INTRODUCTION

Chapter 1

INTRODUCTION

1.1 Overview

An **online book store** is a virtual store on the Internet where customers can browse the catalog and select books of interest. At checkout time, the items in the e-library will be presented as an order. At that time, more information will be needed to complete the request. Usually, the customer will be asked to fill online form. An e- mail notification is sent to the customer as soon as the order is placed.

This project intends different types of forms with many types of books like story, drama, romance, history, adventures, etc. it can manage studying of books online, customers can choose many types of books categories, etc. Here, the user may select desired book and view its price. The user may even search for specific books on the website. Once the user selects a book, he then has to fill in a form and the book is provided for the user.

1.2 Objectives and Goals

The goal of this project is to create a website that serves as a central book store. This website was built with php on the front end and SQL on the back end. Various book-related details are stored in the SQL database. A user visiting the website will find a wide selection of books organized by category. The user can choose a book and see its price. The user can even utilize the website to look for certain books. After the user chooses a book, he/she must fill out a form before the book is booked for the user.

Customers may shop for books online using a web browser thanks to the Online Book Store Project. A customer can create an account, log in, sort books by category, add books to a shopping basket, and pay their bill using their credit card information. When compared to a regular user, the Administrator will have more options. He can edit the author, publisher, book categories, book details, and member information, as well as confirm an order.

1.3 Existing System

In the existing system buying and selling a product is done manually. Price of the product is fixed by the seller. All the details of the product to be sold or purchased are maintained manually. Sellers or buyers not able to get the complete information about the product

1.4 Problem Statement

The problem statement phase examines the current challenges and opportunities in developing an online bookstore platform using PHP and MySQL. This section provides a detailed exploration of user requirements, market needs, and technical considerations.

User Requirements

- **User Accessibility**: Users expect a seamless browsing experience across different devices (desktops, tablets, mobile phones). Responsive design and intuitive navigation are critical to ensure accessibility.
- **Secure Transactions**: Users demand secure payment gateways and data protection measures to safeguard their personal and financial information.
- **Comprehensive Book Catalog**: A diverse and extensive collection of books categorized by genre, author, and popularity is essential to cater to varying user preferences.
- **Efficient Search and Filtering**: Advanced search functionalities and filters (e.g., by price range, publication date) enhance user experience by facilitating quick and precise book discovery.
- **Personalization**: Users appreciate personalized recommendations based on their browsing history, purchase behavior, and preferences.

Market Needs

- **Competitive Landscape**: Competing with established online retailers like Amazon requires unique selling propositions (USPs) such as specialized book categories, exclusive editions, or personalized customer service.
- **Customer Trust**: Building trust through transparent pricing, reliable customer support, and secure transactions is crucial for customer retention and acquisition.
- **Scalability**: The platform should accommodate future growth in terms of user traffic, inventory expansion, and technological advancements without compromising performance.

Technical Considerations

- **Technology Stack**: Choosing PHP for backend development and MySQL for database management offers flexibility, scalability, and compatibility with widely available hosting services.
- **MVC Architecture**: Implementing the Model-View-Controller (MVC) architectural pattern ensures separation of concerns, simplifying code maintenance and scalability.
- **Security Measures**: Incorporating SSL/TLS encryption, hashed passwords, and secure coding practices mitigates risks associated with data breaches, SQL injection attacks, and unauthorized access.
- **Performance Optimization**: Techniques such as caching mechanisms, database indexing, and content delivery networks (CDNs) optimize page loading speed and overall system performance.

Advantages

- **Wide Compatibility**: PHP and MySQL are widely supported across hosting providers, ensuring compatibility and ease of deployment.

- **Community Support**: Both PHP and MySQL have large developer communities, providing access to resources, tutorials, and plugins/extensions for extended functionality.
- **Scalability**: MVC architecture facilitates scalability by allowing independent development of components (models, views, controllers) and easy integration of new features.
- **Cost-Effectiveness**: Open-source nature of PHP and MySQL reduces licensing costs, making them suitable for startups and small businesses.

Disadvantages

- **Security Vulnerabilities**: PHP is susceptible to security vulnerabilities if not managed properly, requiring continuous updates and monitoring.
- **Performance**: PHP's interpreted nature can lead to slower execution compared to compiled languages, necessitating performance optimization techniques.
- **Learning Curve**: MVC architecture may have a steeper learning curve for novice developers unfamiliar with the separation of concerns principle.
- **Database Scaling**: MySQL's scalability may require advanced configuration and partitioning strategies to handle large datasets and high concurrency.

1.5 Scope

The main scope deliverables of the project would be to:

- Analyze and develop detailed specifications and requirements
- Prepare high-level and detailed system design specifications
- Prepare a test plan as well as test cases.
- Develop the system and write the code.
- Unit, integration, and system testing should all be performed.
- Demonstrate a bug-free application after making any necessary changes.

CHAPTER 2
LITERATURE SURVEY

Overview

- The literature survey reviews existing research and developments related to online bookstores, e-commerce platforms, and relevant web technologies. It identifies the strengths and weaknesses of current solutions, examines recent technological advancements, and highlights areas where this project can contribute.

Existing Online Bookstores

- **Amazon**: Amazon is the largest online bookstore, known for its vast inventory, user-friendly interface, and efficient logistics. However, it has faced criticism for its treatment of third-party sellers and lack of personalized user experience.
- **Barnes & Noble**: Barnes & Noble offers a more curated selection of books and an emphasis on community events. Its website is well-designed but lacks some advanced features like dynamic recommendations and user-generated content.
- **Smaller Bookstores**: Independent bookstores like Powell's Books and Thrift Books have niche markets and unique inventories but often struggle with limited resources and less robust technological infrastructure.

Technological Advancements

- **MVC Architecture**: The Model-View-Controller (MVC) pattern helps in organizing code and separating concerns, which makes the development process more manageable and scalable. Frameworks like Laravel and CodeIgniter provide robust features to facilitate MVC-based development.
- **Responsive Design**: With the increasing use of mobile devices, the websites are accessible and user-friendly across different screen sizes.

- **Security Measures**: Ensuring security in e-commerce platforms is crucial. Techniques such as SSL/TLS for secure communications, hashed passwords for user authentication, and secure coding practices are vital for protecting user data.

User Experience and Interface Design

- **Intuitive Navigation**: Studies show that users prefer websites that are easy to navigate. Effective use of menus, search bars, and categorization enhances the user experience. Amazon's recommendation system is a benchmark for personalized navigation.
- **Accessibility**: Ensuring that the website is accessible to users with disabilities is both a legal requirement and a best practice. Techniques include proper use of ARIA roles, keyboard navigation support, and text-to-speech compatibility.
- **Aesthetics and Usability**: The visual appeal of a website significantly impacts user engagement. Clean design, consistent color schemes, and appropriate typography contribute to a positive user experience.

E-commerce Features

- **Shopping Cart Systems**: Essential for any online store, shopping cart systems need to be robust and user-friendly. Features like cart persistence, easy updates, and clear checkout processes are critical.
- **Payment Gateways**: Integrating reliable and secure payment gateways is crucial. Options include PayPal, Stripe, and direct credit card processing. Security measures such as encryption and compliance with PCI DSS standards are mandatory.
- **Inventory Management**: Effective inventory management systems help in tracking stock levels, managing orders, and forecasting demand. Real-time updates and automated alerts are important features.
- **User Reviews and Ratings**: Allowing users to review and rate products enhances credibility and provides valuable feedback.

Challenges in Existing Solutions

- **Scalability Issues**: Many existing online bookstores face challenges in scaling their operations to handle increased traffic and larger inventories. Solutions often require significant investments in infrastructure and optimization.
- **Security Concerns**: E-commerce platforms are frequent targets for cyber-attacks. Ensuring data security and protecting against threats such as SQL injection, cross-site scripting (XSS), and data breaches is critical.
- **User Retention**: Keeping users engaged and encouraging repeat visits requires a combination of excellent user experience, personalized recommendations, and effective marketing strategies.

Recent Research and Findings

- **Machine Learning for Recommendations**: Recent research highlights the effectiveness of machine learning algorithms in providing personalized book recommendations. Techniques such as collaborative filtering and content-based filtering improve user satisfaction and sales.
- **Blockchain for Secure Transactions**: Blockchain technology offers promising solutions for secure and transparent transactions. Its application in e-commerce can enhance trust and reduce fraud.
- **AI Chatbots**: The use of AI-driven chatbots can improve customer service by providing instant responses to user queries, assisting in navigation, and offering personalized suggestions.

Gaps and Opportunities

- **Enhanced Personalization**: While major platforms like Amazon excel in personalized recommendations, smaller online bookstores often lack these capabilities. Implementing advanced recommendation systems can provide a competitive edge.

- **Improved User Engagement**: Features such as social sharing, virtual book clubs, and interactive author sessions can enhance user engagement and build a community around the bookstore.

Testing Methodologies

Unit Testing

- **Purpose**: To verify the correctness of individual units or modules of code.
- **Approach**: Utilizing PHP Unit or similar frameworks to automate tests that isolate and validate specific functionalities.
- **Examples**: Testing CRUD operations for user authentication, validation rules for input forms, and calculations for order totals.

Integration Testing

- **Purpose**: To evaluate the interactions between integrated components and ensure they work together as expected.
- **Approach**: Writing test cases that simulate data flow between frontend and backend systems, API integrations, and module interactions.
- **Examples**: Testing API endpoints for book search and retrieval, checkout processes involving payment gateways, and error handling across interconnected components.

System Testing

- **Purpose**: To validate the entire system against defined requirements and use cases.
- **Approach**: Conducting end-to-end tests that simulate real-world scenarios from user registration to order fulfillment.
- **Examples**: Testing user journeys including browsing books, adding items to cart, checking out, and receiving order confirmation; verifying email

User Acceptance Testing

- **Purpose**: To ensure the system meets user expectations and business goals.
- **Approach**: Involving stakeholders and target users to execute tests based on real- world scenarios and usability expectations.
- **Examples**: Evaluating ease of use, intuitiveness of navigation, responsiveness of search functionalities, and overall user satisfaction through feedback surveys and usability tests.

Validation

Performance Testing

- **Purpose**: To assess system responsiveness, scalability, and stability under various load conditions.
- **Approach**: Conducting load testing using tools like Apache JMeter to simulate concurrent user sessions, measuring response times, throughput, and resource usage.
- **Examples**: Stress testing to determine maximum user capacity, endurance testing to ensure sustained performance over time, and capacity planning based on performance metrics.

Security Testing

- **Purpose**: To identify and mitigate potential vulnerabilities that could compromise the confidentiality, integrity, or availability of the system.
- **Approach**: Performing penetration testing, vulnerability assessments, and code reviews to uncover security weaknesses.
- **Examples**: Testing for OWASP Top 10 vulnerabilities (e.g., SQL injection, XSS), ensuring secure authentication mechanisms, and validating encryption .

Compatibility Testing

- **Purpose**: To ensure consistent user experience across different browsers, devices, and operating systems.
- **Approach**: Testing on multiple configurations and platforms to verify compatibility and correct rendering of UI elements.
- **Examples**: Cross-browser testing on Chrome, Firefox, Safari, and Edge; device testing on desktops, tablets, and smartphones to validate responsive design and functionality.

Results and Analysis

Test Cases and Outcomes

- **Documentation**: Detailed documentation of test cases including expected outcomes, actual results, and discrepancies found.
- **Bug Tracking**: Using issue tracking systems (e.g., JIRA, Bugzilla) to log and prioritize identified issues for resolution.
- **Metrics and Reports**: Generating test reports summarizing test coverage, pass/fail status, and metrics such as defect density and test coverage percentage.

Validation Summary

- **Findings**: Summarizing the overall validation results, including successes, challenges encountered, and critical issues resolved.
- **Recommendations**: Providing recommendations for improvements based on testing outcomes, user feedback, and stakeholder input.
- **Lessons Learned**: Reflecting on lessons learned from testing processes, emphasizing the importance of thorough testing in delivering a high-quality product.

