left-100 translate-middle badge rounded-pill
bg-light text-dark">
                {{ cart_items_count }}
            </span>
        </a>
    <!-- <li class="nav-item dropdown">

        <a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button"
data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            {{ user.username }}
        </a>
        <div class="dropdown-menu" aria-labelledby="userDropdown">
            <a class="dropdown-item" href="#">Profile</a>
            <a class="dropdown-item" href="/myauth/logout/">Logout</a>
        </div>-->

    </li>
    <!--<a class="nav-icon position-relative text-decoration-none" href="#">
        <i class="fa fa-fw fa-user text-dark mr-3"></i>
        <span class="position-absolute top-0 left-100 translate-middle badge rounded-pill
bg-light text-dark">+99</span>
    </a>-->
</div>
</div>

```

```
</div>
</nav>
<!-- Close Header -->

<!--demo header-->
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Website Header</title>
  <style>
    /* Header styles */
    header {
      background-color: #2c621d;
      color: #fff;
      padding: 10px 0;
    }

    .container {
      max-width: 1200px;
      margin: 0 auto;
      display: flex;
      justify-content: space-between;
      align-items: center;
    }

    /* Navigation styles */
    nav ul {
      list-style: none;
      margin: 0;
      padding: 0;
      display: flex;
    }

    nav ul li {
      margin-right: 20px;
    }

    nav ul li:last-child {
      margin-right: 0;
    }

    nav a {
      text-decoration: none;
      color: #fff;
      font-weight: bold;
    }
  </style>
</head>
```

```
.dropdown {
  position: relative;
}

/* Dropdown styles */
.dropdown {
  position: relative;
}

.dropdown-content {
  display: none;
  position: absolute;
  background-color: #333;
  min-width: 120px;
  z-index: 1;
}

.dropdown:hover .dropdown-content {
  display: block;
}

.dropdown-content a {
  padding: 10px;
  text-align: center;
  color: #fff;
  text-decoration: none;
  display: block;
}

/* Add the following styles for hover effect */
.dropdown-content a::before {
  content: "";
  position: absolute;
  bottom: 0;
  left: 0;
  width: 0;
  height: 3px;
  background-color: green;
  transition: width 0.3s ease;
}

.dropdown-content a:hover::before {
  width: 100%; /* Expand the width on hover */
}

.user {
  display: flex;
  align-items: center;
  margin-left: auto; /* Align to the right */
}
```

```
.user img {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  margin-right: 10px; /* Add some space between image and text */
}
header {
  background-color: #2c621d;
  color: #fff;
  padding: 10px 0;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.back {
  padding: 10px;
  display: flex; /* Use flex to align items horizontally */
}

.back ul {
  list-style: none;
  margin: 0;
  padding: 0;
  display: flex;
}

.back ul li {
  margin-right: 20px;
}

.back ul li:last-child {
  margin-right: 0;
}

.back a {
  text-decoration: none;
  color: #fff;
  font-weight: bold;
}

.user.dropdown {
  display: flex;
  align-items: center;
```

```
        margin-left: auto;
    }

    .user.dropdown a {
        display: flex;
        align-items: center;
        color: #fff;
        text-decoration: none;
    }

    .user.dropdown img {
        width: 40px;
        height: 40px;
        border-radius: 50%;
        margin-right: 10px;
    }
/*
    .dropdown-content {
        display: none;
        position: absolute;
        background-color: #343a40;
        min-width: 120px;
        z-index: 1;
    }

    .user.dropdown:hover .dropdown-content {
        display: block;
    }

    .dropdown-content a {
        padding: 10px;
        text-align: center;
        color: #fff;
        text-decoration: none;
        display: block;
    }

    .dropdown-content a:hover {
        background-color: #555;
    }
*/
    .fw-bolder {
text-decoration: none;
color: #fff;
position: relative;
display: inline-block;
padding-bottom: 3px; /* Adjust padding to control the distance between text and underline */
box-sizing: border-box; /* Include border and padding in the total width and height */
transition: border-bottom 0.3s ease; /* Add transition for the underline effect */
    }
}
```

```

.fw-bolder:hover {
    border-bottom: 3px solid #fff; /* Add the underline style on hover */
}

</style>
</head>

<body>
    <header>
        <div class="container">
            <div class="back">
                <ul >
                    <li><a href="http://127.0.0.1:8000/myauth/productallview/" class="fw-bolder">All
Collections </a></li>
                    <li class="dropdown">
                        <div >

                            <a href="#" class="fw-bolder">Categories</a>
                        </div>
                        <div class="dropdown-content bg-light">
                            <!-- Categories loop -->
                            {% for category in categories %}
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/{{ category.id }}"
style="color:black;" class="fw-bolder">{{ category.category_name }}</a>
                                {% endfor %}
                            </div>
                        </li>
                        <li class="dropdown">
                            <a href="#" class="fw-bolder">Sub-categories</a>
                            <div class="dropdown-content bg-light">
                                <!-- Subcategories loop -->
                                {% for subcategory in subcategories %}
                                    <a href="http://127.0.0.1:8000/myauth/productssubcategorylist/{{ subcategory.id
}}}" style="color:black;" class="fw-bolder">{{ subcategory.sub_category_name }}</a>
                                    {% endfor %}
                                </div>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/4" class="fw-
bolder">Men</a>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/5" class="fw-
bolder">Women</a>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/6" class="fw-
bolder">Kids</a>
                            </li>
                </ul>
            </div>
        </div>
    </header>
</body>

```

```

        <li>
            <a href="#" class="fw-bolder">Latest Collections</a>
        </li>
    </ul>
</div>
<div class="user dropdown">
    <a href="#">
        {% if user.profile.profile_picture %}
        
        {% else %}
        
        {% endif %}
        Hello, {{ user.first_name }}
    </a>
    <div class="dropdown-content" style="margin-top: 215px;">
        <a href="/myauth/updateprofile/">Profile</a>
        <a href="/myauth/view_cart/">View Cart</a>
        <a href="/myauth/view_wishlist/">View Wishlist</a>
        <a href="/myauth/payment_history/">View History</a>
        <a href="/myauth/logout/">Logout</a>
    </div>
</div>
</div>
</header>
<!-- Your page content goes here -->
</body>

</html>
<!--demoheader close-->
<!-- Modal -->
<div class="modal fade bg-white" id="templatemo_search" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-lg" role="document">
        <div class="w-100 pt-1 mb-5 text-right">
            <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
        </div>
        <form action="" method="get" class="modal-content modal-body border-0 p-0">

            <div class="input-group mb-2">
                <input type="text" class="form-control" id="inputModalSearch" name="q"
placeholder="Search ...">
                <button type="submit" class="input-group-text bg-success text-light">
                    <i class="fa fa-fw fa-search text-white"></i>
                </button>
            </div>
        </form>
    </div>
</div>
<!-- Start Banner Hero -->

```



