Booksmart Online Book Selling Web Application

Anna George
Nithin Koruth Varghese
Shan Shibu Thomas
Vyshnavi Shaji

Department Of Computer Applications

Saintgits College Of Engineering(Autonomous)
Kottukulam Hills, Pathamuttom P.O., Kottayam 686532November 2024

ABSTRACT

In this project, we develop Online Book Buying and Selling Portal . This project is like an e- bookstore website where books can be bought from the comfort of home through the Internet. A user can buy the books or can upload their books to resell them on the online. An online bookstore is a virtual store on the Internet where customers can browse the catalog and select books of interest. User can select many books and those books stored in cart. At checkout time, the items in the shopping cart will be presented as an order. At that time, more information will be needed to complete the transaction. Usually, the customer will be asked to fill the basic details or select a billing address, a shipping address, a shipping option, and payment option such as cash on delivery or online payment etc.If user have bookstore and he wand to publish there book store online he can easily publish there store throw this website.

Tools used - HTML, CSS, JAVASCRIPT, PHP ,MYSQL ,

EMAIL.JS ,PHP MAILER and BOOTSTRAP

Modules:

- User Module
- Customer Module

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION TO THE PROJECT

The 'BOOKSMART' The Online Book Selling Web Application has been developed to address the challenges of traditional book-selling systems. This software aims to eliminate and, in some cases, reduce the difficulties faced by existing systems. Moreover, this system is tailored to meet the specific needs of the company, enabling smooth and effective operations.The system comprises two main modules: the customer module and the seller module.

Customer Module: This module allows users to search for available books based on criteria such as book title, author, genre, and price range. Users can also read reviews, add books to their cart, and proceed with secure payments.

User Module: This module provides sellers with a user-friendly dashboard where they can manage their inventory, view and confirm orders, and update book details. Sellers can also track sales, manage promotions, and interact with customers.

1.2 ORGANIZATION PROFILE

Founded by a group of pioneering educators, we at Saintgits College of Engineering (Autonomous) seek to expose young minds to the world of technology and encourage all-round development of the mind. Our team of dedicated and caring faculty works with the student to reach academic excellence through a scientifically devised methodology. The high quality of our graduates, many of who have been university rank holders, and our distinctive results and placements are proof of the great distance we have travelled in 20 years. Saintgits College of Engineering has been granted Autonomous status by the University Grants Commission (UGC), making the institution one among the first three Engineering Colleges in Kerala to achieve this coveted status. The status was conferred in recognition of the academic excellence, expert faculty, industry connect, placement records, extracurricular activities and state of the art infrastructure offered by the institution. Saintgits is the only unaided institution in Kerala with 7 NBA accredited programmes.

1.3 OBJECTIVES OF THE PROJECT

The primary objective of the Old Book Selling Web Application is to provide an efficient, user-friendly, and secure platform for purchasing books online. The system aims to simplify the book buying process for customers by

offering features such as a comprehensive book catalog, personalized user accounts, and a seamless shopping experience. Additionally, it seeks to streamline the management of book sales and inventory for sellers, ensuring that books are easily categorized, tracked, and available for purchase.

1.4 Purpose, Scope, and Applicability of the project

Purpose:

The purpose of the Old Book Selling Web Application is to create a convenient and accessible platform for users to browse, purchase, and manage books online. This system aims to simplify the book-buying process by providing a user-friendly interface where customers can search for books, add them to a cart, and complete purchases without leaving their home. Additionally, it serves as a tool for book retailers or sellers to manage their inventory, track sales, and interact with customers through features like order history and account management. The system enhances the overall shopping experience by offering a centralized online solution for both customers and book sellers.