CHAPTER 3

SYSTEM REQUIREMENTS AND SPECIFICATIONS

3.1 Software Requirements

Software requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases

- Windows 7 or higher
- XAMMP control panel
- Note pad++
- My SQL

3.2 Hardware Requirements

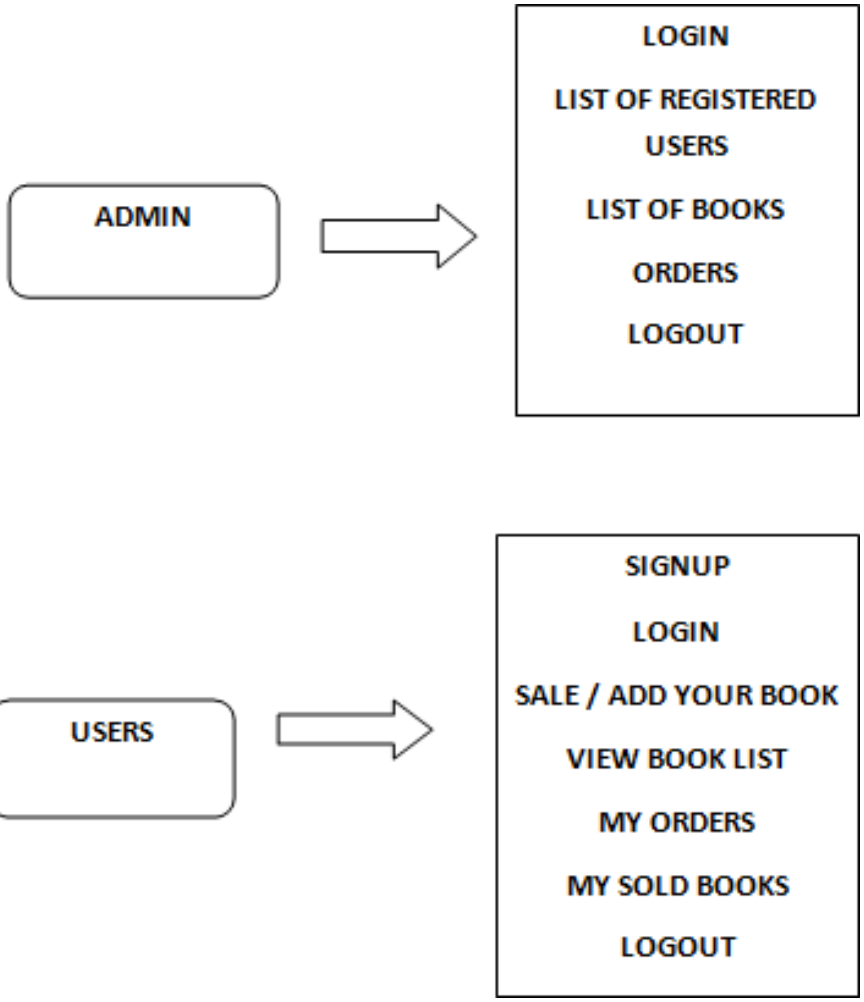
The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, a hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems

- Operating system: windows, Linux
- Processor: minimum intel i3
- Ram: minimum 4gb
- Hard disk: minimum 250gb

CHAPTER 4

SYSTEM DESIGN AND ARCHITECTURE

4.1 System Architecture



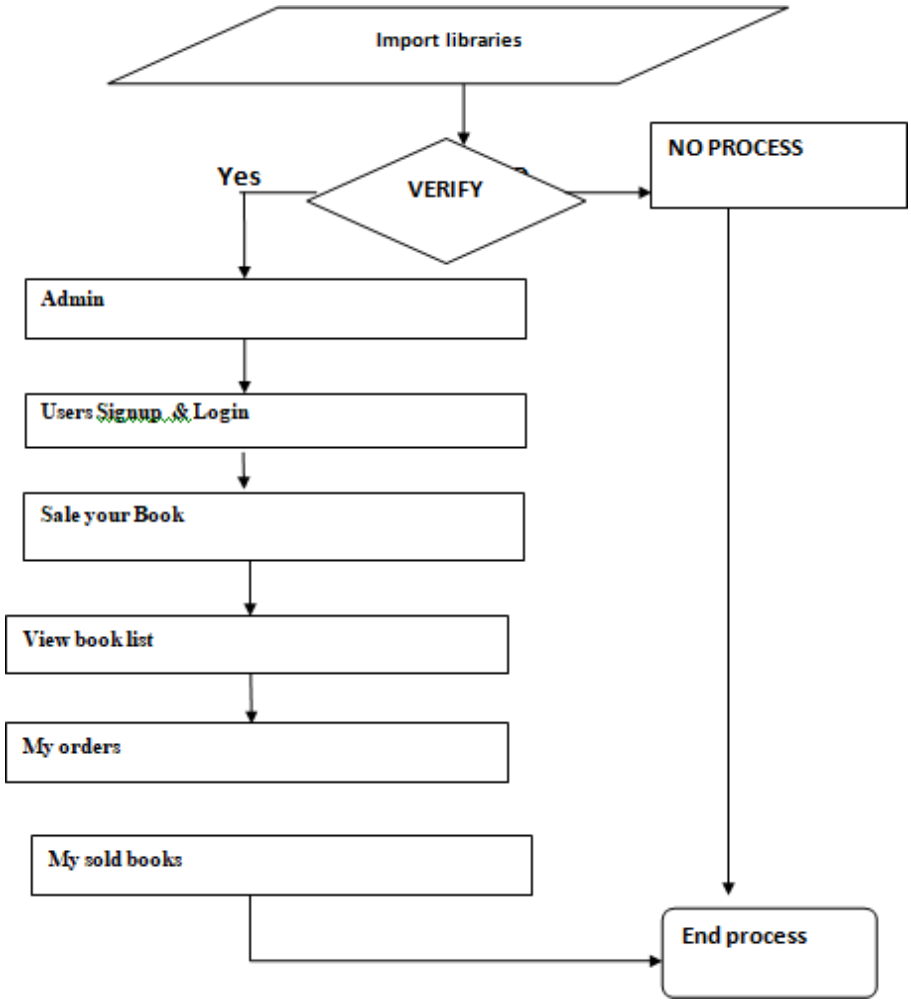
4.2 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

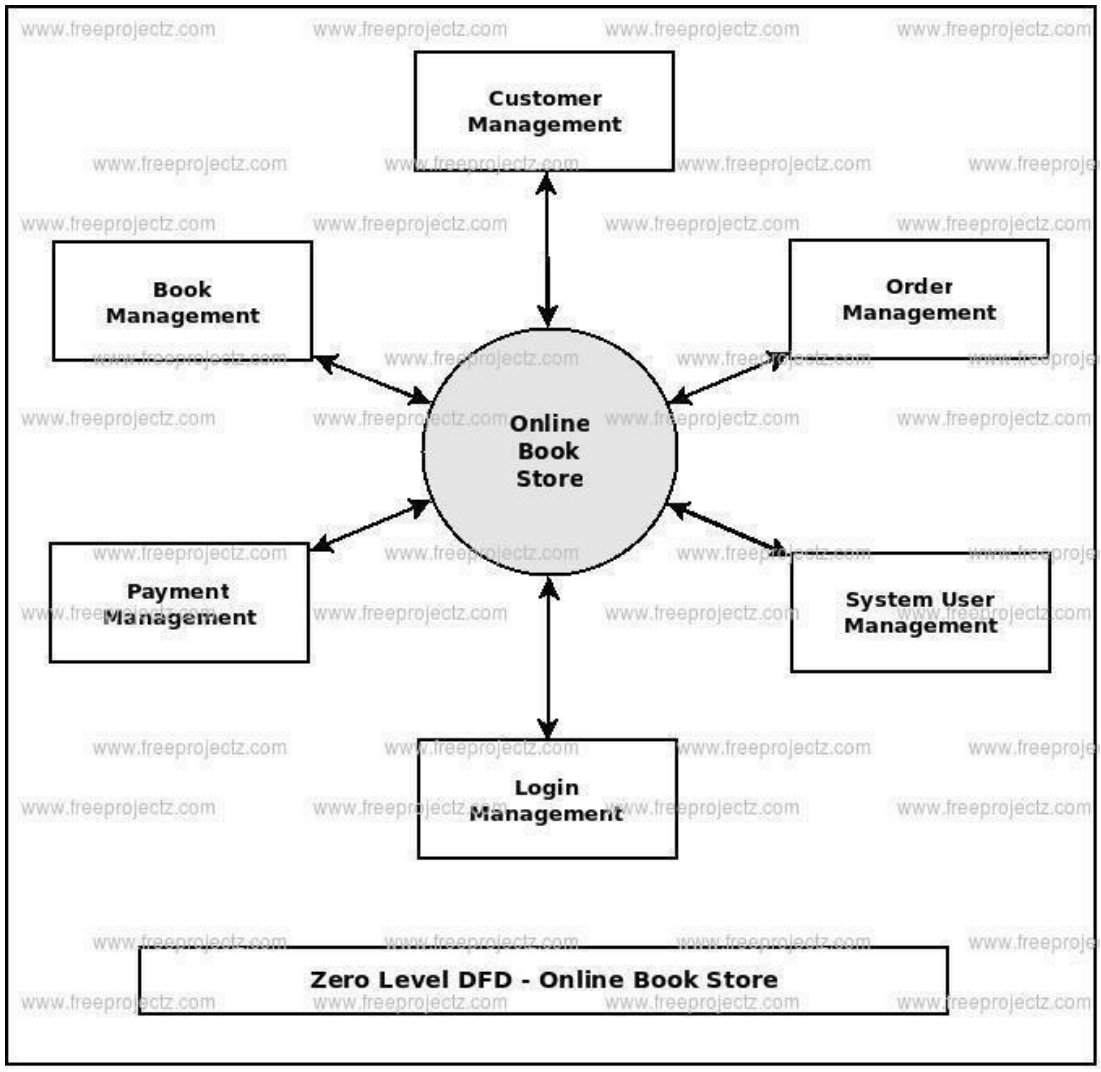
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



DATA FLOW DIAGRAM



4.3 Modules of project

Book Management Module

This module handles all the operations related to managing books in the bookstore.

Features:

Add New Books: Admin can add new books to the inventory by entering details like title, author, ISBN, price, category, description, and cover image.

Update Book Details: Admin can update the information of existing books, such as price ,description, and availability.

Delete Books: Admin can remove books from the inventory if they are no longer available or relevant.

Search and Browse Books: Users can search for books by title, author, or ISBN, and browse through categories and featured lists.

Implementation:

Database Design: Create tables for books with fields for title, author, ISBN, price, category, description, and cover image.

PHP Scripts: Develop scripts for adding, updating, deleting, and retrieving book records from the database.

Admin Interface: Build an admin panel for managing book inventory.

1. Shopping Cart Module

This module manages the shopping cart functionality, allowing users to add, view, and manage their selected books.

Features:

Add Books to Cart: Users can add books to their shopping cart from the book details page. View Cart Contents: Users can view the contents of their cart, including book details and total price.

Update Cart: Users can update the quantity of books in their cart or remove items they no longer want.

Save Cart for Later: Users can save their cart for future purchase sessions.

Implementation:

Session Management: Use PHP sessions to store cart information temporarily.

Database Design: Optionally, create a cart table in the database to persist cart data.

PHP Scripts: Develop scripts to handle adding, updating, and removing items from the cart.

2. Order Management Module

This module oversees the entire ordering process, from checkout to order confirmation and tracking.

Features:

Checkout Process: Users can proceed to checkout, where they provide shipping information and review their order.

Order Summary and Confirmation: Display a summary of the order for user confirmation before finalizing the purchase.

Payment Integration: Integrate with payment gateways to process transactions securely.

Order History and Tracking: Users can view their past orders and track the status of current orders.

PHP Scripts: Develop scripts for processing orders, handling payments, and updating order statuses.

Integration: Implement payment gateway APIs for processing payments.

3. Admin Panel Module

The admin panel is a central interface for managing the bookstore, including book inventory and orders.

Features:

Manage Books: Admin can view, add, edit, and delete books from the inventory.

View and Manage Orders: Admin can view order details, update order statuses, and handle customer inquiries.

Generate Reports: Admin can generate reports on sales, inventory, and user activity. Implementation:

Admin Authentication: Implement a secure login system for admins.

Admin Interface: Build an intuitive interface for managing books and orders.

Reporting Tools: Develop tools for generating and exporting reports.

4. Search and Filter Module

This module enhances the user experience by allowing efficient searching and filtering of books.

Features:

Search by Title, Author, ISBN: Users can search for books using keywords.

Filter by Category, Price Range, Rating: Users can narrow down search results using various filters.

Advanced Search Options: Provide additional parameters for more refined results.

Implementation:

Database Design: Optimize book table indexing for fast search queries.
PHP Scripts: Develop search and filter algorithms to retrieve and display results.
User Interface: Design search bars and filter options for the frontend.

5. Recommendation System Module

This module provides personalized book recommendations to users based on their preferences and behavior.

Features:

Personalized Recommendations: Suggest books based on user's browsing history and purchases.
Bestsellers and New Arrivals: Highlight popular and newly added books.
User-based and Item-based Recommendations: Use collaborative filtering techniques to recommend books.

Implementation:

Database Design: Create tables to store user interactions and book ratings.
Recommendation Algorithms: Implement algorithms for generating recommendations.
Frontend Integration: Display recommended books on the homepage and book details page.

6. Review and Rating Module

This module allows users to leave reviews and ratings for books, enhancing the credibility and social proof of products.

Features:

Submit Reviews and Ratings: Users can write reviews and rate books they have purchased.

View Reviews and Ratings: Display reviews and ratings on the book details page.

Moderate Reviews: Admin can moderate reviews to ensure quality and appropriateness.

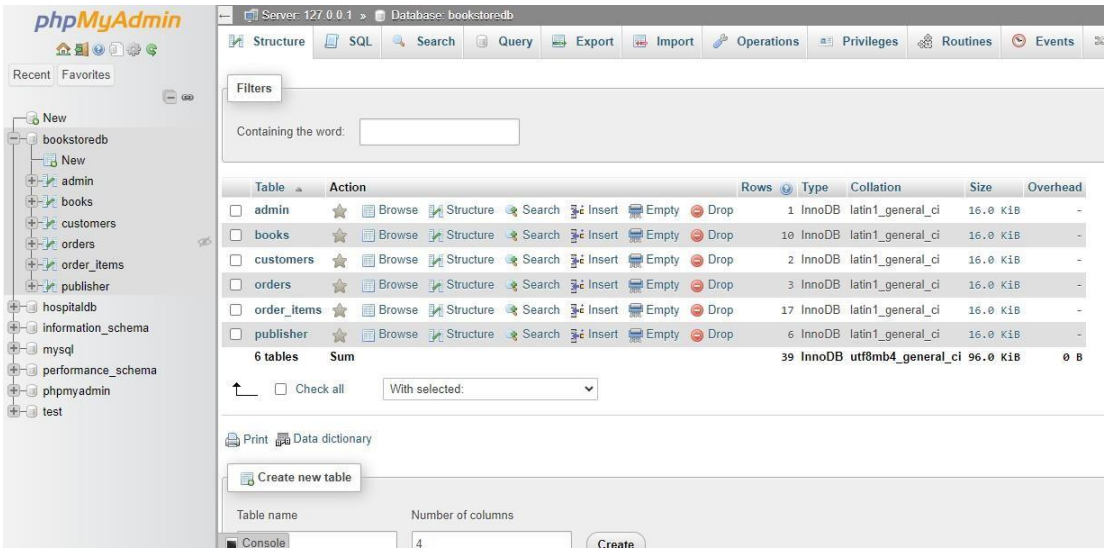
Implementation:

Database Design: Create tables for storing reviews and ratings.

