Project Report On

ShopInn (Shopping Site)

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Advanced Computing

from

C-DAC ACTS (Pune),

Guided by

Mr. Vinu Josy,

Presented By :

Priyanshu Shukla - 240340120141

Rahul Mishra- 240340120142

Sagar Negi – 240340120161

Varun Khadse - 240340120220

Centre of Development of Advanced Computing (C-DAC), Pune,Maharashtra.

CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Priyanshu Shukla - 240340120141

Rahul Mishra- 240340120142

Sagar Negi – 240340120161

Varun Khadse - 240340120220

have successfully completed their **project** titled

“ShopInn (Shopping Site)”

Under the **Guidance** of **Mr. Vinu Josy**

Project Guide **HOD ACTS**

ACKNOWLEDGEMENT

This project **“ShopInn (Shopping Site)”** was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC ACTS).

We all are very glad to mention the name of **Mr. Vinu Josy** for his valuable guidance to work on this project. His guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

Our most heartfelt thank goes to Ms **Namrata mam** (Course Coordinator, PG DAC) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in C-DAC ACTS,

Pune.

Priyanshu Shukla - 240340120141

Rahul Mishra- 240340120142

Sagar Negi – 240340120161

Varun Khadse - 240340120220

TABLE OF CONTENTS

Here’s a structured list based on the provided Table of Contents:

सी डैक CDAC ॲक्टस acts Authorized Training Centre

Project Synopsis: ShopInn

Project Definition :

The ShopInn is an e-commerce application that is a comprehensive solution designed to enhance the online shopping experience for end-users while providing robust management tools for sellers. This application is being developed using cutting-edge technologies including React for the front-end and Spring Boot for the back-end. The primary aim of the project is to offer a seamless, user-friendly platform where customers can easily browse, search for, and purchase products, while sellers can manage their product listings and monitor their sales. Additionally, the application includes a powerful administrative interface for analyzing sales data, overseeing platform operations, and ensuring compliance with business requirements. The system integrates with external payment gateways such as Razorpay, ensuring secure and efficient transaction processing.

Core Modules/Functionalities :

The E-Commerce Application is comprised of several core modules that collectively provide a comprehensive online shopping experience. Each module is designed to meet specific business requirements while ensuring ease of use and reliability.

1. User Registration and Authentication: This module ensures that users, including customers and sellers, can register on the platform securely. The authentication process involves secure login mechanisms, password encryption, and the implementation of user roles to manage access rights. Users can log in using their credentials, which are securely stored in the database.

2. Product Browsing and Searching: This module enables users to explore a wide range of products available on the platform. Users can browse products by categories, utilize search functionality to find specific items, and access detailed product descriptions. The user interface is designed to be responsive and intuitive, ensuring that users have a smooth browsing experience on any device.

3. Shopping Cart Management: This module allows users to add products to a virtual shopping cart. Users can view the contents of their cart, modify quantities, or remove items as needed. The cart is persistently managed throughout the session, ensuring that users do not lose their selections during navigation.

4. Order Placement and Checkout: The checkout module provides a streamlined process for finalizing purchases. Users can proceed to checkout, where they will provide shipping details and choose a payment method. The system integrates with external payment gateways like Razorpay to handle transactions securely. Upon successful order placement, users receive a confirmation email with order details and tracking information.

5. Order History and Tracking: This feature allows users to access their order history and track the status of their current orders. The system maintains detailed records of all transactions, enabling users to review past purchases and check delivery statuses at any time.

6. Seller Panel: The seller panel is a dedicated interface for sellers to manage their product listings. Sellers can add new products, update existing listings, and categorize their offerings for better organization. The panel also provides tools for tracking sales performance and receiving customer feedback.

7. Admin Panel: The admin panel is designed for platform administrators to oversee operations. Administrators have access to comprehensive analytics tools that allow them to monitor sales trends, analyze user behavior, and manage accounts. The admin panel also includes features for reviewing customer feedback, managing seller accounts, and ensuring compliance with regulations.

Team Composition

The development of this e-commerce application is being undertaken by a team of four skilled professionals. The team members bring diverse expertise in front-end development, back-end development, database management, and project coordination. Each member is responsible for specific aspects of the project, ensuring that all components are developed with high quality and efficiency.