```

<div id="template-mo-zay-hero-carousel" class="carousel slide" data-bs-ride="carousel">
  <ol class="carousel-indicators">
    <li data-bs-target="#template-mo-zay-hero-carousel" data-bs-slide-to="0"
class="active"></li>
    <li data-bs-target="#template-mo-zay-hero-carousel" data-bs-slide-to="1"></li>
    <li data-bs-target="#template-mo-zay-hero-carousel" data-bs-slide-to="2"></li>
  </ol>
  <div class="carousel-inner">
    <div class="carousel-item active">
      <div class="container">
        <div class="row p-5">
          <div class="mx-auto col-md-8 col-lg-6 order-lg-last">
            
          </div>
          <div class="col-lg-6 mb-0 d-flex align-items-center">
            <div class="text-align-left align-self-center">
              <h1 class="h1 text-success"><b>BAGGAGE</b> Heaven</h1>
              <h3 class="h2">Bags Designed Beyond Imagination</h3>
            </div>
            <p>
              YOUR BAG SHOULD MAKE YOU FEEL GOOD AND EXPRESS
              YOUR PERSONALITY.
              EMBRACE YOUR NOMADIC SELF AND LIFT YOUR SPIRITS WITH
              A COLOURFUL HANDMADE BAG - STAND OUT IN THE CROWD
              WITH COOL BOHEMIAN VIBES!
            </p>
          </div>
        </div>
      </div>
    </div>
    <div class="carousel-item">
      <div class="container">
        <div class="row p-5">
          <div class="mx-auto col-md-8 col-lg-6 order-lg-last">
            
          </div>
          <div class="col-lg-6 mb-0 d-flex align-items-center">
            <div class="text-align-left">
              <div class="img">
                <h1 class="fw-bold">
                  FOUND WHO
                  IS<br>
                  RIGHT<br> TO
                  <br> TOUCH
                </h1>
                YOUR
                <br> SHOULDER
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

<style>
.img {
  font-weight: bolder;
  font-size: 16px;
  letter-spacing: 1px;

  color: #189b4a;
}
.img p{
  font-size: xx-large !important;
  font-family: 'Martian Mono', monospace;
}
</style>

<h1 class="h1">BEAUTIFY THE ELAGANCE</h1>
<h3 class="h2">With Quality Bags </h3>

</div>
</div>
</div>
</div>
</div>
<div class="carousel-item">
<div class="container">
<div class="row p-5">
<div class="mx-auto col-md-8 col-lg-6 order-lg-last">

</div>
<div class="col-lg-6 mb-0 d-flex align-items-center">
<div class="text-align-left">
<h1 class="h1">Bag Reflecting Your Style..</h1>
<h3 class="h2">Exclusive Collection of Stylish Bags </h3>
<p>

</p>
</div>
</div>
</div>
</div>
</div>
</div>
<div>
<a class="carousel-control-prev text-decoration-none w-auto ps-3" href="#template-mo-
zay-hero-carousel" role="button" data-bs-slide="prev">
<i class="fas fa-chevron-left"></i>
</a>
<a class="carousel-control-next text-decoration-none w-auto pe-3" href="#template-mo-
zay-hero-carousel" role="button" data-bs-slide="next">
<i class="fas fa-chevron-right"></i>
</a>

```

```
</div>
<!-- End Banner Hero -->
<div class="section">
  <a href="http://127.0.0.1:8000/myauth/productallview/" class="button">
    <span class="text">Shop Now</span>
    <span class="arrow">→</span>
  </a>
  <a href="http://127.0.0.1:8000/myauth/productallview/" class="button">
    <span class="text">Explore More</span>
    <span class="arrow">→</span>
  </a>
</div>
<style>
.section {
  display: flex;
  justify-content: space-between;
  background-color: #f0f4ef;
  padding: 10px 20px;
  border-radius: 5px; /* Add rounded corners to the section */
}

.button {
  text-align: center;
  background-color: #2c621d; /* Button background color */
  padding: 10px 20px; /* Button padding */
  border-radius: 5px; /* Add rounded corners to the buttons */
  text-decoration: none;
  color: #fff;
  font-weight: bold;
  display: flex;
  align-items: center;
  transition: background-color 0.3s;
}

.button:hover {
  background-color: #ff3300; /* Change background color on hover */
}

.text {
  margin-right: 10px; /* Add space between text and arrow */
}

.arrow {
  font-size: 18px; /* Increase arrow size */
}

/* Add more styles as needed */

</style>
```

```

<!--form-->

<!-- Start Categories of The Month -->
<!--<section class="container py-5">
  <div class="row text-center pt-3">
    <div class="col-lg-6 m-auto">
      <h1 class="h1">CATEGORIES </h1>
      <p>
        Find All Your Favourites Brands For Bags And LUGGAGE
      </p>
    </div>
  </div>
  <div class="row">
    <div class="col-12 col-md-4 p-5 mt-3">
      <a href="#"></a>
      <h5 class="text-center mt-3 mb-3">KIDS</h5>
      <p class="text-center"><a class="btn btn-success">Go Shop</a></p>
      <p>Discover our delightful collection of kids bags designed to add a playful touch to their
everyday adventures. These bags are not only adorable but also functional, perfect for carrying their
school essentials, toys, or snacks.
        With vibrant colors and cute designs, these bags are sure to bring smiles to your little
ones' faces.
      </p>
    </div>
    <div class="col-12 col-md-4 p-5 mt-3">
      <a href="#"></a>
      <h2 class="h5 text-center mt-3 mb-3">WOMEN</h2>
      <p class="text-center"><a class="btn btn-success">Go Shop</a></p>
      <p> Elevate your style with our exquisite range of women's bags. From elegant handbags
to chic totes, our collection is designed to complement every outfit and occasion. Crafted with
attention to detail and featuring a variety of colors and textures,
        our women's bags are the perfect blend of fashion and functionality.
      </p>
    </div>
    <div class="col-12 col-md-4 p-5 mt-3">
      <a href="#"></a>
      <h2 class="h5 text-center mt-3 mb-3">MEN</h2>
      <p class="text-center"><a class="btn btn-success">Go Shop</a></p>
      <p> Explore our sophisticated collection of men's bags that cater to the modern
gentleman's needs. Whether you're heading to the office, going on a weekend getaway, or hitting the
gym, we have a bag to suit your lifestyle. With sturdy materials and sleek designs,
        our men's bags offer both style and durability.
      </p>
    </div>
  </div>

```

```

</div>
</section>
End Categories of The Month -->

```

```

<section class="bg-light">
  <div class="container py-5">
    <div class="row text-center py-3">
      <div class="col-lg-6 m-auto">
        <h1 class="h1"></h1>
        <p>

          </p>
        </div>
      <style>
        .head {
          font-size: 24px; /* Increase font size for emphasis */
          margin: 10px 0;
          color: #459a37; /* Set the font color */
          font-family: 'Your Preferred Font', sans-serif; /* Choose a beautiful font */
          text-transform: uppercase; /* Convert text to uppercase */
          text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2); /* Add a subtle text shadow */
          letter-spacing: 2px; /* Adjust letter spacing */
          line-height: 1.2; /* Adjust line height for readability */
          text-align: center; /* Center-align the text */
        }