PHP Scripts: Develop scripts for submitting, retrieving, and moderating reviews.

User Interface: Design review and rating forms and display sections.

4.4 Database Tables



CHAPTER 5

IMPLEMENTATION

5.1 Programming language

Frontend Development

HTML Templates and CSS Styling

- **Homepage**: Design and layout of the homepage showcasing featured books, promotional banners, and navigation links.
- **Book Listings**: Displaying books with details such as title, author, price, and cover image in a grid or list view.
- **Book Details Page**: Individual pages for each book with detailed descriptions, reviews, and related recommendations.
- **Shopping Cart:** Interface for viewing cart contents, updating quantities, and proceeding to checkout.

Responsive Design and Cross-Browser Compatibility

- Ensuring the website adapts seamlessly to various screen sizes and devices using responsive design principles.
- Testing compatibility across popular web browsers (Chrome, Firefox, Safari, Edge) to ensure consistent user experience.

JavaScript and Frontend Interactivity

- Implementing client-side interactivity for features such as dynamic search suggestions, AJAX-based book filtering, and interactive shopping cart updates.
- Using JavaScript frameworks/libraries (e.g., jQuery) for enhanced DOM manipulation and event handling.

Backend Development

PHP Framework Setup

- Choosing and configuring a PHP framework (e.g., Laravel, CodeIgniter) based on project requirements and developer familiarity.
- Setting up directory structure, routing, and environment configurations for MVC pattern implementation.

Database Design and Management

- Designing the database schema using MySQL Workbench or similar tools, focusing on tables for users, books, categories, orders, and reviews.
- Establishing relationships (one-to-many, many-to-many) between entities and defining primary keys, foreign keys, and constraints.

Model Development

- Creating PHP classes for database interactions (CRUD operations) using ORM (Object-Relational Mapping) techniques provided by the chosen framework.
- Implementing business logic such as book availability checks, order processing, and user authentication methods.

Controller Implementation

- Developing controllers to handle user inputs, process requests from frontend components, and interact with models to fetch or update data.
- Implementing middleware for authentication, authorization, and input validation to ensure security and data integrity.

User Authentication and Authorization

- Implementing secure user registration and login functionalities with password hashing and salting for enhanced security.

- Managing user sessions and implementing roles-based access control (RBAC) to restrict access to administrative features.

Book Management Features

- **Adding and Editing Books**: Form interfaces for administrators to add new books with details like title, author, price, and upload cover images.
- **Inventory Management**: Implementing features for updating book quantities, marking books as out of stock, and managing book categories.
- **Search and Filtering**: Implementing advanced search functionalities allowing users to search by title, author, category, and price range.

Shopping Cart and Checkout Process

- **Cart Functionality**: Implementing features to add/remove items from the cart, update quantities, and calculate subtotal and total prices dynamically.
- **Checkout Process**: Creating a multi-step checkout process with user information collection, shipping options, order summary review, and integration with payment gateways (e.g., PayPal, Stripe).

Admin Panel Development

- **Dashboard**: Overview of key metrics such as total sales, new user registrations, and inventory status.
- **User Management**: CRUD operations for managing user accounts, including roles and permissions management.
- **Order Management**: Interface for viewing orders, updating order statuses (e.g., processing, shipped), and generating order reports.

Security Measures and Data Protection

- **SSL/TLS Implementation**: Enabling secure communication between clients and the server to protect sensitive user data.

- **Input Validation and Sanitization**: Implementing server-side validation to prevent SQL injection, XSS attacks, and other security vulnerabilities.
- **Data Backup and Recovery**: Setting up regular database backups and implementing disaster recovery plans to ensure data integrity and availability.

Deployment and Testing

- **Deployment Strategy**: Deploying the application to a web server (e.g., Apache, Nginx) or cloud platform (e.g., AWS, Azure) following best practices for security and performance.
- **Testing Methods**: Conducting unit tests for individual components, integration tests for combined functionalities, and user acceptance tests to validate user workflows and functionality.

Performance Optimization

- **Caching Strategies**: Implementing caching mechanisms (e.g., Redis, Memcached) to store frequently accessed data and reduce database queries.
- **Code Optimization**: Identifying and optimizing performance bottlenecks such as slow database queries or inefficient algorithms.
- **Load Testing**: Using tools like Apache JMeter or LoadRunner to simulate concurrent user traffic and monitor system performance metrics.

5.2 Selection of Platform

The selection of a platform for developing the Online Book Store involves careful consideration of the technologies that will best meet the needs of the project. The LAMP stack (Linux, Apache, MySQL, PHP) has been chosen for its robustness, ease of use, and wide adoption in the web development community. Below is a detailed explanation of each component and why it was selected:

Linux

- **Open Source:** Linux is an open-source operating system, meaning it is free to use, modify, and distribute. This reduces the overall cost of the project.

- **Stability and Security:** Linux is known for its stability and security, making it a reliable choice for web servers. It has a strong community that regularly updates and patches the OS.

- **Compatibility:** Linux supports a wide range of hardware and software, making it a versatile choice for various development environments.

Apache

- **Widely Used:** Apache is one of the most widely used web servers in the world, which ensures a large community for support and a plethora of resources for troubleshooting and learning.

- **Flexibility:** Apache is highly configurable, allowing developers to customize its features and functionalities according to the specific needs of the project.

- **Modular Architecture:** Apache's modular architecture allows for the integration of various features as modules, which can be added or removed as needed, providing great flexibility in server management.

MySQL

- Relational Database Management System (RDBMS): MySQL is a powerful RDBMS that is essential for managing the structured data of the Online Book Store, such as user information, book details, and transaction records.

- Performance and Scalability: MySQL offers high performance and scalability, which is crucial for handling a large number of transactions and growing user base efficiently.

- Community Support: Being open-source, MySQL has a strong community and extensive documentation, making it easier to find solutions to any issues that may arise.

PHP

- Server-Side Scripting: PHP is a server-side scripting language specifically designed for web development. It is embedded within HTML, making it easy to integrate dynamic content into web pages.

- Ease of Learning and Use: PHP is relatively easy to learn and use, which can speed up the development process. It has extensive documentation and a large community for support.

- Compatibility with MySQL: PHP has built-in support for MySQL, making database interactions smooth and efficient. It also supports various other databases, providing flexibility in future enhancements.

Additional Considerations

Development Environment

- **IDE/Editor**: Visual Studio Code is recommended due to its robust features, extensions, and integrations that facilitate PHP and MySQL development. Other IDEs such as Php Storm or Sublime Text can also be used based on developer preference.

- **Version Control:** Git for version control ensures that changes in the codebase are tracked and managed efficiently. Platforms like GitHub or GitLab can be used for repository hosting and collaboration.

Hosting and Deployment

- **Cloud Services:** Cloud platforms like AWS, Digital Ocean, or Linode can be used for hosting the Linux server, providing scalability and reliability. These platforms offer various tools for managing and deploying applications.

- **Security:** Implementing SSL/TLS for secure data transmission, regular backups, and robust firewall configurations are essential for maintaining the security of the application.

By choosing the LAMP stack, the Online Book Store project leverages reliable, scalable, and well-supported technologies that are ideal for developing and deploying a robust web application.

5.2Methodology/Technology Used

5.3.1 MySQL

MySQL is a popular open-source relational database management system (RDBMS) that has gained widespread adoption for its performance, scalability, and ease of use. Here's a more detailed look into MySQL, its features, uses, and advantages:

Features of MySQL

1. **Relational Database Management System (RDBMS)**:
    - MySQL organizes data into tables, where each table consists of rows and columns. It supports relational operations, allowing tables to establish relationships with each other through keys (primary keys and foreign keys).
2. **Structured Query Language (SQL)**:
    - MySQL uses SQL as its primary language for interacting with databases. SQL enables users to create, retrieve, update, and delete data, as well as define and manipulate schema (database structure).
3. **Cross-Platform Compatibility**:
    - MySQL is compatible with various operating systems including Linux, Windows, macOS, FreeBSD, and others. This flexibility makes it suitable for a wide range of environments.
4. **Performance and Scalability**:
    - MySQL is known for its high performance and scalability. It employs various techniques such as indexing, caching, and query optimization to ensure fast data retrieval and processing, even with large datasets .

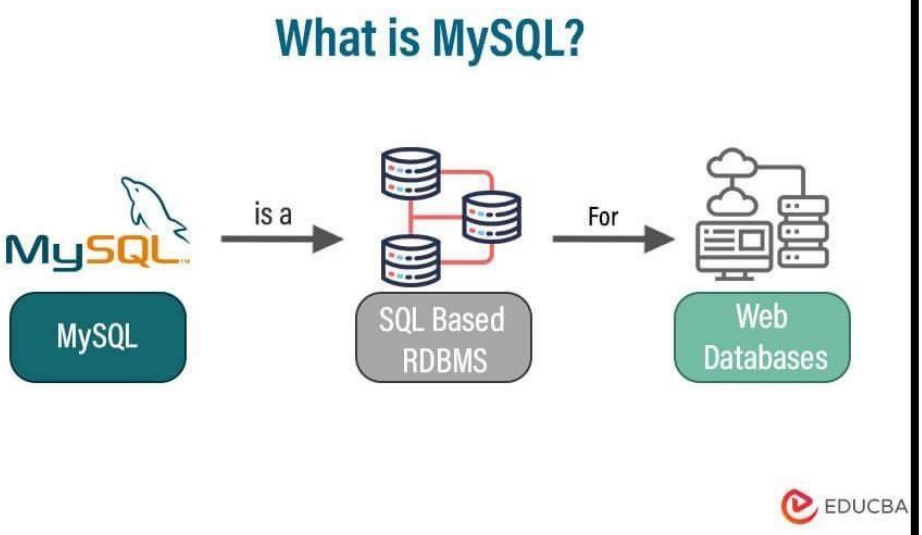
- MySQL provides robust security features including user authentication, access control (granting and revoking privileges), encryption for data in transit (SSL/TLS) and at rest, and auditing capabilities to track database activities.

5. **Storage Engines**:
   - MySQL supports multiple storage engines, each with its own characteristics and optimizations. The most commonly used storage engines include InnoDB (transactional), MyISAM (non-transactional), and MEMORY (in-memory).
6. **Replication and High Availability**:
   - MySQL offers built-in support for replication, allowing databases to be replicated across multiple servers for improved performance, scalability, and fault tolerance. It also supports clustering and other high availability solutions.
7. **Community Support and Development**:
   - MySQL has a large and active community of developers and users who contribute to its development, provide support through forums, and share resources such as tutorials, blogs, and extensions (like plugins and connectors).

Uses of MySQL

1. **Web Applications**:
   - MySQL is extensively used as the backend database management system for web applications. It integrates seamlessly with web development technologies such as PHP, Python, Ruby on Rails, Java, and others.

2. **Content Management Systems (CMS)**:
    - Many popular CMS platforms, including WordPress, Drupal, Joomla, and Magento, rely on MySQL to store and manage website content, user data, configurations, and more.
3. **E-commerce Platforms**:
    - MySQL is widely used in e-commerce applications to handle product catalogs, customer orders, transactional data, and inventory management. Its reliability and performance are crucial for ensuring a smooth shopping experience.
4. **Data Warehousing and Business Intelligence**:
    - MySQL is employed in data warehousing environments where large volumes of data are stored, processed, and analyzed to generate insights for decision-making. It supports complex queries and reporting requirements.
    -
5. **Mobile and IoT Applications**:
    - MySQL is suitable for mobile and IoT (Internet of Things) applications that require lightweight and scalable database solutions. It can handle the data storage and retrieval needs of mobile apps, syncing data between devices and servers.
6. **Telecommunications and Networking**:
    - MySQL finds applications in telecommunications and networking industries for managing network configurations, user profiles, billing information, and call detail records (CDRs).
7. **Education and Research**:
    - MySQL is used in educational institutions and research organizations for managing educational data, research databases, student information systems, and academic research projects.



XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends. The name XAMPP stands for:

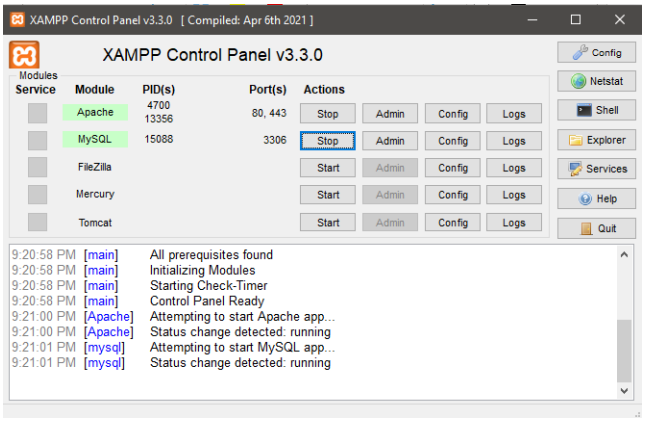
- **X**: Stands for cross-platform compatibility (works on various operating systems).
- **A**: Apache HTTP Server, the web server software included in the package.
- **M**: MySQL or MariaDB, the database management system.
- **P**: PHP, Perl, or Python, the programming languages used for dynamic web development.
- **P**: Perl or Python, depending on the package variant.