Team Members:

- **Priyanshu Shukla - 240340120141**
- **Rahul Mishra- 240340120142**
- **Sagar Negi – 240340120161**
- **Varun Khadse - 240340120220**

Technology Stack

The e-commerce application is built using a modern technology stack that ensures scalability, security, and maintainability. The primary technologies and tools used in the development of this application include:

- Front-End: React is utilized for building the user interface, providing a dynamic and responsive experience for users. React's component-based architecture allows for reusable UI elements and a consistent look and feel across the platform.

- Back-End: Spring Boot is employed for the back-end development, offering a robust framework for building RESTful APIs and handling business logic. Spring Boot's ability to create stand-alone, production-grade applications with minimal configuration makes it ideal for this project. -

Database: MySQL is used for data storage, ensuring reliable and efficient management of user information, product details, and transaction records. The database is designed to handle a large volume of data while maintaining high performance.

- External Services: Razorpay is integrated for payment processing, providing secure and seamless transactions. The system may also integrate with other third-party services for shipping and customer reviews, depending on future requirements.

- Communication: RESTful APIs facilitate communication between the front-end and backend, ensuring smooth data flow and interaction across the system.

- Security: Security is implemented through Spring Security , and authentication is done through JWT authentication. Role based authentication is implemented so that sensitive data is kept secured and user can only access the relevant data for them.

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to detail the requirements for an e-commerce application designed with React for the front-end and Spring Boot for the back-end. This document aims to provide a clear understanding of the application's functionalities, architecture, and constraints, ensuring that all stakeholders have a shared vision of the project.

Objectives -

Clear Definition of Requirements: This document aims to precisely define the functional and non-functional requirements of the e-commerce application. By doing so, it ensures that all parties involved including developers, designers, testers, and stakeholders have a clear understanding of what the application should accomplish and how it should operate.

System Architecture Overview: The SRS will outline the system architecture, including the interaction between the front-end and back-end components, as well as integration points with external services such as payment gateways and shipping providers..

Scope of Work: By detailing the scope of the application, the SRS helps manage expectations and ensures that the project stays focused on its primary goals. It describes the features and functionalities that will be included in the system, as well as those that will be excluded.

Design Constraints and Assumptions: The document will identify design constraints, such as compatibility with specific web browsers, adherence to security standards, and performance requirements. It also outlines assumptions and dependencies, such as reliance on third-party services and internet connectivity.

Stakeholder Communication: The SRS serves as a communication tool among all stakeholders, including project managers, developers, end-users, sellers, and administrators. By providing a detailed and clear set of requirements, the document helps ensure that everyone involved has a shared understanding of the project's objectives, deliverables, and limitations.

Basis for Testing and Validation: The requirements outlined in this SRS will form the basis for testing and validation of the application. Detailed functional and non-functional requirements provide a foundation for creating test cases and ensuring that the application meets its specified goals and quality standards.

1.2 Scope

The e-commerce application will enable users to:

- Browse and search for products.
- Add products to a shopping cart.
- Place orders and proceed through checkout.
- View order history and track orders. Sellers will have capabilities to:
- Add, edit, and delete products.
- Monitor their sales performance. Administrators will be able to:
- Manage user accounts, including sellers.
- Analyze sales data and generate reports. The system will integrate with external payment gateways and potentially third-party services for shipping and reviews.

1.3 Definitions, Acronyms, and Abbreviations

React: A JavaScript library for building user interfaces, particularly single-page applications.

Spring Boot: A Java-based framework that simplifies the development of production-ready, stand-alone Spring-based applications.

REST: Representational State Transfer, an architectural style for designing networked applications using HTTP methods.

API: Application Programming Interface, a set of protocols and tools for building software applications.

1.4 References

React Documentation: https://reactjs.org/docs/getting-started.html

Spring Boot Documentation: https://spring.io/projects/spring-boot

1.5 Overview

This document provides a structured description of the e-commerce application, covering system functionality, architecture, and design requirements. It serves as a foundational guide for developers, testers, and stakeholders to ensure that the application meets its objectives.