      </style>
    </div>
    <div class="row">
      <div class="head">

        <center><h1 class="h1">FEATURED PRODUCTS</h1></center>
        </div>
        <br><br>

        {% for item in items %}
        <div class="col-12 col-md-4 mb-4">
          <div class="card h-100">

            <a href="{% url 'prodetailview' item.id %}?subcat_id={{ item.sub_category_id_id
            }}">

              
            </a>

```

```

<div class="card-body">
  <ul class="list-unstyled d-flex justify-content-between">
    <li>
      <i class="text-warning fa fa-star"></i>
      <i class="text-warning fa fa-star"></i>
      <i class="text-warning fa fa-star"></i>
      <i class="text-muted fa fa-star"></i>
      <i class="text-muted fa fa-star"></i>
    </li>
    <li class="text-muted text-right">₹ {{item.current_price}}</li>
  </ul>
  <a href="shop-single.html" class="h2 text-decoration-none text-
dark">{{item.product_name}}</a>
  <p class="card-text">
    <!-- {{item.about_product}} -->
  </p>
  <p class="text-muted">Reviews (24)</p>
  <div class="product-actions">
    <a href="{% url 'add_to_cart' item.id %}"><button class="add-to-cart">Add to
Cart</button></a>

    <a href="{% url 'add_to_wishlist' item.id %}"><button class="wishlist">Add to
Wishlist</button></a>
  </div>
</div>
</div>
</div>
{% endfor %}

<style>
  .product-actions {
    display: flex;
    justify-content: center;
    gap: 10px;
    margin-top: 20px;
  }
  .add-to-cart,
  .wishlist,
  .buy-now-button {
    background-color: #ff6600;
    color: #fff;
    border: none;
    padding: 10px 20px;
    font-size: 18px;
    cursor: pointer;
    transition: background-color 0.3s;
  }
  .add-to-cart:hover,
  .wishlist:hover,
  .buy-now-button:hover {

```

```

background-color: #21b721;
}
    </style>

</div>
</div>

<div class="head">

    <center><h1 class="h1">CATEGORIES</h1></center>
    </div><br><br>

</section>
</section>
    <!-- <section> close =====-->
    <!-- =====-->

<!-- Start Categories of The Month -->
<section class="container py-5">

    <div class="row text-center pt-3">
        <!--<div class="col-lg-6 m-auto">
            <h1 class="h1">CATEGORIES </h1>
            <p>
                Find All Your Favourites Brands For Bags And LUGGAGE
            </p>
        </div>-->
    </div>

    <div class="row">
        <div class="col-12 col-md-4 p-5 mt-3">
            <a href="http://127.0.0.1:8000/myauth/productscategorylist/6">
                
            </a>
            <h5 class="text-center mt-3 mb-3">KIDS</h5>
            <p class="text-center">
                <a href="http://127.0.0.1:8000/myauth/productscategorylist/6" class="btn btn-
success">Go Shop</a>
            </p>
            <p>
                Discover our delightful collection of kids bags designed to add a playful touch to their
                everyday adventures. These bags are not only adorable but also functional, perfect for carrying their
                school essentials, toys, or snacks. With vibrant colors and cute designs, these bags are sure to bring
                smiles to your little ones' faces.
            </p>
        </div>

        <div class="col-12 col-md-4 p-5 mt-3">

```

```

        <a href="http://127.0.0.1:8000/myauth/productscategorylist/5"></a>
        <h2 class="h5 text-center mt-3 mb-3">WOMEN</h2>
        <p class="text-center"><a href="http://127.0.0.1:8000/myauth/productscategorylist/5"
class="btn btn-success">Go Shop</a></p>
        <p> Elevate your style with our exquisite range of women's bags. From elegant handbags
to chic totes, our collection is designed to complement every outfit and occasion. Crafted with
attention to detail and featuring a variety of colors and textures,
        our women's bags are the perfect blend of fashion and functionality.

        </p>
    </div>
    <div class="col-12 col-md-4 p-5 mt-3">
        <a href="http://127.0.0.1:8000/myauth/productscategorylist/4"></a>
        <h2 class="h5 text-center mt-3 mb-3">MEN</h2>
        <p class="text-center"><a href="http://127.0.0.1:8000/myauth/productscategorylist/4"
class="btn btn-success">Go Shop</a></p>
        <p> Explore our sophisticated collection of men's bags that cater to the modern
gentleman's needs. Whether you're heading to the office, going on a weekend getaway, or hitting the
gym, we have a bag to suit your lifestyle. With sturdy materials and sleek designs,
        our men's bags offer both style and durability.

        </p>
    </div>
</div>
</section>
<!-- End Categories of The Month -->

<!-- ===== Clients Section ===== -->
<section id="clients" class="clients section-bg">
    <div class="container" data-aos="zoom-in">

        <div class="row">

            <div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
                
            </div>

            <div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
                
            </div>

            <div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
                
            </div>

            <div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
                
            </div>

```



```

</div>

<div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
  
</div>

<div class="col-lg-2 col-md-4 col-6 d-flex align-items-center justify-content-center">
  
</div>

</div>

</div>
</section><!-- End Clients Section -->

<!--start about-->

<section id="about" class="about">
  <div class="container" data-aos="fade-up">

    <div class="row">
      <div class="col-lg-6" data-aos="fade-right" data-aos-delay="100">
        
        <br><br>
        <!--<div class="section-title">
          <h2></h2>
          <h3><span>ABOUT US</span></h3>
          <p>Welcome to the official website of bag Emporium.</p>
        </div>-->

      </div>
      <div class="col-lg-6 pt-4 pt-lg-0 content d-flex flex-column justify-content-center" data-
aos="fade-up" data-aos-delay="100">
        <h3></h3>
        <h3><span>ABOUT US</span></h3>
        <p>Welcome to the official website of bag Emporium.</p>
        <br>

        <p class="fst-italic">
          Welcome to Bag Emporium, your ultimate destination for bags !
          At Bag Emporium, we're more than just a store; we're your partner in style, convenience,
and self-expression.
        </p>
        <ul>
          <li>
            <i class="bx bx-store-alt"></i>
            <div>
              <h5></h5>
              <p>We take pride in our extensive collection of bags, curated to suit every occasion and

```

purpose. From trendy clutches for elegant evenings to rugged backpacks for adventurous journeys, we have it all under one roof.

Our commitment to quality is unwavering. Each bag in our inventory undergoes rigorous quality checks to ensure it meets our high standards.

We source materials from trusted suppliers to create durable, long-lasting products.

We believe that fashion should be accessible to everyone. That's why we offer competitive prices without compromising on quality. You can indulge in the latest trends without breaking the bank.

BAG EMPORIUM

Products

- Trolley Bags
- Duffle Bags
- Duffle Trolley
- Briefcase
- Backpacks
- Rucksacks