Uses of XAMPP:

1. **Local Web Development**: XAMPP is primarily used to set up a local web server environment on your computer. This environment mimics a live server, allowing developers to test and develop websites or web applications locally before deploying them to a live server.
2. **Testing and Debugging**: It provides a convenient way to test websites or applications without needing internet access. This is particularly useful for debugging and ensuring that everything works correctly before making it live.
3. **Learning and Education**: XAMPP is widely used in educational settings for teaching web development and programming languages like PHP and

MySQL/MariaDB. It provides a controlled environment where students can experiment with different web technologies without needing access to a remote server.

4. **Open-Source Projects**: Many open-source projects and CMS (Content Management Systems) like WordPress, Joomla, Drupal, etc., recommend XAMPP as a development environment. It simplifies the setup process and allows contributors to work on the project locally.
5. **Quick Setup**: XAMPP is easy to install and configure compared to setting up individual components (Apache, MySQL/MariaDB, PHP, etc.) separately. It provides a bundled package with all necessary components pre-configured, saving time and effort.
6. **Compatibility**: It works across different operating systems including Windows, macOS, and Linux, making it a versatile choice.



5.3.2 JavaScript

JavaScript is a programming language that is primarily used for creating interactive effects within web browsers. It's an essential tool for front-end web development, allowing developers to add dynamic content, interactivity, and functionality to websites.

Key Points about JavaScript:

1. **Client-Side Scripting**: JavaScript runs on the client side (in the user's web browser), enabling dynamic updates and interactive elements without requiring the web page to reload.
2. **Versatility**: It's not limited to web browsers anymore. With technologies like Node.js, JavaScript can also be used for server-side development, making it capable of handling full-stack web development.
3. **Interactivity**: JavaScript allows developers to respond to user actions (like clicks and keyboard input) and modify the webpage content accordingly. This interactivity enhances user experience by creating responsive and engaging web applications.
4. **Integration with HTML/CSS**: JavaScript works seamlessly with HTML (markup language) and CSS (styling language), allowing developers to manipulate HTML elements, style them dynamically, and control their behavior based on user interactions or events.
5. **Libraries and Frameworks**: There are numerous JavaScript libraries (like jQuery, React, Angular, Vue.js) and frameworks available that simplify complex tasks, speed up development, and provide reusable components for building modern web applications.
6. **Ecosystem**: JavaScript has a vibrant and active community, with extensive documentation, tutorials, and resources available for developers of all levels. This ecosystem continually evolves with new updates, features.

Simple Uses of JavaScript:

1. **Interactive Websites**:
    - JavaScript enables features like dropdown menus, forms that validate user input, sliders, and interactive maps—all without reloading the entire webpage.
2. **User Interface Enhancements**:
    - It powers things like animated graphics, scrolling effects, pop-up windows, and custom buttons that respond to user actions.

3. **Data Handling**:
    - JavaScript helps fetch and display data from servers dynamically, making it useful for live updates, news feeds, and real-time information.
4. **Browser Extensions**:
    - Developers use JavaScript to create extensions that add new features or customize browser behavior, enhancing user experience.
5. **Web Applications**:
    - JavaScript frameworks and libraries (like React, Angular, Vue.js) are used to build complex web applications, including social media platforms, online stores, and collaborative tools.
6. **Game Development**:
    - With HTML5 Canvas and libraries like Phaser.js, JavaScript can create browser-based games with animations, physics, and interactive elements.
7. **Server-Side Development**:
    - Node.js allows JavaScript to run on servers, handling backend tasks like database operations, authentication, and managing server requests.

5.3.3 PHP

PHP (Hypertext Preprocessor) is a server-side scripting language primarily used for web development but also capable of general-purpose programming.

Simple uses of PHP:

1. **Dynamic Websites**:
    - PHP is used to build websites that show different content to users based on their actions (like filling out forms or clicking buttons).
2. **Web Applications**:
    - Many web apps, from simple blogs to large e-commerce sites, use PHP for handling user data, managing content, and interacting with databases like MySQL.
3. **Form Handling**:
    - PHP processes form submissions, validates input (like checking if an email address is valid), and sends data to databases or other services.
4. **Content Management Systems (CMS)**:
    - Platforms like WordPress and Joomla are built with PHP, allowing users to easily manage and update website content.
5. **Server-Side APIs**:
    - PHP can create APIs that let different software systems communicate with each other, exchanging data in formats like JSON or XML.

5.3.4 HTML

HTML (Hypertext Markup Language) is the standard markup language used to create and structure web pages and web applications. Here's an overview of HTML.

uses:

- **Building Websites**:

- HTML forms the backbone of websites, providing the structure and content that users interact with. It defines the layout, text, images, links, and other elements that make up web pages.

- **Semantic Markup**:

- HTML uses semantic tags to describe the meaning and structure of content. This helps search engines understand and index web pages correctly, improving search engine optimization (SEO) for better visibility online.

- **Creating Forms**:

- HTML includes form elements (<form>, <input>, <textarea>, <select>, etc.) for collecting user input such as text, checkboxes, radio buttons, and dropdown menus. Submitted form data can be processed using server-side scripts like PHP.

- **Embedding Media**:

- HTML allows embedding images (<img>), videos (<video>), audio (<audio>), and other multimedia content directly into web pages. This enhances user engagement and enriches the browsing experience.

- **Linking and Navigation**:

- HTML provides anchor tags (<a>) to create hyperlinks that connect different web pages or sections within the same page. Links facilitate navigation between pages and help users find related information quickly.

- **Accessibility**:

- HTML supports accessibility features by including attributes like alt text for images (<img alt="description">), title attributes for tooltips, and semantic tags that improve screen reader compatibility for users with disabilities.

➢ Integration with CSS and JavaScript:

- HTML works together with CSS for styling (defining colors, fonts, layout) and JavaScript for adding interactivity (validating forms, creating animations, handling user events) to web pages.

5.3.5 CSS

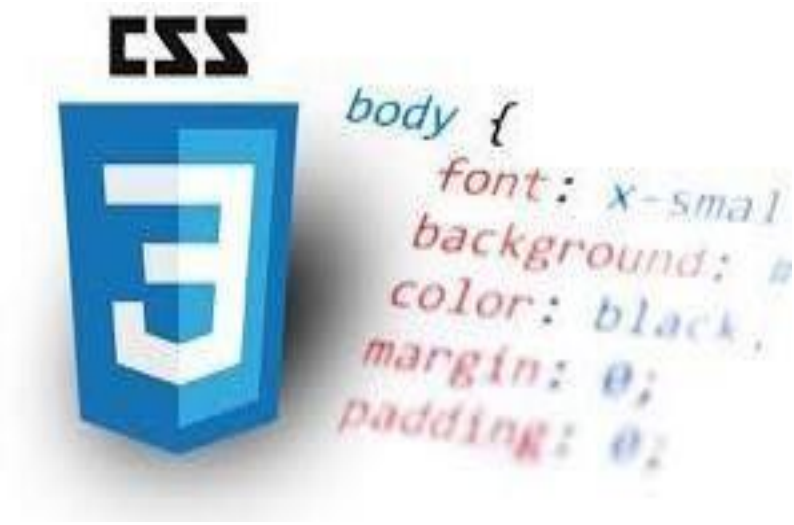
CSS stands for Cascading style sheets. It describes to the user how to display HTML elements on the screen in a proper format. CSS is the language that is used to style HTML documents. In simple words, cascading style sheets are a language used to simplify the process of making a webpage.

Uses of CSS:

1. **Styling Web Pages**:
    - CSS is essential for creating visually appealing and consistent web designs. It defines colors, fonts, spacing, backgrounds, borders, and other aesthetic properties that make web content attractive and readable.
2. **Responsive Design**:
    - CSS facilitates responsive web design, allowing web pages to adapt and display correctly on different devices and screen sizes (e.g., desktops, tablets, smartphones). Techniques like media queries adjust styles based on viewport dimensions.
3. **Layout Control**:
    - CSS provides layout control to arrange elements on a web page using techniques like floats, flexbox, and grid layout. This enables designers to create complex and responsive page layouts without relying solely on

HTML's structural elements.

4. **Accessibility**:
    - CSS supports accessibility by defining visual cues (like contrast ratios, text sizes, and spacing) that improve readability and usability for users with disabilities. It works alongside HTML to ensure content is accessible to all users.
5. **Animation and Effects**:
    - CSS3 introduces features for creating animations, transitions, and interactive effects without JavaScript. This includes animations of elements (e.g., fading in/out), transitions between states (e.g., hover effects), and transformations (e.g., scaling and rotating elements).
6. **Customization and Theming**:
    - CSS allows customization and theming of web applications and content management systems (CMS). Developers can create custom stylesheets or themes to match branding guidelines, preferences, or user preferences.



5.3.6 Bootstrap

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

- It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

Uses of Bootstrap

- Faster and Easier Web Development.
- It creates Platform-independent web pages.
- It creates Responsive Web-pages.
- It designed to be responsive to mobile devices too

How to use Bootstrap in a webpage:

There are two ways to include Bootstrap on the website.

- Include Bootstrap from the CDN link.
- Download Bootstrap from getbootstrap.com and use it.

CHAPTER 6

SCREENSHOTS AND CODING

Establishing Database Connection:

```
<?php
	function db_connect(){
		$conn = mysqli_connect("localhost", "root", "", "bookstoredb");
		if(!$conn){
			echo "Can't connect database " . mysqli_connect_error($conn);
			exit;
		}
		return $conn;
	}

	function select4LatestBook($conn){
		$row = array();
		$query = "SELECT book_isbn, book_image FROM books ORDER BY
book_isbn DESC";
		$result = mysqli_query($conn, $query);
		if(!$result){
		   echo "Can't retrieve data " . mysqli_error($conn);
		   exit;
		}
		for($i = 0; $i < 4; $i++){
			array_push($row, mysqli_fetch_assoc($result));
		}
		return $row;
	}

	function getBookByIsbn($conn, $isbn){
		$query = "SELECT book_title, book_author, book_price FROM books WHERE
book_isbn = '$isbn'";
		$result = mysqli_query($conn, $query);
```

```
                if(!$result){
                        echo "Can't retrieve data " . mysqli_error($conn);
                        exit;
                }
                return $result;
        }

        function getOrderId($conn, $customerid){
                $query = "SELECT orderid FROM orders WHERE customerid = '$customerid'";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "retrieve data failed!" . mysqli_error($conn);
                        exit;
                }
                $row = mysqli_fetch_assoc($result);
                return $row['orderid'];
        }

function insertIntoOrder($conn, $customerid, $total_price, $date, $ship_name,
$ship_address, $ship_city, $ship_zip_code, $ship_country){
                $query = "INSERT INTO orders VALUES
                ('" . $customerid . "', '" . $total_price . "', '" . $date . "', '" . $ship_name . "', '" .
$ship_address . "', '" . $ship_city . "', '" . $ship_zip_code . "', '" . $ship_country . "')";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "Insert orders failed " . mysqli_error($conn);
                        exit;
                }
        }

function getbookprice($isbn){
$conn = db_connect();
                $query = "SELECT book_price FROM books WHERE book_isbn = '$isbn'";
                $result = mysqli_query($conn, $query);
```

```
                if(!$result){
                        echo "get book price failed! " . mysqli_error($conn);
                        exit;
                }
                $row = mysqli_fetch_assoc($result);
                return $row['book_price'];
        }

        function getCustomerId($name, $address, $city, $zip_code, $country){
                $conn = db_connect();
                $query = "SELECT customerid from customers WHERE
                name = '$name' AND
                address= '$address' AND
                city = '$city' AND
                zip_code = '$zip_code' AND
                country = '$country'";
                $result = mysqli_query($conn, $query);
                // if there is customer in db, take it out
                if($result){
                        $row = mysqli_fetch_assoc($result);
                        return $row['customerid'];
                } else {
                        return null;
                }
}

        function setCustomerId($name, $address, $city, $zip_code, $country){
                $conn = db_connect();
                $query = "INSERT INTO customers VALUES
                        ('" . $name . "', '" . $address . "', '" . $city . "', '" . $zip_code . "', '" .
$country . "')";

                $result = mysqli_query($conn, $query);
                if(!$result){
```

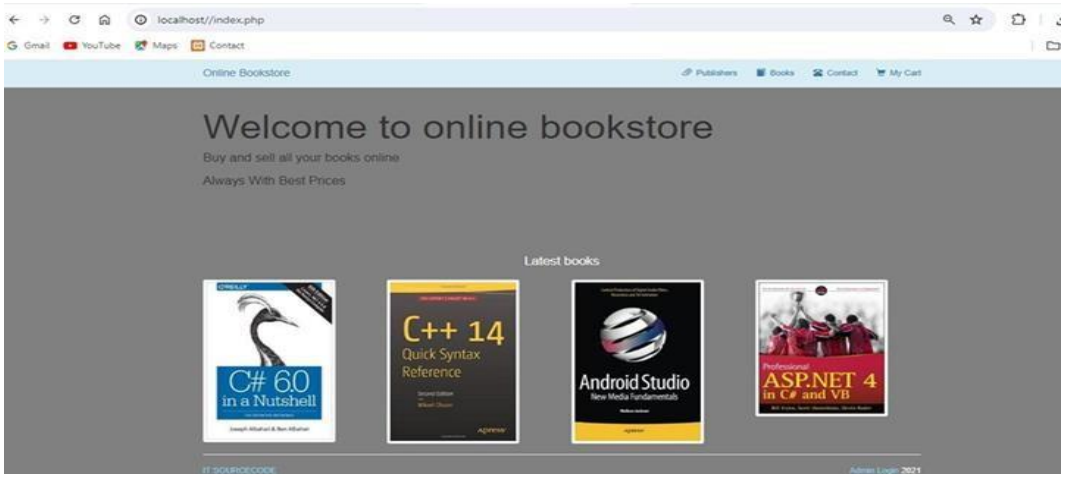
```
                        echo "insert false !" . mysqli_error($conn);
                        exit;
                }
                $customerid = mysqli_insert_id($conn);
                return $customerid;
        }

        function getPubName($conn, $pubid){
                $query = "SELECT publisher_name FROM publisher WHERE publisherid =
'$pubid'";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "Can't retrieve data " . mysqli_error($conn);
                        exit;
                }
                if(mysqli_num_rows($result) == 0){
                        echo "Empty books ! Something wrong! check again";
                        exit;
                }

                $row = mysqli_fetch_assoc($result);
                return $row['publisher_name'];
        }

        function getAll($conn){
                $query = "SELECT * from books ORDER BY book_isbn DESC";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "Can't retrieve data " . mysqli_error($conn);
                        exit;
                }
                return $result;
        }
?>
```