---

2. Software Requirement and Specification

2.1 Overview

The e-commerce application is envisioned as a standalone web-based system. It will interact with various external systems such as payment gateways (e.g., Razorpay), shipping services, and possibly customer review platforms. The front-end, developed in React, provides a dynamic and responsive user interface. The back-end, built with Spring Boot, handles business logic, data management, and integration with external services.

2.2 Product Functions

The application will include the following core functions:

- User Registration and Authentication: Allow users to create accounts, log in, and manage their profiles securely.

- Product Browsing and Searching: Enable users to browse product categories and perform keyword searches.

- Shopping Cart Management: Provide functionalities for users to add, view, edit, and remove items in their shopping cart.

- Order Placement and Checkout: Facilitate the checkout process, including shipping and payment information, and send order confirmation emails.

- Order History and Tracking: Allow users to view their past orders and track the status of current orders.

- Seller Panel: Equip sellers with tools to manage their products, categories, and view sales performance.

- Admin Panel: Offer administrative features for managing seller accounts, analyzing sales data, and overseeing user reviews.

2.3 User Classes and Characteristics

End Users: Customers who will interact with the application to browse, purchase, and manage their orders.

Seller Users: Vendors who will use the application to list and manage their products, track sales, and manage categories.

Admin Users: Administrators who will have access to manage user accounts, analyze sales trends, and oversee reviews.

2.4 Operating Environment

Front-End: The React application will be hosted on a web server and will be compatible with major browsers such as Chrome, Firefox, Safari, and Edge.

Back-End: The Spring Boot application will be deployed on an application server, managing business logic and data operations.

Database: MySQL will be used for data storage, including user information, product details, and order history.

Payment Gateway: Integration with services like Razorpay for processing payments.

2.5 Design and Implementation Constraints

Browser Compatibility: The application must be tested and optimized for all major web browsers.

RESTful API: The back-end must adhere to RESTful principles for communication with the front-end.

Security: Implementation of encryption, secure authentication, and role-based access control to protect user data and transactions.

2.6 Assumptions and Dependencies

Browser Access: Users will access the application through modern web browsers.

Internet Connectivity: Reliable internet access is required for the application's functionality.

Third-Party Services: Dependence on external services for payments, shipping, and possibly customer reviews, which must be reliable.

---

3. Tools and Technologies Used

1. **Spring Boot**: Utilized to develop the backend of the application, providing a robust framework for building Java-based web applications with ease.

2. **Spring Data JPA:** Implemented for data access, allowing seamless interaction with the MySQL database to store and retrieve sports data efficiently.

3. RESTful Web Services: In the context of an e-commerce web application like Book Charm, RESTful web services play a crucial role in facilitating communication. These services adhere to the principles of Representational State Transfer (REST), which emphasizes a stateless, standardized approach for building web services.

4. Node JS: Employed for web scraping, enabling the application to extract live scores andmatch details from various sports websites and APIs.

5. Express JS: Express.js is a web application framework for Node.js that simplifies the creation of robust, scalable APIs and web applications by providing a set of middleware and routing mechanisms. It streamlines the process of handling HTTP requests, making it efficient for building server-side components in a Node.js application.

6. Spring Web: Used for handling web requests and responses, managing controllers, and serving static resources to the frontend.

7. Aiven cloud database MySQL: Chosen as the relational database management system to store book data on cloud ,including user detail ,seller detail, book information and admin details.

8. JWT (JSON Web Tokens): Implemented for secure user authentication and authorization, ensuring that only authenticated users can access into account and buy books and seller can add books .