```

<div class="col-md-4 pt-5">
  <h2 class="h2 text-light border-bottom pb-3 border-light">Further Info</h2>
  <ul class="list-unstyled text-light footer-link-list">
    <li><a class="text-decoration-none" href="#">Home</a></li>
    <li><a class="text-decoration-none" href="#">About Us</a></li>
    <li><a class="text-decoration-none" href="#">Shop</a></li>
    <!--<li><a class="text-decoration-none" href="#">FAQs</a></li>-->
    <li><a class="text-decoration-none" href="#">Contact</a></li>
  </ul>
</div>

</div>

<div class="row text-light mb-4">
  <div class="col-12 mb-3">
    <div class="w-100 my-3 border-top border-light"></div>
  </div>
  <!-- <div class="col-auto me-auto">
    <ul class="list-inline text-left footer-icons">
      <li class="list-inline-item border border-light rounded-circle text-center">
        <a class="text-light text-decoration-none" target="_blank"
href="http://facebook.com/"><i class="fab fa-facebook-f fa-lg fa-fw"></i></a>
      </li>
      <li class="list-inline-item border border-light rounded-circle text-center">
        <a class="text-light text-decoration-none" target="_blank"
href="https://www.instagram.com/"><i class="fab fa-instagram fa-lg fa-fw"></i></a>
      </li>
      <li class="list-inline-item border border-light rounded-circle text-center">
        <a class="text-light text-decoration-none" target="_blank"
href="https://twitter.com/"><i class="fab fa-twitter fa-lg fa-fw"></i></a>
      </li>
      <li class="list-inline-item border border-light rounded-circle text-center">
        <a class="text-light text-decoration-none" target="_blank"
href="https://www.linkedin.com/"><i class="fab fa-linkedin fa-lg fa-fw"></i></a>
      </li>
    </ul>
  </div>-->
  <!-- <div class="col-auto">
    <label class="sr-only" for="subscribeEmail">Email address</label>
    <div class="input-group mb-2">
      <input type="text" class="form-control bg-dark border-light" id="subscribeEmail"
placeholder="Email address">
      <div class="input-group-text btn-success text-light">Subscribe</div>
    </div>
  </div>
</div> -->
</div>

</footer>
<!-- End Footer -->

```

```

<!-- Start Script -->
<script src="{ % static 'assets/js/jquery-1.11.0.min.js' % }"></script>
<script src="{ % static 'assets/js/jquery-migrate-1.2.1.min.js' % }"></script>
<script src="{ % static 'assets/js/bootstrap.bundle.min.js' % }"></script>
<script src="{ % static 'assets/js/templatemo.js' % }"></script>
<script src="{ % static 'assets/js/custom.js' % }"></script>
<!-- End Script -->
</body>

</html>

```

Add to Cart

```

{ % load static % }
<!DOCTYPE html>
<html lang="en">

<head>

    <title>Customer Home Page</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="apple-touch-icon" href="{ % static 'assets/img/apple-icon.png' % }">
    <link rel="shortcut icon" type="image/x-icon" href="{ % static 'assets/img/favicon.ico' % }">

    <link rel="stylesheet" href="{ % static 'assets/css/bootstrap.min.css' % }">
    <link rel="stylesheet" href="{ % static 'assets/css/templatemo.css' % }">
    <link rel="stylesheet" href="{ % static 'assets/css/custom.css' % }">

    <!-- Load fonts style after rendering the layout styles -->
    <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Roboto:wght@100;200;300;400;500;700;900&display=swap"
    <link rel="stylesheet" href="{ % static 'assets/css/fontawesome.min.css' % }">
    <!--

https://templatemo.com/tm-559-zay-shop

-->
</head>

<body>
    <!-- Start Top Nav -->
    <nav class="navbar navbar-expand-lg bg-dark navbar-light d-none d-lg-block"
    id="templatemo_nav_top">
        <div class="container text-light">
            <div class="w-100 d-flex justify-content-between">

```

```

        <div>
            <i class="fa fa-envelope mx-2"></i>
            <a class="navbar-sm-brand text-light text-decoration-none"
href="mailto:info@company.com">info@company.com</a>
            <i class="fa fa-phone mx-2"></i>
            <a class="navbar-sm-brand text-light text-decoration-none" href="tel:010-020-
0340">010-020-0340</a>
        </div>
        <div>
            <a class="text-light" href="https://fb.com/templatemo" target="_blank"
rel="sponsored"><i class="fab fa-facebook-f fa-sm fa-fw me-2"></i></a>
            <a class="text-light" href="https://www.instagram.com/" target="_blank"><i
class="fab fa-instagram fa-sm fa-fw me-2"></i></a>
            <a class="text-light" href="https://twitter.com/" target="_blank"><i class="fab fa-
twitter fa-sm fa-fw me-2"></i></a>
            <a class="text-light" href="https://www.linkedin.com/" target="_blank"><i class="fab
fa-linkedin fa-sm fa-fw"></i></a>
        </div>
    </div>
</div>
</nav>
<!-- Close Top Nav -->

```

```

<!-- Header -->
<nav class="navbar navbar-expand-lg navbar-light shadow">
    <div class="container d-flex justify-content-between align-items-center">
        

        <a class="navbar-brand text-success logo h1 align-self-center" href="index.html">

        </a>

        <button class="navbar-toggler border-0" type="button" data-bs-toggle="collapse" data-bs-
target="#templatemo_main_nav" aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>

        <div class="align-self-center collapse navbar-collapse flex-fill d-lg-flex justify-content-lg-
between" id="templatemo_main_nav">
            <div class="flex-fill">
                <ul class="nav navbar-nav d-flex justify-content-between mx-lg-auto">
                    <li class="nav-item">
                        <a class="nav-link" href="/myauth/home">Home</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="#about">About</a>
                    </li>
                </ul>
            </div>
        </div>
    </div>
</nav>

```

```

        <li class="nav-item">
            <a href="http://127.0.0.1:8000/myauth/productallview/">Shop</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="#contact">Contact</a>
        </li>
        <!--<li class="nav-item">
            <a class="nav-link" href="/myauth/logout/">Logout</a>
        </li>-->

    <!-- <li class="nav-item">
        <a class="nav-link" href="contact.html">Welcome { { user.username } } </a>

    </li>

    <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            Profile
        </a>
        <div class="dropdown-menu" aria-labelledby="navbarDropdown">
            <a class="dropdown-item" href="">Signup</a>
            <a class="dropdown-item" href=""></a>
            <a class="dropdown-item" href="/myauth/logout/">Logout</a>
        </div>
    </li>-->

    <!--<li class="nav-item dropdown">

        <a class="nav-link dropdown-toggle" href="#" id="userDropdown"
role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            Welcome { { user.username } }
        </a>
        <div class="dropdown-menu" aria-labelledby="userDropdown">
            <a class="dropdown-item" href="#">Profile</a>
            <a class="dropdown-item" href="/myauth/logout/">Logout</a>
        </div>

    </li>-->

</div>
<div class="navbar align-self-center d-flex">
    <div class="d-lg-none flex-sm-fill mt-3 mb-4 col-7 col-sm-auto pr-3">
        <div class="input-group">
            <input type="text" class="form-control" id="inputMobileSearch"
placeholder="Search ...">
            <div class="input-group-text">

```