Homepage

(User Module) Code:

```
<?php
  session_s
  tart();
  $count = 0;
  $title = "Index";
  require_once            "./template/header.php";
  require_once
  "./functions/database_functions.php";
  $conn = db_connect();
  $row = select4LatestBook($conn);
?>
<!-- Example row of columns -->
    <p class="lead text-center text-muted" style="color: white">Latest books</p>
    <div class="row">
      <?php foreach($row as $book) { ?>
        <div class="col-md-3">
                <a href="book.php?bookisbn=<?php echo $book['book_isbn']; ?>">
          <img class="img-responsive img-thumbnail" src="./bootstrap/img/<?php echo
$book['book_image']; ?>">
        </a>
         </div>
      <?php } ?>
    </div>
<?php
  if(isset($conn))
  {mysqli_close($conn);}
  require_once
  "./template/footer.php";?>
```

Output:



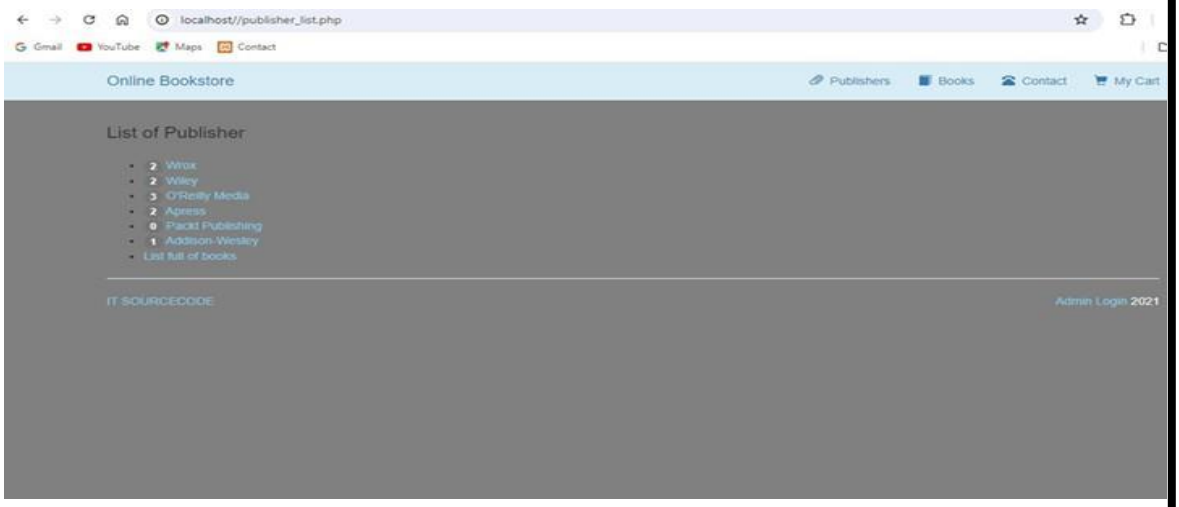
Publishers

ModuleCode:

```
<?php
        session_start();
        require_once "./functions/database_functions.php";
        $conn = db_connect();

        $query = "SELECT * FROM publisher ORDER BY publisherid";
        $result = mysqli_query($conn,
        $query);if(!$result){
                echo "Can't retrieve data " .
                mysqli_error($conn);exit;
        }
        if(mysqli_num_rows($result) == 0){
                echo "Empty publisher ! Something wrong! check again";
                exit;}
        $title = "List Of
        Publishers"; require
        "./template/header.php";
?>
```

```
        <p class="lead">List of Publisher</p>
        <ul>
        <?php
                while($row = mysqli_fetch_assoc($result)){
                        $count = 0;
                        $query = "SELECT publisherid FROM books";
                        $result2 = mysqli_query($conn, $query);
                        if(!$result2){
                                echo "Can't retrieve data " . mysqli_error($conn);
                                exit;
                        }
                        while ($pubInBook = mysqli_fetch_assoc($result2)){
                                if($pubInBook['publisherid'] == $row['publisherid']){
                                        $count++;
                                }
                        }
        ?>
                <li>
                        <span class="badge"><?php echo $count; ?></span>
                   <a style="color: skyblue" href="bookPerPub.php?pubid=<?php echo
$row['publisherid']; ?>"><?php echo $row['publisher_name']; ?></a>
                </li>
        <?php } ?>
                <li>
                        <a style="color: skyblue" href="books.php">List full of books</a>
                </li>
        </ul>
<?php
        mysqli_close($conn);
        require "./template/footer.php";
?>
```

Output:



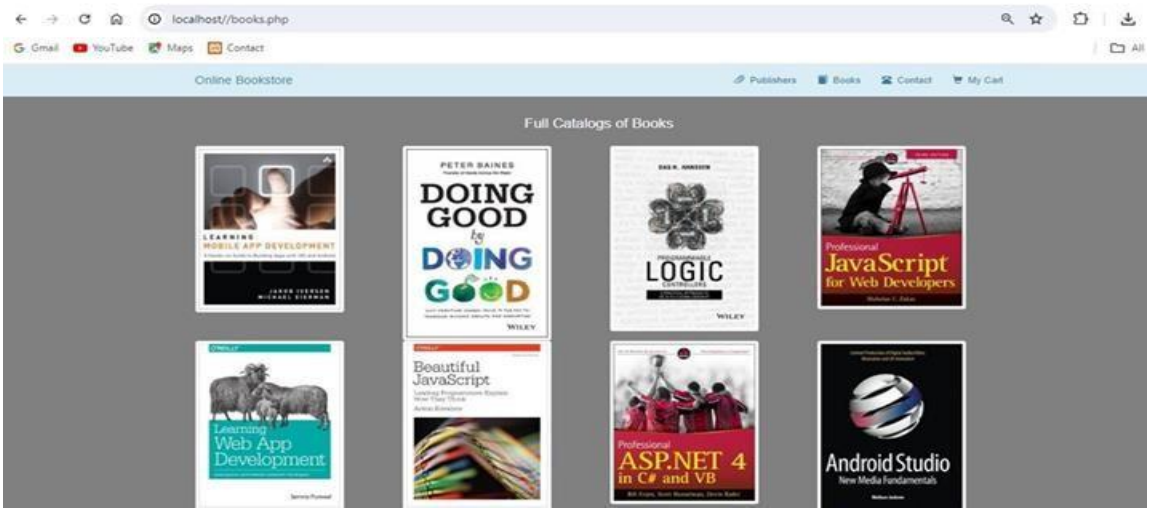
Listof

Books code

```
<?php
 session_
 start();
 $count = 0;
 // connecto database
 require_once "./functions/database_functions.php";
 $conn = db_connect();

 $query = "SELECT book_isbn, book_image FROM books";
 $result = mysqli_query($conn,
 $query);if(!$result){
  echo "Can't retrieve data " .
  mysqli_error($conn);exit;
 }
 $title = "Full Catalogs of
 Books"; require_once
 "./template/header.php";
?>
 <p style="color: white" class="lead text-center text-muted">Full Catalogs of Books</p>
```

```
   <?php for($i = 0; $i < mysqli_num_rows($result); $i++){ ?>
     <div class="row">
       <?php while($query_row = mysqli_fetch_assoc($result)){ ?>
         <div class="col-md-3">
           <a href="book.php?bookisbn=<?php echo $query_row['book_isbn']; ?>">
             <img class="img-responsive img-thumbnail" src="./bootstrap/img/<?php echo
$query_row['book_image']; ?>">
           </a>
         </div>
        <?php
         $count++;
         if($count >= 4){
            $count = 0;
            break;
          }
         } ?>
     </div>
<?php
     }
  if(isset($conn)) { mysqli_close($conn); }
  require_once "./template/footer.php";
?>
```

Output:



Contact

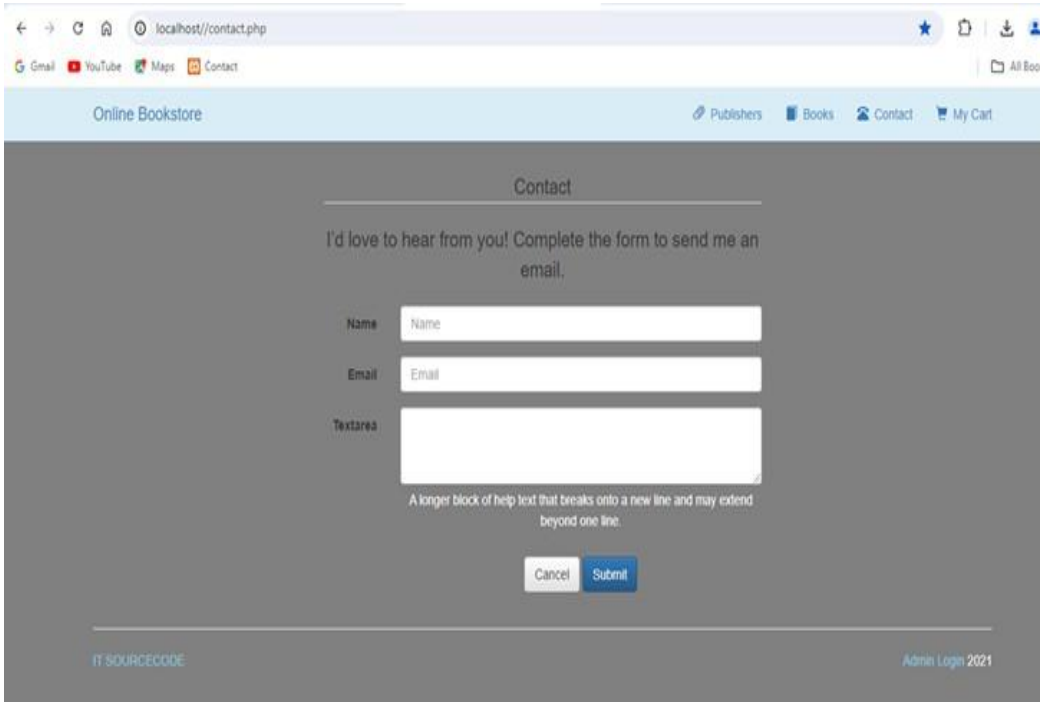
ModuleCode:

```
<?php
  $title = "Contact";
  require_once "./template/header.php";
?>
   <div class="row">
      <div class="col-md-3"></div>
                <div class="col-md-6 text-center">
                        <form class="form-horizontal">
          <fieldset>
                                    <legend>Contact</legend>
                                    <p class="lead">I'd love to hear from you! Complete the
                                    form to send me an email.<p>

                                    <div class="form-group">

                                    <label for="inputName" class="col-lg-2 control-

                                    Label">Name</label>
```

```
                                        <div class="col-lg-10">
                                        <input type="text" class="form-control"
id="inputName" placeholder="Name">
                                        </div>
                                    </div>
                                    <div class="form-group">
                                        <label for="inputEmail" class="col-lg-2 control-
label">Email</label>
                                        <div class="col-lg-10">
                                        <input type="text" class="form-control"
id="inputEmail" placeholder="Email">
            </div>
                                    </div>
                                    <div class="form-group">
                                        <label for="textArea" class="col-lg-2 control-
label">Textarea</label>
                                        <div class="col-lg-10">
                                        <textarea class="form-control" rows="3"
id="textArea"></textarea>
                                        <span style="color: white" class="help-block">A longer
block of help text that breaks onto a new line and may extend beyond one line.</span>
            </div>
            </div>
                                    <div class="form-group">
                                        <div class="col-lg-10 col-lg-offset-2">
                                        <button type="reset" class="btn btn-
default">Cancel</button>
                                        <button type="submit" class="btn btn-
primary">Submit</button>
                                    </div>
                                      </div>
                                </fieldset>
                            </form>
                    </div>
```

```
<div class="col-md-3"></div>
        </div>
    <?php
     require_once "./template/footer.php";
    ?>
```

Output:



My Cart

PageCode:

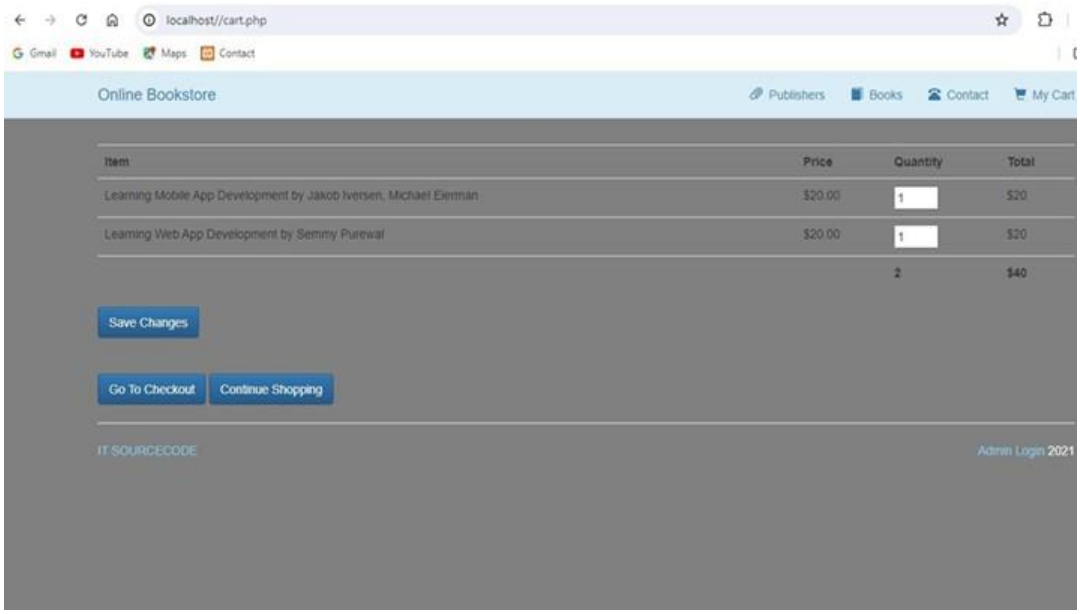
```
    <?php
            // the shopping cart needs sessions, to start one
/* Array of session(
    cart => array (book_isbn (get from $_POST['book_isbn'])
    => number of books
                        ),
                        items => 0,
                        total_price => '0.00'
```

```
        )
*/
session_start();
require_once "./functions/database_functions.php";
require_once "./functions/cart_functions.php";

// book_isbn got from form post method, change this place later.
if(isset($_POST['bookisbn'])){
        $book_isbn = $_POST['bookisbn'];
}

if(isset($book_isbn)){
        //      new     iem     selected
        if(!isset($_SESSION['cart'])){
                // $_SESSION['cart'] is associative array that bookisbn => qty
                $_SESSION['cart'] = array();

                $_SESSION['total_items'] = 0;
                $_SESSION['total_price'] = '0.00';
        }

        if(!isset($_SESSION['cart'][$book_isbn])){
                $_SESSION['cart'][$book_isbn] = 1;
        } elseif(isset($_POST['cart'])){
                $_SESSION['cart'][$book_isbn]++;
                unset($_POST);
        }
}

// if save change button is clicked , change the qty of each bookisbn
if(isset($_POST['save_change'])){
```

```
                foreach($_SESSION['cart'] as $isbn =>$qty){
                        if($_POST[$isbn] == '0'){
                                unset($_SESSION['cart']["$isbn"]);
                        } else {
                                $_SESSION['cart']["$isbn"] = $_POST["$isbn"];
                        }
                }
        }

        // print out header here
        $title = "Your shopping cart";
        require "./template/header.php";

 if(isset($_SESSION['cart']) && (array_count_values($_SESSION['cart']))){
$_SESSION['total_price'] = total_price($_SESSION['cart']);
 $_SESSION['total_items'] = total_items($_SESSION['cart']);
 ?>
        <form action="cart.php" method="post">
                <table class="table">
                        <tr>
                                <th>Item</th>
                                <th>Price</th>
                                <th>Quantity</th>
                                <th>Total</th>
                        </tr>
                        <?php
                        foreach($_SESSION['cart'] as $isbn => $qty){
                                        $conn = db_connect();
                                        $book          =
 mysqli_fetch_assoc(getBookByIsbn($conn,  $isbn));
                        ?>
```

```
                        <tr>
                                <td><?php echo $book['book_title'] . " by " .
$book['book_author']; ?></td>
                                <td><?php echo "$" . $book['book_price']; ?></td>
                                <td><input type="text" value="<?php echo $qty; ?>"
size="2" name="<?php echo $isbn; ?>"></td>
                                <td><?php echo "$" . $qty * $book['book_price']; ?></td>
                        </tr>
                        <?php } ?>
<tr>
                        <th> </th>
                        <th> </th>
                        <th><?php echo $_SESSION['total_items']; ?></th>
                        <th><?php echo "$" . $_SESSION['total_price']; ?></th>
</tr>
</table>
                <input type="submit" class="btn btn-primary" name="save_change"
value="Save Changes">
        </form>
        <br/><br/>
        <a href="checkout.php" class="btn btn-primary">Go To Checkout</a>
        <a href="books.php" class="btn btn-primary">Continue Shopping</a>
<?php
        } else {
                echo "<p class=\"text-warning\">Your cart is empty! Please make sure
you add some books in it!</p>";
        }
        if(isset($conn)){ mysqli_close($conn); }
        require_once "./template/footer.php";
?>
```

Output:

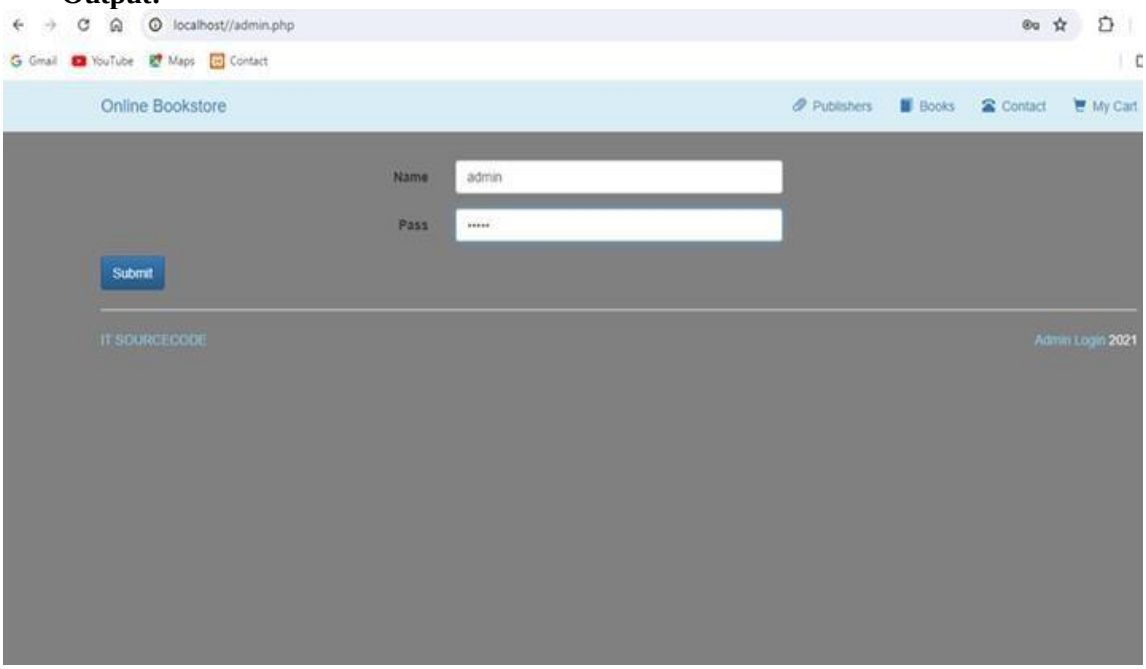


Admin Module Login Page

```
<?php
$title = "Administration
section"; require_once
"./template/header.php";
?>
<form class="form-horizontal" method="post" action="admin_verify.php">
<div class="form-group">
<label for="name" class="control-label col-md-4">Name</label>
<div class="col-md-4">
<input type="text" name="name" class="form-control">
</div>
</div>
<div class="form-group">
<label for="pass" class="control-label col-md-4">Pass</label>
<div class="col-md-4">
```

```
<input type="password" name="pass" class="form-control">
</div></div>
</div><input type="submit" name="submit" class="btn btn-primary">
<input type="submit" name="submit" class="btn btn-primary">
</form>
<?php
require_once "./template/footer.php";
?>
```

Output:



Book Description

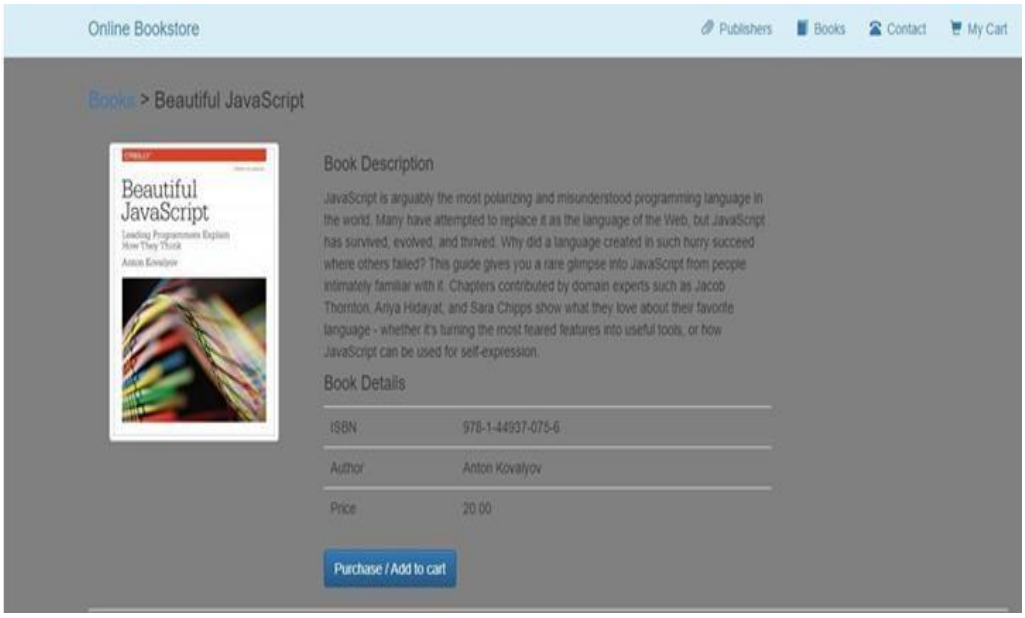
PageCode:

```
<?php
 session_
 start();
 $book_isbn = $_GET['bookisbn'];
 // connecto database
 require_once "./functions/database_functions.php";
 $conn = db_connect();
```

```
 $query = "SELECT * FROM books WHERE book_isbn = '$book_isbn'";
 $result = mysqli_query($conn, $query);
 if(!$result){
  echo "Can't retrieve data " . mysqli_error($conn);
  exit;
 }

 $row = mysqli_fetch_assoc($result);
 if(!$row){
  echo "Empty book";
  exit;
 }
 $title    =    $row['book_title'];
 require "./template/header.php";
?>
    <!-- Example row of columns -->
    <p class="lead" style="margin: 25px 0"><a href="books.php">Books</a> > <?php echo
$row['book_title']; ?></p>
    <div class="row">
      <div class="col-md-3 text-center">
        <img class="img-responsive img-thumbnail" src="./bootstrap/img/<?php echo
$row['book_image']; ?>">
      </div>
      <div class="col-md-6">
        <h4>Book Description</h4>
        <p><?php echo $row['book_descr']; ?></p>
        <h4>Book Details</h4>
        <table class="table">
          <?php foreach($row as $key => $value){
           if($key == "book_descr" || $key == "book_image" || $key == "publisherid" || $key ==
"book_title"){
             continue;
           }
           switch($key){
```

```
            case "book_isbn":
              $key = "ISBN";
              break;
            case "book_title":
              $key = "Title";
              break;
            case "book_author":
              $key = "Author";
              break;
            case "book_price":
              $key = "Price";
              break;
          }
        ?>
        <tr>
          <td><?php echo $key; ?></td>
          <td><?php echo $value; ?></td>
        </tr>
        <?php }
          if(isset($conn)) {mysqli_close($conn); }
        ?>
      </table>
      <form method="post" action="cart.php">
        <input type="hidden" name="bookisbn" value="<?php echo $book_isbn;?>">
        <input type="submit" value="Purchase / Add to cart" name="cart" class="btn btn-
primary">
      </form>
        </div>
    </div>
<?php
  require "./template/footer.php";
?>
```

Output:



Address

FormCode:

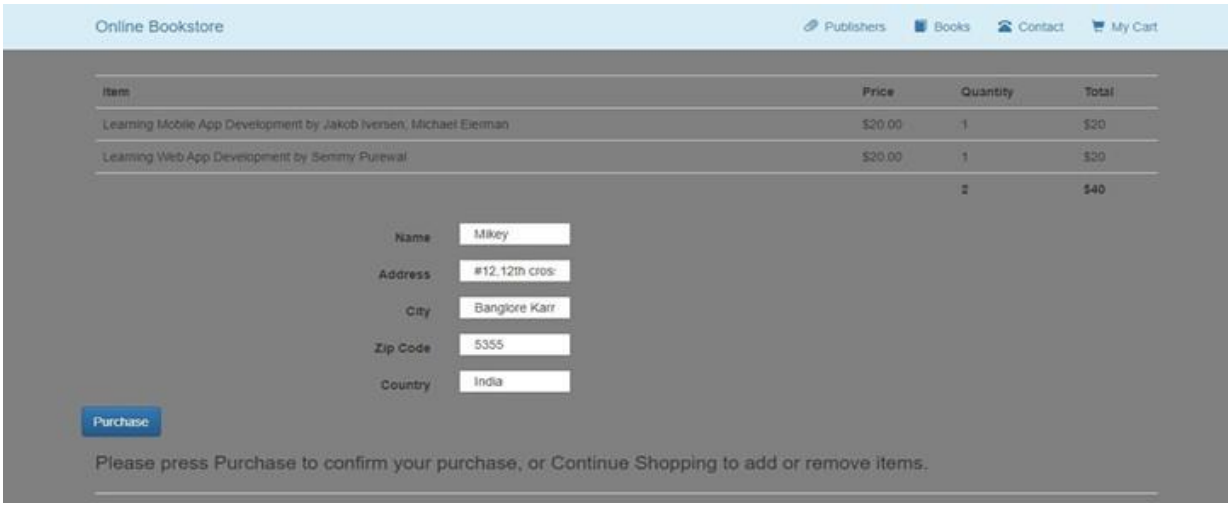
```
<?php
        // the shopping cart needs sessions, to start one
        /*
                Array of session(
                        cart => array (
                                book_isbn (get from $_GET['book_isbn']) => number of books
                        ),
                        items => 0,
                        total_price => '0.00'
                )
        */
        session_start();
        require_once "./functions/database_functions.php";
        // print out header here
```

```
        $title = "Checking out";
        require "./template/header.php";

        if(isset($_SESSION['cart']) && (array_count_values($_SESSION['cart']))){
?>
<table class="table">
<tr>
                        <th>Item</th>
                        <th>Price</th>
                <th>Quantity</th>
                <th>Total</th>
            </tr>
                <?php
                        foreach($_SESSION['cart'] as $isbn => $qty){
                                $conn = db_connect();
                                $book = mysqli_fetch_assoc(getBookByIsbn($conn,
$isbn));
                        ?>
                <tr>
                        <td><?php echo $book['book_title'] . " by " . $book['book_author'];
?></td>
                        <td><?php echo "$" . $book['book_price']; ?></td>
                        <td><?php echo $qty; ?></td>
                        <td><?php echo "$" . $qty * $book['book_price']; ?></td>
                </tr>
                <?php } ?>
                <tr>
                        <th> </th>
                        <th> </th>
                        <th><?php echo $_SESSION['total_items']; ?></th>
                        <th><?php echo "$" . $_SESSION['total_price']; ?></th>
</tr>
        </table>
        <form method="post" action="purchase.php" class="form-horizontal">
```

```
                <?php if(isset($_SESSION['err']) && $_SESSION['err'] == 1){ ?>
                        <p class="text-danger">All fields have to be filled</p>
                        <?php } ?>
                <div class="form-group">
                        <label for="name" class="control-label col-md-4">Name</label>
                        <div class="col-md-4">
                                <input type="text" name="name" class="col-md-4" class="form-
control">
                        </div>
                </div>
                <div class="form-group">
                        <label for="address" class="control-label col-md-4">Address</label>
                        <div class="col-md-4">
                                <input type="text" name="address" class="col-md-4"
class="form-control">
                        </div>
</div>
                <div class="form-group">
                        <label for="city" class="control-label col-md-4">City</label>
                        <div class="col-md-4">
                                <input type="text" name="city" class="col-md-4" class="form-
control">
                        </div>
                </div>
                <div class="form-group">
                        <label for="zip_code" class="control-label col-md-4">Zip Code</label>
                        <div class="col-md-4">
                                <input type="text" name="zip_code" class="col-md-4"
class="form-control">
                        </div>
</div>
                <div class="form-group">
                        <label for="country" class="control-label col-md-4">Country</label>
                        <div class="col-md-4">
```

```
                                    <input type="text" name="country" class="col-md-4"
class="form-control">
                                </div>
</div>
                <div class="form-group">
                        <input type="submit" name="submit" value="Purchase" class="btn btn-
primary">
                </div>
        </form>
        <p class="lead">Please press Purchase to confirm your purchase, or Continue Shopping
to add or remove items.</p>
<?php
        } else {
                echo "<p class=\"text-warning\">Your cart is empty! Please make sure you add
some books in it!</p>";
        }
        if(isset($conn)){ mysqli_close($conn); }
        require_once "./template/footer.php";
?>
```

Output:



Payment Gateway

Code:

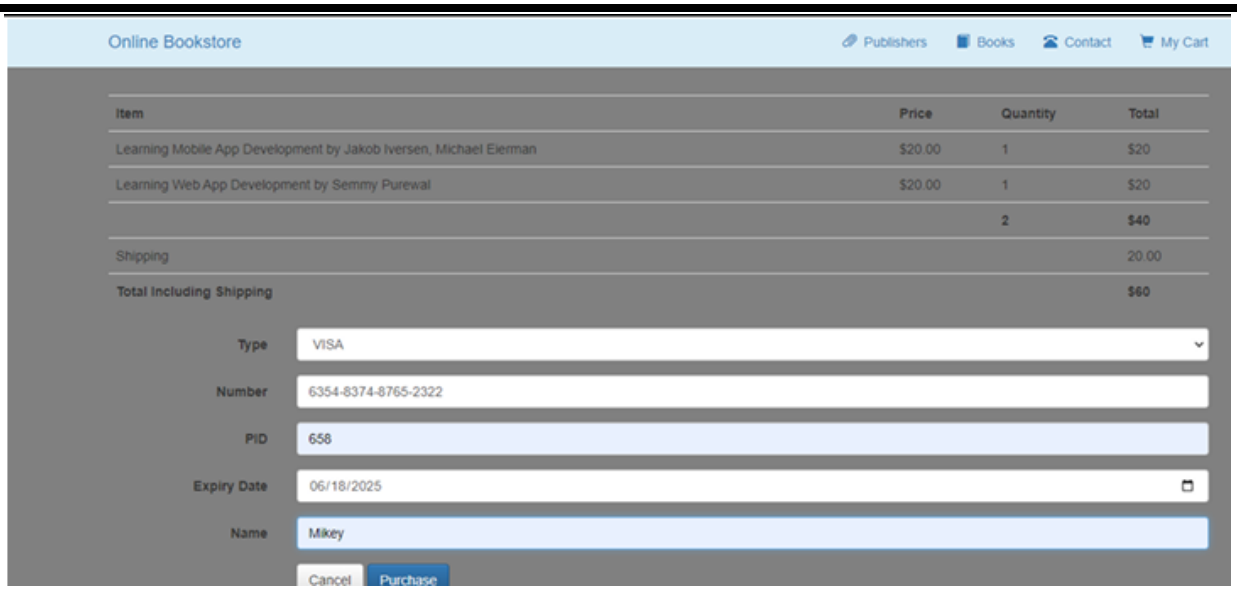
```
<?php
	session_start();
	$_SESSION['err'] = 1;
	foreach($_POST as $key => $value){
		if(trim($value) == ''){
			$_SESSION['err'] = 0;
		}
		break;
	}

	if($_SESSION['err'] == 0){
		header("Location: checkout.php");
	} else {
		unset($_SESSION['err']);
	}



	$_SESSION['ship']	=	array();
	foreach($_POST as $key => $value){
		if($key != "submit"){
			$_SESSION['ship'][$key] = $value;
		}
	}
	require_once "./functions/database_functions.php";
	// print out header here
	$title = "Purchase";
	require "./template/header.php";
	// connect database
	if(isset($_SESSION['cart']) && (array_count_values($_SESSION['cart']))){
?>
	<table class="table">
```

```
<tr>
                        <th>Item</th>
                        <th>Price</th>
                <th>Quantity</th>
                <th>Total</th>
            </tr>
                <?php
                        foreach($_SESSION['cart'] as $isbn => $qty){
                                $conn = db_connect();
                                $book = mysqli_fetch_assoc(getBookByIsbn($conn,
$isbn));
                        ?>
                <tr>
                        <td><?php echo $book['book_title'] . " by " . $book['book_author'];
?></td>
                        <td><?php echo "$" . $book['book_price']; ?></td>
                        <td><?php echo $qty; ?></td>
                        <td><?php echo "$" . $qty * $book['book_price']; ?></td>
                </tr>
                <?php } ?>
                <tr>
                        <th> </th>
                        <th> </th>
                        <th><?php echo $_SESSION['total_items']; ?></th>
                        <th><?php echo "$" . $_SESSION['total_price']; ?></th>
                </tr>
                <tr>
                        <td>Shipping</td>
                        <td> </td>
                        <td> </td>
                        <td>20.00</td>
                </tr>
                <tr>
                        <th>Total Including Shipping</th>
```

```
                        <th> </th>
                        <th> </th>
                        <th><?php echo "$" . ($_SESSION['total_price'] + 20); ?></th>
                </tr>
        </table>
        <form method="post" action="process.php" class="form-horizontal">
                <?php if(isset($_SESSION['err']) && $_SESSION['err'] == 1){ ?>
                <p class="text-danger">All fields have to be filled</p>
                <?php } ?>
    <div class="form-group">
        <label for="card_type" class="col-lg-2 control-label">Type</label>
        <div class="col-lg-10">
                <select class="form-control" name="card_type">
                <option value="VISA">VISA</option>
                <option value="MasterCard">MasterCard</option>
                <option value="American Express">American Express</option>
                </select>
        </div>
    </div>
    <div class="form-group">
        <label for="card_number" class="col-lg-2 control-label">Number</label>
        <div class="col-lg-10">
                <input type="text" class="form-control" name="card_number">
        </div>
    </div>
    <div class="form-group">
        <label for="card_PID" class="col-lg-2 control-label">PID</label>
        <div class="col-lg-10">
      <input type="text" class="form-control" name="card_PID">
        </div>
    </div>
    <div class="form-group">
        <label for="card_expire" class="col-lg-2 control-label">Expiry Date</label>
        <div class="col-lg-10">
```

```
<input type="date" name="card_expire" class="form-control">
          </div>
        </div>
        <div class="form-group">
          <label for="card_owner" class="col-lg-2 control-label">Name</label>
          <div class="col-lg-10">
<input type="text" class="form-control" name="card_owner">
          </div>
        </div>
        <div class="form-group">
          <div class="col-lg-10 col-lg-offset-2">
                    <button type="reset" class="btn btn-default">Cancel</button>
                    <button type="submit" class="btn btn-primary">Purchase</button>
          </div>
 </div>
</form>
          <p class="lead">Please press Purchase to confirm your purchase, or Continue Shopping
 to add or remove items.</p>
 <?php
          } else {
                    echo "<p class=\"text-warning\">Your cart is empty! Please make sure you add
 some books in it!</p>";
          }
          if(isset($conn)){ mysqli_close($conn); }
          require_once "./template/footer.php";
 ?>
```

Output:



Adding New Book

Code:

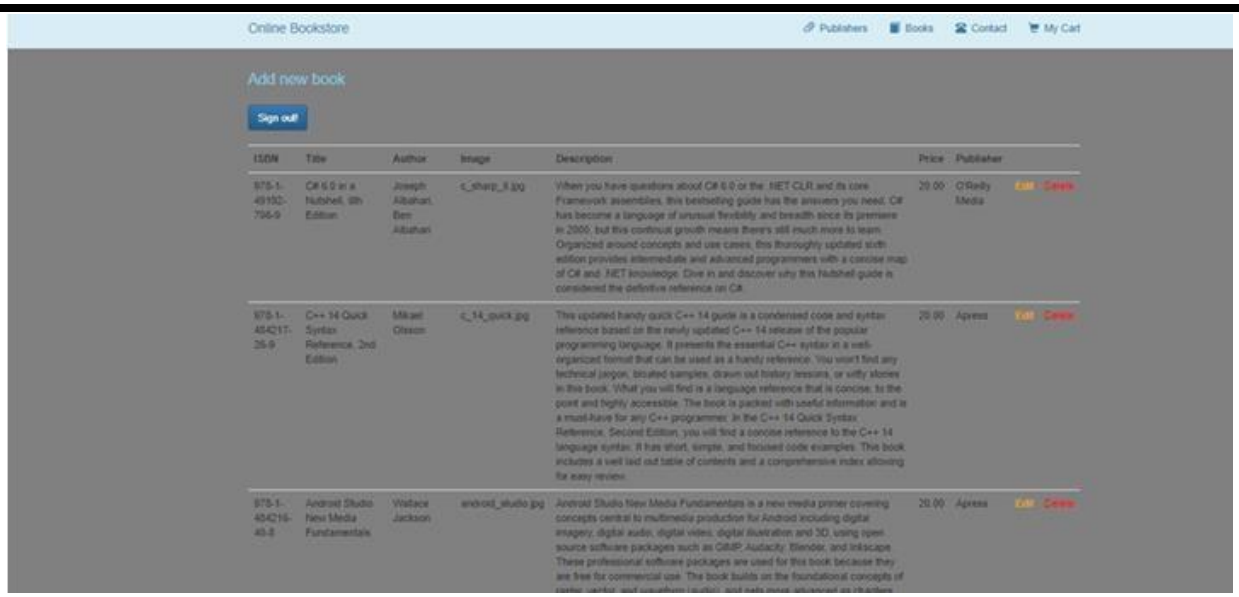
```
<?php
        session_start();
        require_once "./functions/admin.php";
        $title = "Add new book";
        require "./template/header.php";
        require "./functions/database_functions.php";
        $conn = db_connect();

        if(isset($_POST['add'])){
                $isbn = trim($_POST['isbn']);
                $isbn = mysqli_real_escape_string($conn, $isbn);

                $title = trim($_POST['title']);
                $title = mysqli_real_escape_string($conn, $title);

                $author = trim($_POST['author']);
```

```
                $author = mysqli_real_escape_string($conn, $author);

                $descr = trim($_POST['descr']);
                $descr = mysqli_real_escape_string($conn, $descr);

                $price = floatval(trim($_POST['price']));
                $price = mysqli_real_escape_string($conn, $price);

                $publisher = trim($_POST['publisher']);
                $publisher = mysqli_real_escape_string($conn, $publisher);

                // add image
                if(isset($_FILES['image']) && $_FILES['image']['name'] != ""){
                        $image = $_FILES['image']['name'];
                        $directory_self = str_replace(basename($_SERVER['PHP_SELF']), '',
$_SERVER['PHP_SELF']);
                        $uploadDirectory = $_SERVER['DOCUMENT_ROOT'] .
$directory_self . "bootstrap/img/";
                        $uploadDirectory .= $image;
                        move_uploaded_file($_FILES['image']['tmp_name'], $uploadDirectory);
                }

                // find publisher and return pubid
                // if publisher is not in db, create new
                $findPub = "SELECT * FROM publisher WHERE publisher_name =
'$publisher'";
                $findResult = mysqli_query($conn, $findPub);
                if(!$findResult){
                        // insert into publisher table and return id
                        $insertPub = "INSERT INTO publisher(publisher_name) VALUES
('$publisher')";
                        $insertResult = mysqli_query($conn, $insertPub);
                        if(!$insertResult){
                                echo "Can't add new publisher " . mysqli_error($conn);
```

```
        exit;
                        }
                        $publisherid = mysql_insert_id($conn);
                } else {
                        $row = mysqli_fetch_assoc($findResult);
                        $publisherid = $row['publisherid'];
                }


                $query = "INSERT INTO books VALUES ('" . $isbn . "', '" . $title . "', '" .
$author . "', '" . $image . "', '" . $descr . "', '" . $price . "', '" . $publisherid . "')";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "Can't add new data " . mysqli_error($conn);
                        exit;
                } else {
                        header("Location: admin_book.php");
                }
}
?>
<form method="post" action="admin_add.php" enctype="multipart/form-data">
        <table class="table">
        <tr>
                                <th>ISBN</th>
                                <td><input type="text" name="isbn"></td>
                        </tr>
                        <tr>
                                <th>Title</th>
                                <td><input type="text" name="title" required></td>
                        </tr>
                        <tr>
                                <th>Author</th>
                                <td><input type="text" name="author" required></td>
                        </tr>
```

```
                        <tr>
                                <th>Image</th>
                                <td><input type="file" name="image"></td>
                        </tr>
                        <tr>
                                <th>Description</th>
                                <td><textarea name="descr" cols="40"
rows="5"></textarea></td>
                        </tr>
                        <tr>
                                <th>Price</th>
                                <td><input type="text" name="price" required></td>
                        </tr>
                        <tr>
                                <th>Publisher</th>
                                <td><input type="text" name="publisher" required></td>
                        </tr>
                </table>
                <input type="submit" name="add" value="Add new book" class="btn btn-
primary">
                <input type="reset" value="cancel" class="btn btn-default">
        </form>
        <br/>
<?php
        if(isset($conn)) {mysqli_close($conn);}
        require_once "./template/footer.php";
?>
```