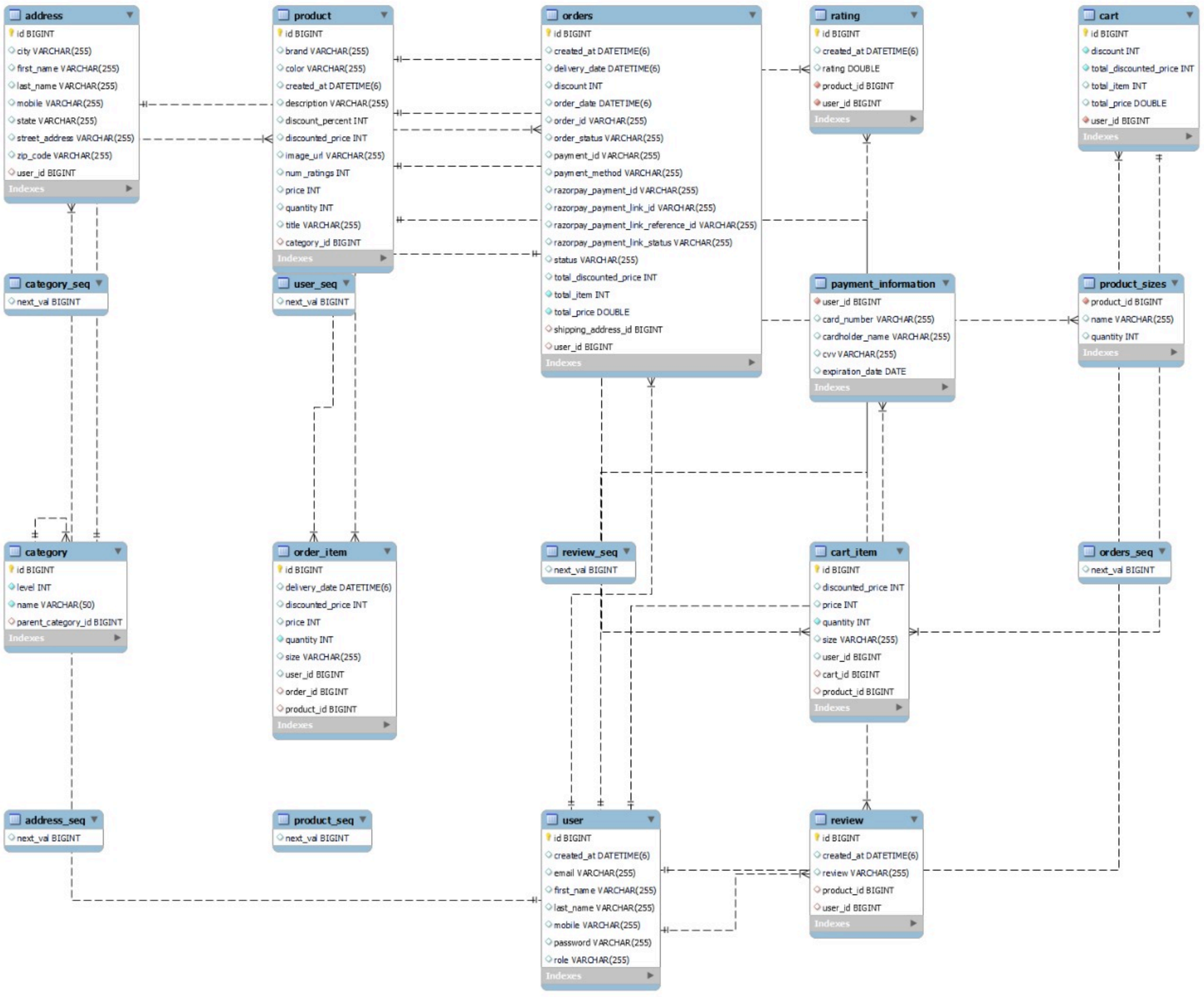
9. **Axios:** In the context of a web application like Book Charm, Axios is likely used as a client-side HTTP library. Axios simplifies the process of making asynchronous HTTP requests from the frontend to the backend. It is instrumental in fetching data from the server, handling API calls, and facilitating smooth communication between the frontend and backend components,and seamless user interactions in the e-commerce application.

10. **React:** Employed to build the frontend of the application, offering a component-based architecture for creating dynamic and interactive user interfaces.
11. **CSS:** Used for styling the frontend components with utility-first CSS classes,allowing for rapid prototyping and customization of the user interface.
12. **Material UI:** Leveraged to enhance the visual appeal and user experience of the application by incorporating pre-designed React components following Google's Material Design principles.