```

        <i class="fa fa-fw fa-search"></i>
      </div>
    </div>
  </div>
  <div>
    <a class="nav-icon d-none d-lg-inline" href="/myauth/search/" >
      <i class="fa fa-fw fa-search text-dark mr-2"></i>
    </a>
    <a
      class="nav-icon          position-relative          text-decoration-none"
      href="/myauth/view_cart/">
      <i class="fa fa-fw fa-cart-arrow-down text-dark mr-1"></i>
      <span class="position-absolute top-0 left-100 translate-middle badge rounded-pill
      bg-light text-dark">
        {{ cart_items_count }}
      </span>
    </a>
  <!-- <li class="nav-item dropdown">

    <a class="nav-link dropdown-toggle" href="#" id="userDropdown" role="button"
    data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      {{ user.username }}
    </a>
    <div class="dropdown-menu" aria-labelledby="userDropdown">
      <a class="dropdown-item" href="#">Profile</a>
      <a class="dropdown-item" href="/myauth/logout/">Logout</a>
    </div>-->

  </li>
  <!--<a class="nav-icon position-relative text-decoration-none" href="#">
    <i class="fa fa-fw fa-user text-dark mr-3"></i>
    <span class="position-absolute top-0 left-100 translate-middle badge rounded-pill
    bg-light text-dark">+99</span>
  </a-->
</div>
</div>

</div>
</nav>
<!-- Close Header -->
<!--demo header-->
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Website Header</title>
  <style>
    /* Header styles */
    header {
      background-color: #2c621d;

```

```
        color: #fff;
        padding: 10px 0;
    }

    .container {
        max-width: 1200px;
        margin: 0 auto;
        display: flex;
        justify-content: space-between;
        align-items: center;
    }

    /* Navigation styles */
    nav ul {
        list-style: none;
        margin: 0;
        padding: 0;
        display: flex;
    }

    nav ul li {
        margin-right: 20px;
    }

    nav ul li:last-child {
        margin-right: 0;
    }

    nav a {
        text-decoration: none;
        color: #fff;
        font-weight: bold;
    }

    /* Dropdown styles */
    .dropdown {
        position: relative;
    }

    .dropdown-content {
        display: none;
        position: absolute;
        background-color: #333;
        min-width: 120px;
        z-index: 1;
    }

    .dropdown:hover .dropdown-content {
        display: block;
    }
```



```
.dropdown-content a {
  padding: 10px;
  text-align: center;
  color: #fff;
  text-decoration: none;
  display: block;
}

.dropdown-content a:hover {
  background-color: #555;
}

.user {
  display: flex;
  align-items: center;
  margin-left: auto; /* Align to the right */
}

.user img {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  margin-right: 10px; /* Add some space between image and text */
}

header {
  background-color: #2c621d;
  color: #fff;
  padding: 10px 0;
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.back {
  padding: 10px;
  display: flex; /* Use flex to align items horizontally */
}

.back ul {
  list-style: none;
  margin: 0;
  padding: 0;
  display: flex;
}
```

```
.back ul li {
    margin-right: 20px;
}

.back ul li:last-child {
    margin-right: 0;
}

.back a {
    text-decoration: none;
    color: #fff;
    font-weight: bold;
}

.user.dropdown {
    display: flex;
    align-items: center;
    margin-left: auto;
}

.user.dropdown a {
    display: flex;
    align-items: center;
    color: #fff;
    text-decoration: none;
}

.user.dropdown img {
    width: 40px;
    height: 40px;
    border-radius: 50%;
    margin-right: 10px;
}

.dropdown-content {
    display: none;
    position: absolute;
    background-color: #343a40;
    min-width: 120px;
    z-index: 1;
}

.user.dropdown:hover .dropdown-content {
    display: block;
}

.dropdown-content a {
    padding: 10px;
    text-align: center;
    color: #fff;
}
```

```

        text-decoration: none;
        display: block;
    }

    .dropdown-content a:hover {
        background-color: #555;
    }
</style>
</head>

<body>
    <header>
        <div class="container">
            <div class="back">
                <ul>
                    <li><a href="http://127.0.0.1:8000/myauth/productallview/">All Collections
</a></li>
                    <li class="dropdown">
                        <a href="#">Categories</a>
                        <div class="dropdown-content">
                            <!-- Categories loop -->
                            {% for category in categories %}
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/{{ category.id
}}">{{ category.category_name }}</a>
                                {% endfor %}
                            </div>
                        </li>
                        <li class="dropdown">

                            <a href="#">Sub-categories</a>
                            <div class="dropdown-content">
                                <!-- Subcategories loop -->
                                {% for subcategory in subcategories %}
                                    <a href="http://127.0.0.1:8000/myauth/productssubcategorylist/{{
subcategory.id }}">{{ subcategory.sub_category_name }}</a>
                                    {% endfor %}
                                </div>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/4">Men</a>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/5">Women</a>
                            </li>
                            <li>
                                <a href="http://127.0.0.1:8000/myauth/productscategorylist/6">Kids</a>
                            </li>
                            <li>

```

```

        <a href="#">Latest Collections</a>
    </li>

</ul>
</div>
<div class="user dropdown">
    <a href="#">
        { % if user.profile.profile_picture % }
        
        { % else % }
        
        { % endif % }
        Hello, {{ user.first_name }}
    </a>
    <div class="dropdown-content" style="margin-top: 215px;">
        <a href="/myauth/updateprofile/">Profile</a>
        <a href="/myauth/view_cart/">View Cart</a>
        <a href="/myauth/view_wishlist/">View Wishlist</a>
        <a href="/myauth/logout/">Logout</a>
    </div>
</div>
</div>
</div>
</header>
<!-- Your page content goes here -->
</body>

</html>

<!--demoheader close-->

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My Cart</title>
    <style>
        .body_design {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 20px;
            background-color: #f4f4f4;
            color: #333;
            display: flex;
            flex-direction: column;
            align-items: center;
            width: 100%;
        }
    </style>

```

```
h1 {
  text-align: center;
}

table {
  width: 80%;
  border-collapse: collapse;
  margin-top: 20px;
}

th, td {
  border: 1px solid #ddd;
  padding: 10px;
  text-align: left;
}

th {
  background-color: #f2f2f2;
}

img {
  width: 100px;
  height: 100px;
  object-fit: cover;
  border: 1px solid #ddd;
  border-radius: 4px;
  OBJECT-FIT: contain;
}

.quantity {
  width: 50px;
  text-align: center;
}

.increment-btn,
.decrement-btn,
.remove-btn,
.checkout-btn,
.back-btn {
  background-color: #4caf50;
  color: #fff;
  border: none;
  padding: 5px 10px;
  cursor: pointer;
  display: inline-block;
  margin-right: 10px;
}
```

```

.increment-btn:hover,
.decrement-btn:hover,
.remove-btn:hover,
.checkout-btn:hover,
.back-btn:hover {
    background-color: #45a049;
}