Applicability:

The Old Book Selling Web Application can be effectively utilized across various contexts where there is a need to sell books online. Independent bookstores can leverage the system to expand their reach, sell books online, and manage orders and inventory more efficiently. Publishers and distributors can use the platform to sell books directly to customers, offering discounts or special promotions. Academic and educational platforms, such as educational institutions or online learning platforms, can utilize the system to sell textbooks, study materials, and other educational content. Additionally, the system can be integrated as a specialized store within larger e-commerce platforms, allowing books to be part of a broader range of products, thereby enhancing the overall online shopping experience.

Scope of the Project

The scope of the **Online Book Selling System** includes several key features that ensure a complete, functional online bookstore. These features include:

1. **Book Catalog**: A comprehensive list of books with details such as title, author, price, and description. Books can be organized into categories for easy browsing.

2. **User Authentication**: The system supports user registration and login, allowing customers to create accounts, manage their personal details, and track orders.
3. **Shopping Cart and Checkout**: Users can add books to their cart, modify quantities, and proceed to a secure checkout page.
4. **Order Management**: The system allows users to view their order history and manage pending or completed orders.
5. **Admin Panel (Optional)**: An administrative interface for managing books, users, and orders, providing an essential tool for the store owner to update inventory and track sales.

CHAPTER-2REQUIREMENTS AND ANALYSIS

Problem Definition- Existing System

The current online book selling systems face several challenges that compromise the overall customer experience and operational efficiency for sellers. One of the primary issues is the **lack of personalization and flexibility** in terms of book recommendations, availability, and delivery options. Many existing platforms rely on generic search algorithms or do not offer customized recommendations based on the user's reading history or preferences. As a result, customers often find themselves overwhelmed with irrelevant book options, or worse, unable to find the specific titles they're looking for. This limitation in personalization not only makes the shopping experience less enjoyable but also discourages customers from exploring other books they may enjoy, ultimately leading to missed sales opportunities.

2.2 Hardware Specification for Development, Implementation

When developing and implementing a bus booking system, the hardware specifications should be carefully considered to ensure optimal performance and stability. The following are some hardware recommendations for such a system:

1. Processor: A powerful processor is essential for the smooth functioning of the bus booking system. An Intel Core i5 or i7 processor would be ideal for this purpose.
2. RAM: Sufficient RAM is necessary to ensure the system can handle multiple concurrent users without slowing down. A minimum of 8 GB RAM is recommended, but 16 GB or more would be better.
3. Storage: Adequate storage space is necessary for storing the system files, databases, and backups. A solid-state

drive (SSD) would be the best option for faster read/write speeds and improved system performance.

4. Network card: A high-speed network card is important for fast data transfer and seamless connectivity. A gigabit Ethernet card would be a good choice.

2.3Preliminary Product Description

An Old Book Selling Web Application is a web-based platform designed to simplify the process of purchasing books. It allows customers to browse, select, and buy books online, offering a convenient and hassle-free experience. The system can be accessed through any web browser on various devices, including laptops, desktops, smartphones, and tablets, making it highly accessible for users at any time and from anywhere.

The Old Book Selling Web Application is built with a user-friendly interface that enables customers to easily search for books by title, author, genre, or ISBN. It displays detailed information for each book, including descriptions, author details, prices, and availability. Users can select the books they wish to purchase, choose formats (print or eBook), and select their preferred delivery method. The system supports various payment methods, including credit/debit cards, mobile wallets, and online banking, allowing customers to complete their purchases with ease.

2.4 Software/Tools

There are several software and tools that can be used to develop and implement a bus booking system. Here are some of the most popular ones:

1. PHP: PHP is a popular server-side scripting language that is widely used to develop web applications. It is an open-source language that is easy to learn and has a large community of developers. It can be used to develop the server-side components of the bus booking system.

2. MySQL: MySQL is an open-source relational database management system that is widely used for web applications. It can be used to store and manage the bus booking system's data, including bus schedules, ticket availability, and customer information.

3. HTML/CSS/JavaScript: HTML, CSS, and JavaScript are the three core technologies used to develop web applications. They are used to develop the user interface of the bus booking system and make it more user-friendly.