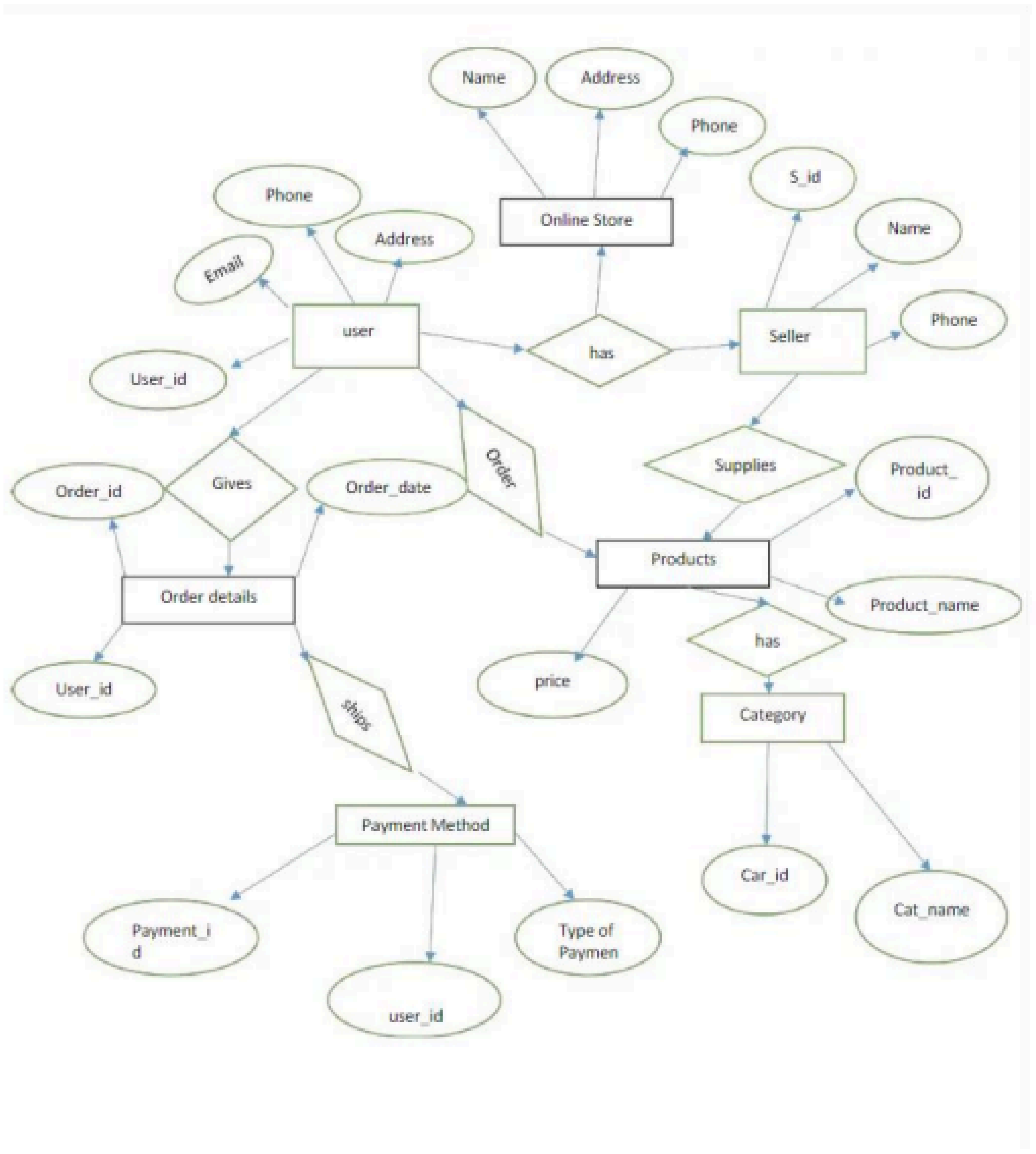
13. **Git:** Implemented as a version control system to track changes in the source code, enabling collaboration among developers, and facilitating code management and deployment workflows.

14. **Razor-pay Payment Integration:** In the Book Charm project, Razorpay is integrated as the payment gateway, enabling secure and streamlined online transactions. Razorpay provides a developer-friendly API, allowing seamless integration for processing payments, managing subscriptions, and ensuring a reliable end-to-end payment experience for users in the e-commerce web application.