#total-price {
    font-size: 24px;
    font-weight: bold;
    margin-top: 20px;
    align-self: flex-end;
}

.buttons-container {
    display: flex;
    margin-top: 20px;
}
</style>
</head>
<body>
<div class="body_design">
<h1> Cart Items</h1>

<table>
<thead>
<tr>
<th>Product Name</th>
<th>Product Image</th>
<th>Quantity</th>
<th>Price</th>

<th>Action</th>
</tr>
</thead>
<tbody>

{ % for item in cart_items % }
<tr>
<td>{{ item.product.product_name }}</td>
<td>
<a href="{ % url 'prodetailview' item.product.id % }">

</a>
</td>
<td>
<input type="number" class="quantity" value="{ { item.quantity } }" min="1">

```

```

<input type="hidden" class="stock" value="{{ item.product.stock }}">

<input type="hidden" class="id_no" value="{{ item.id_number_info }}">

<input type="hidden" class="cart_id" value="{{ item.cart_id }}">

<input type="hidden" class="item_id" value="{{ item.id }}">

<button class="decrement-btn">-</button>
<button class="increment-btn">+</button>
</td>

<td class="item-price">
  <input type="hidden" class="update_price" value="{{ item.product.current_price
}}">
  ₹ <span data-price="{{ item.product.current_price }}">{{ item.product.current_price
}}</span></td>
<td>
  <!-- <a href="{{ url 'prodetailview' item.product.id }}" class="view-details">View
Details</a>-->

  <form class="delete-form" method="POST" action="{{ url 'delete_cartitem' item.id
% }}">
    {{ csrf_token }}
    <button type="button" class="remove-btn" style="width: 100px; height:
40px;">Remove</button><br><br>
  </form>

  <form class="delete-form" method="POST" action="{{ url 'prodetailview'
item.product.id % }}">
    {{ csrf_token }}
    <button type="button" class="remove-btn" style="width: 100px; height: 40px;">View
Details</button>
  </form>

</td>
</tr>
{{ % endfor % }}
</tbody>
</table>
<p id="total-price">Total Price: ₹<span id="total-price-value"></span></p>

<div class="buttons-container">
  <a href="/myauth/home/">

```

```

    <button class="">Back to Shopping</button>
</a>

<!-- Add a margin to create space -->
<span style="margin-right: 10px;"></span>
<button class="checkout-btn">Checkout</button>

<!-- <form action="https://www.example.com/payment/success/" method="POST">
<script
  src="https://checkout.razorpay.com/v1/checkout.js"
  data-key="rzp_test_UWsnOOpJ8I5HSD"
  data-amount="{ { total_amount_in_paise } }"
  data-currency="INR"
  data-order_id="{ { order.id } }"
  data-buttontext="Pay with Razorpay"
  data-name="Acme Corp"
  data-description="A Wild Sheep Chase is the third novel by Japanese author Haruki
Murakami"
  data-image="https://example.com/your_logo.jpg"
  data-prefill.name="Pranav"
  data-prefill.email="pranav@gmail.com"
  data-theme.color="#F37254"
></script>
  <input type="hidden" custom="Hidden Element" name="hidden"/>
</form> </div> -->
<script>
  document.addEventListener('DOMContentLoaded', function() {
    const quantityInputs = document.querySelectorAll('.quantity');
    const decrementBtns = document.querySelectorAll('.decrement-btn');
    const incrementBtns = document.querySelectorAll('.increment-btn');
    const removeBtns = document.querySelectorAll('.remove-btn');
    const itemPrices = document.querySelectorAll('.item-price span');
    const totalPriceElem = document.getElementById('total-price-value');

    const checkoutBtn = document.querySelector('.checkout-btn');

    // checkoutBtn.addEventListener('click', function(event) {
    //   event.preventDefault();
    //   // Get the cart_id value
    //   const cartId = document.querySelector('.cart_id').value;
    //   window.location.href = `/checkout/${cartId}`;

    // });
    checkoutBtn.addEventListener('click', function(event) {
      event.preventDefault();

      const items = [];

```



```
const rows = document.querySelectorAll('tbody tr');

rows.forEach(function(row) {
  const itemId = parseInt(row.querySelector('.item_id').value);
  const quantity = parseInt(row.querySelector('.quantity').value);
  const price = parseFloat(row.querySelector('.update_price').value);

  items.push({
    "item_id": itemId,
    "quantity": quantity,
    "price": price
  });
});

// You can now use the 'items' array to send JSON data to your backend.
console.log(items);

window.location.href =
`/myauth/checkout/?items=${encodeURIComponent(JSON.stringify(items))}`;
});

function updateItemPrice(item, quantity) {
  const price = parseFloat(item.querySelector('.item-price span').getAttribute('data-price'));
  const itemPriceElem = item.querySelector('.item-price span');
  const totalPrice = price * quantity;
  itemPriceElem.textContent = totalPrice.toFixed(2);

  const hiddenInputp = item.querySelector('.update_price');
  hiddenInputp.value = totalPrice.toFixed(2);
}

function updateTotalPrice() {
  let totalPrice = 0;
  itemPrices.forEach(function(price) {
    totalPrice += parseFloat(price.textContent);
  });
  totalPriceElem.textContent = totalPrice.toFixed(2);
}

quantityInputs.forEach(function(quantityInput, index) {
  const decrementBtn = decrementBtns[index];
  const incrementBtn = incrementBtns[index];
  const item = quantityInput.closest('tr');

  const stockDataInput = item.querySelector('.stock');

  incrementBtn.addEventListener('click', function() {
    let quantity = parseInt(quantityInput.value);

    let stockdata = parseInt(stockDataInput.value);
```

```
        if(quantity >= stockdata) {
            alert("Only "+stockdata+" stocks available")
        }
        else if(quantity < 1) {
            alert("Invalid quantity")
        }
        else {
            quantity++;
            quantityInput.value = quantity;
            updateItemPrice(item, quantity);
            updateTotalPrice();
        }
    });

    decrementBtn.addEventListener('click', function() {
        let quantity = parseInt(quantityInput.value);
        if (quantity > 1) {
            quantity--;
            quantityInput.value = quantity;
            updateItemPrice(item, quantity);
            updateTotalPrice();
        }
    });