Output:



Books Editing:

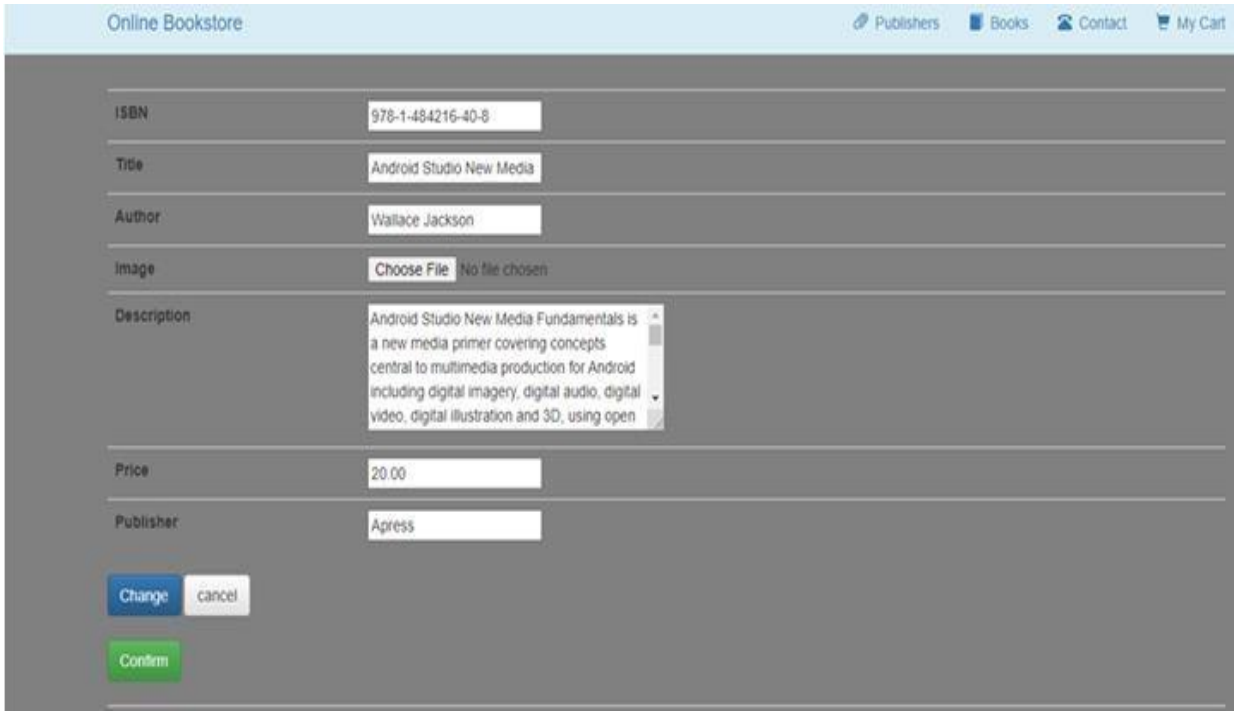
Code:

```
<?php
        // if save change happen
        if(!isset($_POST['save_change'])){
                echo "Something wrong!";
                exit;
        }
        $isbn = trim($_POST['isbn']);
        $title = trim($_POST['title']);
        $author = trim($_POST['author']);
        $descr = trim($_POST['descr']);
        $price = floatval(trim($_POST['price']));
        $publisher = trim($_POST['publisher']);

        if(isset($_FILES['image']) && $_FILES['image']['name'] != ""){
                $image = $_FILES['image']['name'];
                $directory_self = str_replace(basename($_SERVER['PHP_SELF']), '',
                $_SERVER['PHP_SELF']);
                $uploadDirectory = $_SERVER['DOCUMENT_ROOT'] . $directory_self .
"bootstrap/img/";
```

```
                $uploadDirectory
                =$image; move_uploaded_file($_FILES['image']['tmp_name'],
                $uploadDirectory);
        }

        require_once("./functions/database_functions.php");
        $conn = db_connect();

        // if publisher is not in db, create new
        $findPub = "SELECT * FROM publisher WHERE publisher_name = '$publisher'";
        $findResult = mysqli_query($conn, $findPub);
        if(!$findResult){
                // insert into publisher table and return id
                $insertPub = "INSERT INTO publisher(publisher_name) VALUES
('$publisher')";
                $insertResult = mysqli_query($conn, $insertPub);
                if(!$insertResult){
                        echo "Can't add new publisher " . mysqli_error($conn);
                        exit;
                }
        }

        $query = "UPDATE books SET
        book_title      =       '$title',
        book_author     =      '$author',
        book_descr      =       '$descr',
        book_price      =      '$price'";
        if(isset($image)){
                $query .= ", book_image='$image' WHERE book_isbn = '$isbn'";
        } else {
                $query .= " WHERE book_isbn = '$isbn'";
        }
```

```
        // two cases for fie , if file submit is on => change a lot
        $result = mysqli_query($conn, $query);


        if(!$result){
                echo "Can't update data " . mysqli_error($conn);
                exit;
        } else {
                header("Location: admin_edit.php?bookisbn=$isbn");
        }
?>
```

Output:



Order Confirmed:

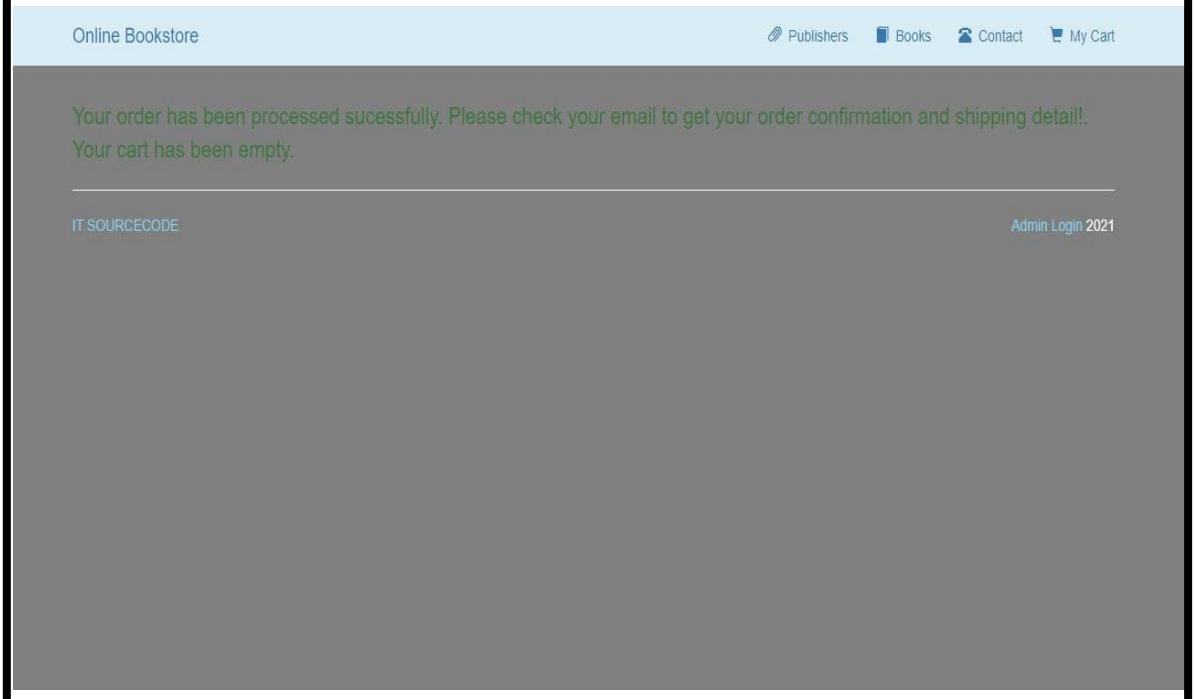
Code:

```
<?php
        session_start();

        $_SESSION['err'] = 1;
        foreach($_POST as $key => $value){
                if(trim($value) == ''){
                        $_SESSION['err'] = 0;
                }
                break;
        }

        if($_SESSION['err'] == 0){
                header("Location: purchase.php");
        } else {
                unset($_SESSION['err']);
        }

        require_once "./functions/database_functions.php";
        // print out header here
        $title = "Purchase Process";
        require "./template/header.php";
        // connect database
        $conn = db_connect();
        extract($_SESSION['ship']);

        // validate post section
        $card_number = $_POST['card_number'];
        $card_PID = $_POST['card_PID'];
```

```
        $card_expire = strtotime($_POST['card_expire']);
        $card_owner = $_POST['card_owner'];

        // find customer
        $customerid = getCustomerId($name, $address, $city, $zip_code, $country);
        if($customerid == null) {
                // insert customer into database and return customerid
                $customerid = setCustomerId($name, $address, $city, $zip_code, $country);
        }
        $date = date("Y-m-d H:i:s");
        insertIntoOrder($conn, $customerid, $_SESSION['total_price'], $date, $name, $address,
$city, $zip_code, $country);
        // take orderid from order to insert order items
        $orderid  =  getOrderId($conn,  $customerid);
        foreach($_SESSION['cart'] as $isbn => $qty){
                $bookprice = getbookprice($isbn);
                $query = "INSERT INTO order_items VALUES
                ('$orderid', '$isbn', '$bookprice', '$qty')";
                $result = mysqli_query($conn, $query);
                if(!$result){
                        echo "Insert value false!" . mysqli_error($conn2);
                        exit;
                }
        }
        session_unset();
?>
        <p class="lead text-success">Your order has been processed sucessfully. Please check
your email to get your order confirmation and shipping detail!.
        Your cart has been empty.</p>
<?php
        if(isset($conn)){
```

```
                mysqli_close($conn);
        }
        require_once "./template/footer.php";
?>
```

Output:



CHAPTER 7

CONCLUSION

The development of the online bookstore using PHP and MySQL has resulted in a robust e-commerce platform with essential features such as user registration, secure browsing, efficient search, shopping cart management, and secure checkout. The project successfully implemented responsive design principles, ensuring seamless user experience across devices. Rigorous testing validated functionality, security measures, and compatibility, while lessons learned emphasized best practices in MVC architecture, security protocols, and user-centric design. Future enhancements could include advanced search capabilities, social media integration, product diversification, and enhanced analytics. This project serves as a foundation for scalable, user-friendly e-commerce solutions, reflecting our commitment to quality, security, and continuous improvement in web development.

FUTURE WORK:

Moving forward, the online bookstore platform can enhance its capabilities by implementing advanced search functionalities, integrating social media features for increased engagement, and expanding its product offerings to include digital books and related merchandise. Personalized recommendation systems based on user preferences, improved analytics for better insights, and mobile optimization or a dedicated app for enhanced accessibility on mobile devices are crucial next steps. Internationalization efforts to support multi-language options and localization, continuous security updates, and initiatives for community engagement and sustainability will further enrich user experience and maintain competitiveness in the evolving landscape of e-commerce.

REFERENCES

Certainly! Here are some references that could be cited for the future work outlined for the online bookstore project:

1. **Advanced Search Functionalities:** Reference to articles or books on implementing advanced search algorithms in web applications.

2. **Social Media Integration:** Articles or documentation from social media platforms' developer resources on integrating social sharing and user-generated content.

3. **Product Diversification:** E-commerce industry reports or case studies on expanding product offerings to include digital goods and related merchandise.

4. **Personalization and Recommendation Systems:** Research papers or industry white papers on personalized recommendation algorithms in e-commerce.

5. **Analytics:** Resources on analytics tools and methodologies for e-commerce platforms, such as Google Analytics documentation or relevant studies.

6. **Mobile Optimization:** Articles or guidelines from mobile development resources on optimizing web applications for mobile devices or developing native apps.

7. **Internationalization**: Resources on internationalization and localization strategies for web applications, possibly from web development guides or localization industry reports.

8. **Security Updates:** Cybersecurity reports or guidelines on best practices for web application security and compliance with data protection regulations.