4. Project Flow Diagram



5.Project E-R(Entity relationship) Diagram



6. Advantages

Scalability

- **Component-Based Architecture**: React's component-based structure allows developers to build modular, reusable components that can be easily updated or replaced. This modularity promotes scalability, as new features can be added or existing ones can be modified without affecting other parts of the application.

- **Microservices Capabilities**: Spring Boot supports the development of microservices, which are small, independently deployable services that work together. This architecture enables scaling of individual components or services as needed, improving the overall scalability of the application.

Responsive Design

- **Dynamic User Experience**: React ensures that the application provides a dynamic and interactive user experience. Its ability to update the user interface in real-time based on user interactions or data changes enhances usability and user satisfaction.

- **Cross-Device Compatibility**: The use of responsive design principles ensures that the application is accessible and fully functional across various devices, including desktops, tablets, and smartphones. This approach improves accessibility and user engagement.

Secure Transactions

- **Encryption**: The application employs encryption techniques to protect sensitive user data, including personal information and payment details. This ensures that data is secure both in transit (during transmission) and at rest (when stored).

- **Secure APIs**: RESTful APIs used for communication between the front-end and back-end are designed with security in mind. This includes implementing authentication and authorization mechanisms to prevent unauthorized access.

Admin and Seller Tools

- **Comprehensive Management Features**: The application includes robust tools for both administrators and sellers. For sellers, this means functionalities for managing products, categories, and sales performance. For administrators, it includes features for managing user accounts, analyzing sales data, and overseeing overall system operations.

- **Streamlined Operations**: These management tools help streamline day-to-day operations, improve efficiency, and provide valuable insights into sales and user behavior.

---

7. Screenshots

Home Page Overview: A screenshot of the home page showcasing the primary navigation, featured products, and promotional banners.

Product Listing Overview: A screenshot displaying the product listing page with product categories, filters, and search functionality.

Shopping Cart Overview: A screenshot of the shopping cart page showing added items, quantities, and the option to proceed to checkout.

Checkout Page Overview: A screenshot illustrating the checkout process, including shipping information, payment options, and order summary.

Admin Panel Overview: A screenshot of the admin panel interface, featuring tools for managing user accounts, viewing sales reports, and accessing system analytics.

[Note: Include actual screenshots from the application to provide a visual representation of these interfaces.]

8. FUTURE SCOPE

Mobile Application

- **Native or Hybrid Development**: Develop native mobile applications for iOS and Android platforms or hybrid applications using frameworks like React Native. This will expand the application's reach and accessibility, catering to users who prefer mobile devices for shopping.

Advanced Analytics

- **In-Depth Reporting**: Implement advanced analytics tools and features to provide deeper insights into sales trends, customer behavior, and product performance. This can include customizable reports, visual data dashboards, and predictive analytics.

AI Integration

- **Personalized Recommendations**: Integrate artificial intelligence algorithms to offer personalized product recommendations based on user behavior, preferences, and purchase history.

- **Customer Support Chatbots**: Implement AI-powered chatbots to provide real-time assistance to customers, answer common queries, and enhance the overall customer support experience.

Enhanced Security

- **Multi-Factor Authentication (MFA)**: Explore additional security measures such as multi-factor authentication to provide an extra layer of protection for user accounts.

- **Advanced Encryption**: Investigate advanced encryption techniques to further safeguard sensitive data and ensure compliance with evolving security standards.

9. Conclusion

The e-commerce application detailed in this SRS is designed to offer a comprehensive and efficient online shopping experience. By leveraging React for a dynamic, responsive front-end and Spring Boot for a robust, scalable back-end, the application aims to meet the needs of users, sellers, and administrators. The combination of these technologies ensures a secure, high-performance platform that is capable of handling a growing user base and evolving business requirements. The system's design and technology stack are carefully chosen to provide a reliable, user-friendly experience while supporting future enhancements and scalability.

10. References

1. **https://spring.io/projects/spring-boot**
2. **https://spring.io/projects/spring-data-jpa**
3. **https://restfulapi.net/**
4. **https://www.mysql.com/**
5. **https://spring.io/projects/spring-web**
6. **https://reactjs.org/**
7. **https://nodejs.org/**