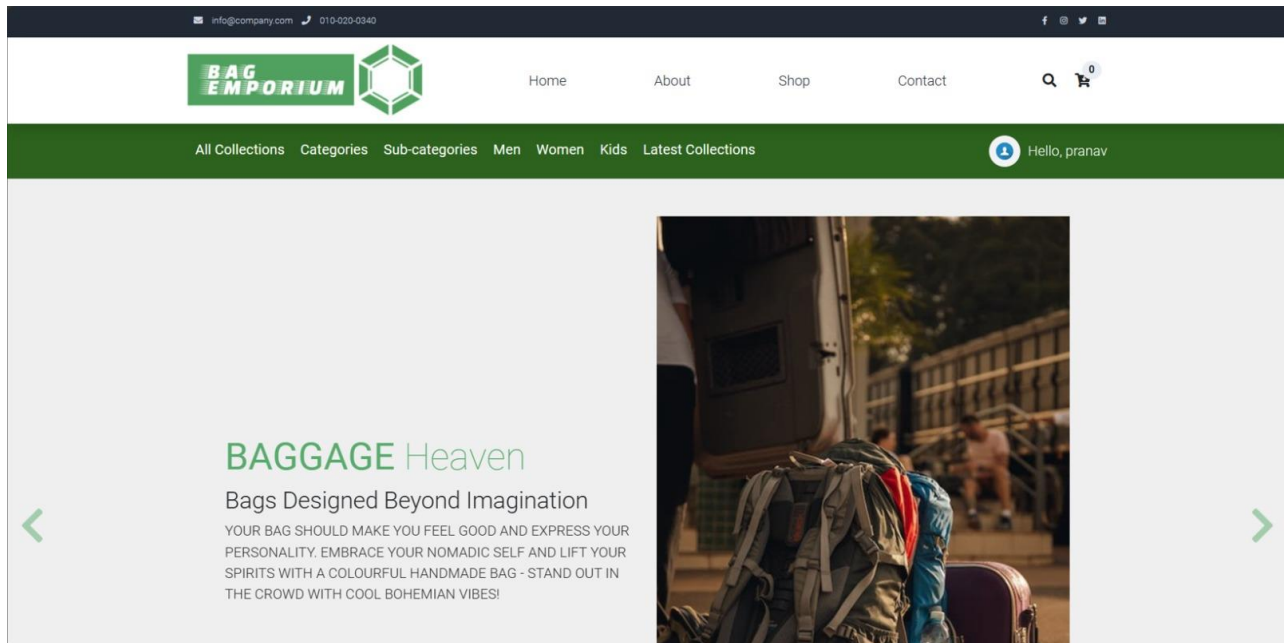
    quantityInput.addEventListener('input', function() {
        let quantity = parseInt(quantityInput.value);
        updateItemPrice(item, quantity);
        updateTotalPrice();
    });
});

removeBtns.forEach(function(removeBtn) {
    removeBtn.addEventListener('click', function(event) {
        event.preventDefault();
        const deleteForm = removeBtn.closest('.delete-form');
        deleteForm.submit();
    });
});

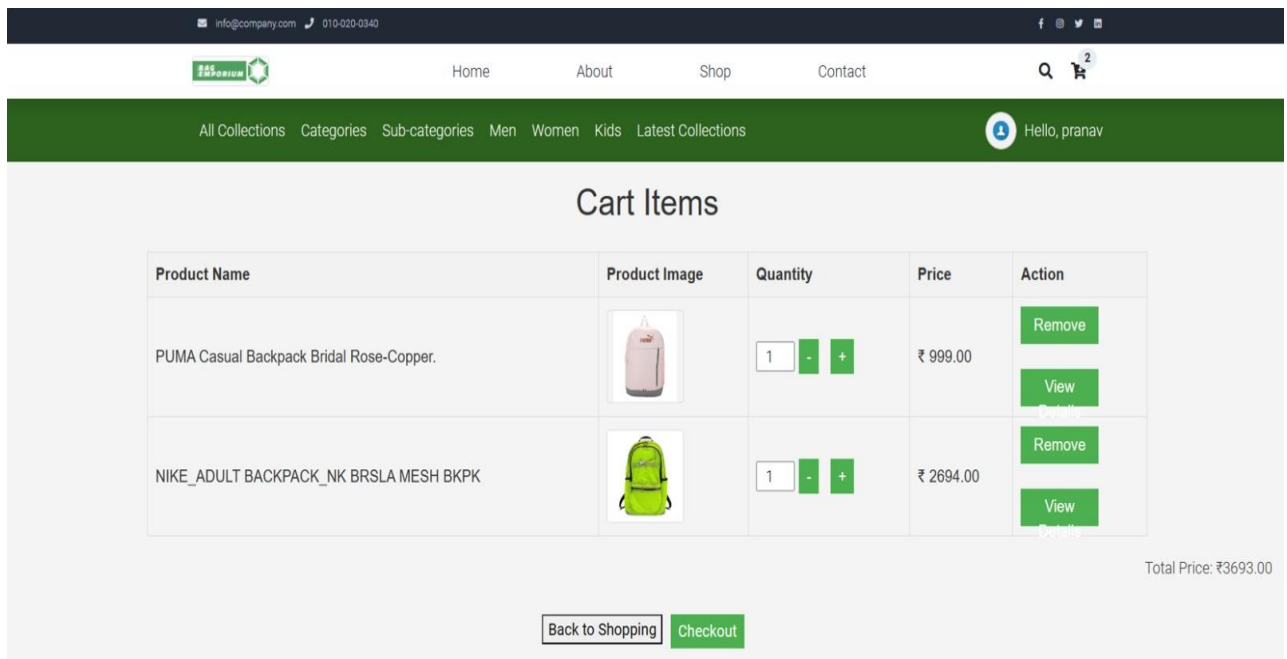
    updateTotalPrice();
});
</script>
</div>
</body>
</html>
```

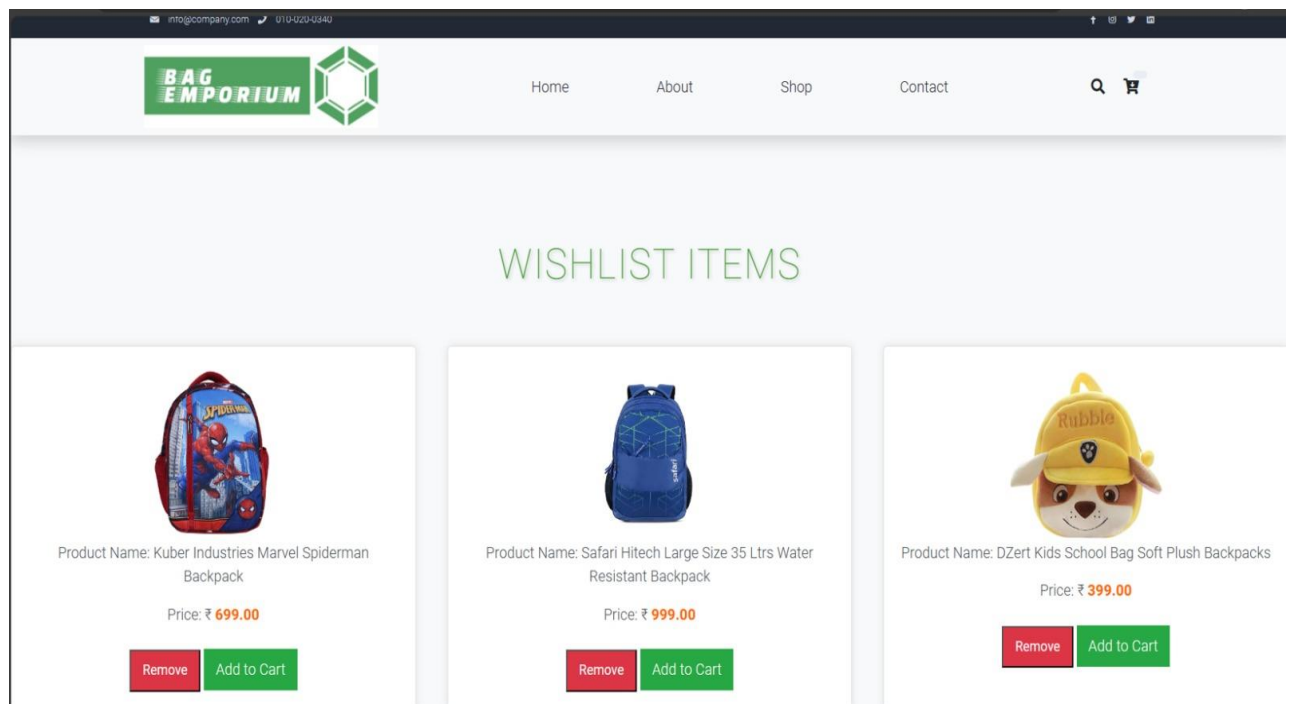
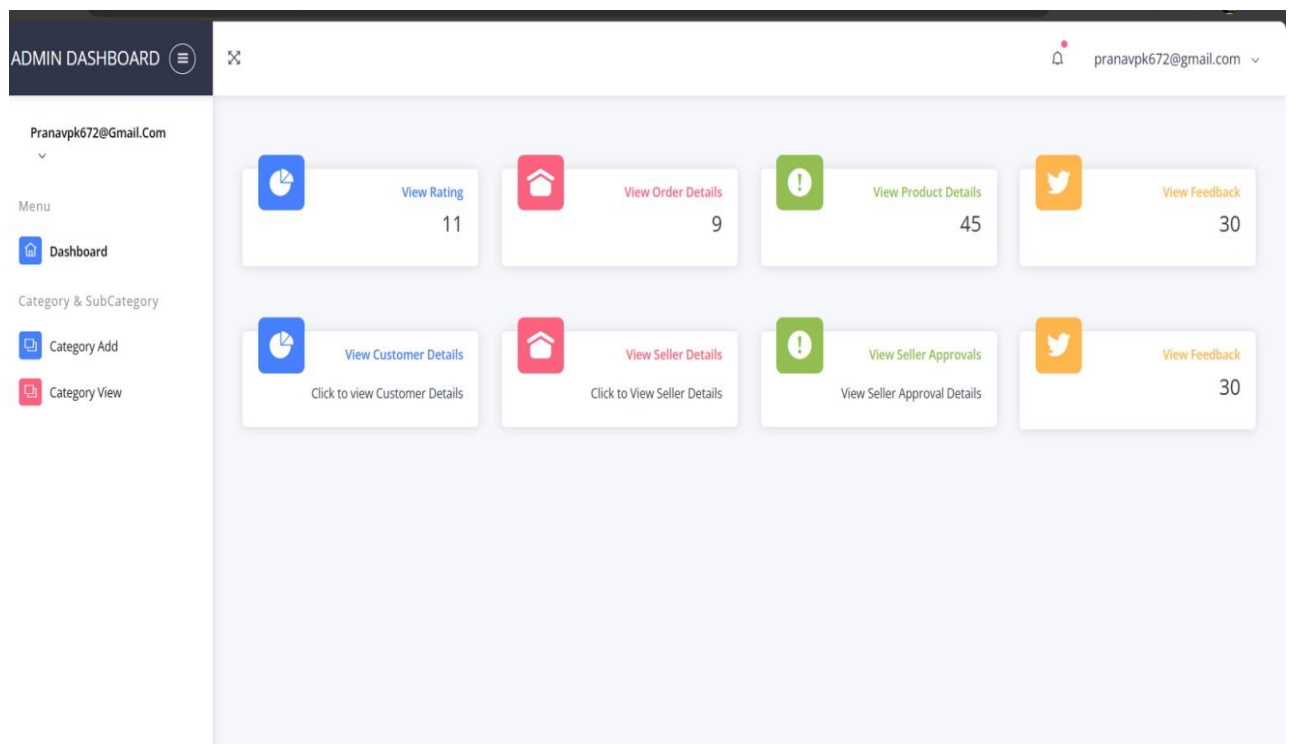
9.2 SCREEN SHOTS

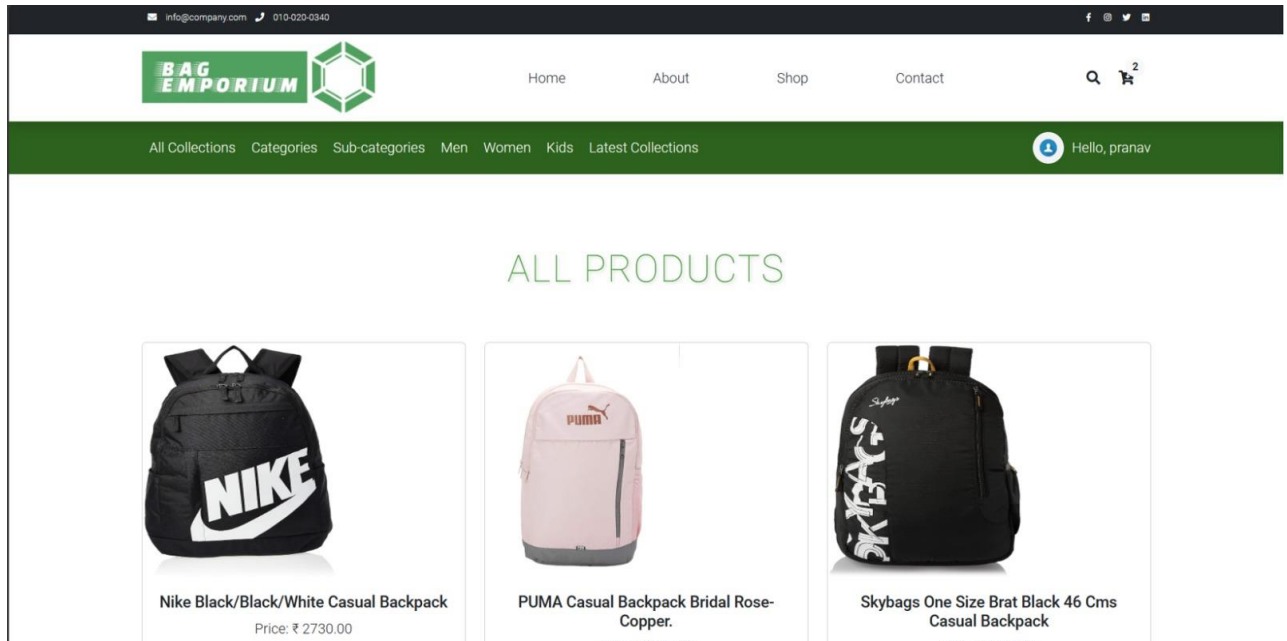
Customer: Home Page



Customer : Add to Cart



Customer : Add to Wishlist**Admin Dashboard: Home page**

Customer: Product View Page**Payment:**