4. Bootstrap: Bootstrap is a popular front-end development framework that makes it easy to develop responsive and mobile-first web applications. It can be used to develop the user interface of the bus booking system and make

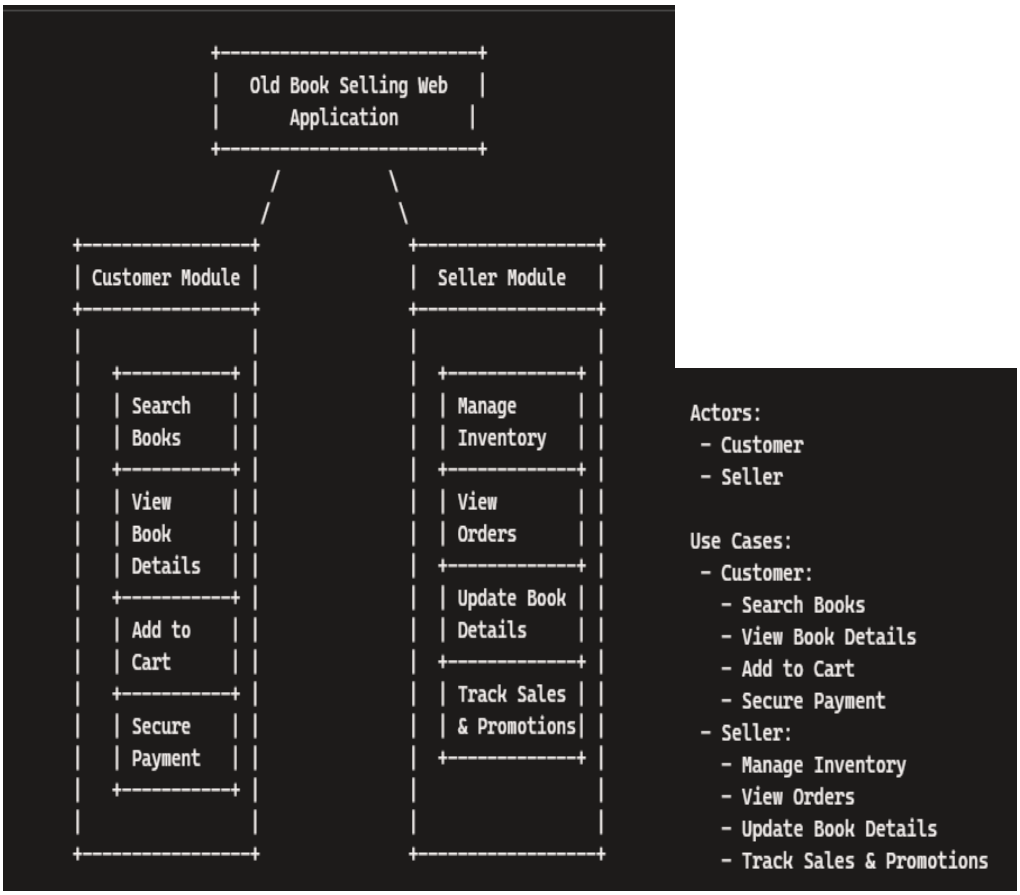
it compatible with different devices and screen sizes.

5. Payment gateways: Payment gateways such as PayPal, Stripe, and Braintree can be integrated into the bus booking system to facilitate secure and convenient payment processing.

2.5 Conceptual Models

Conceptual models are graphical representations of the relationships between different entities in a system. They are used to simplify complex systems and help stakeholders understand how the system works. In the context of a bus booking system, there are different conceptual models that can be used to represent the system's different aspects. Here are some examples:

1. Use case diagram: A use case diagram is a graphical representation of the system's functionalities from the user's perspective. It shows the different actors (e.g., passengers, bus operators, administrators) and the different use cases (e.g., search for bus routes, book a ticket, manage bookings). It helps stakeholders understand how the system will be used and what functionalities it will offer.



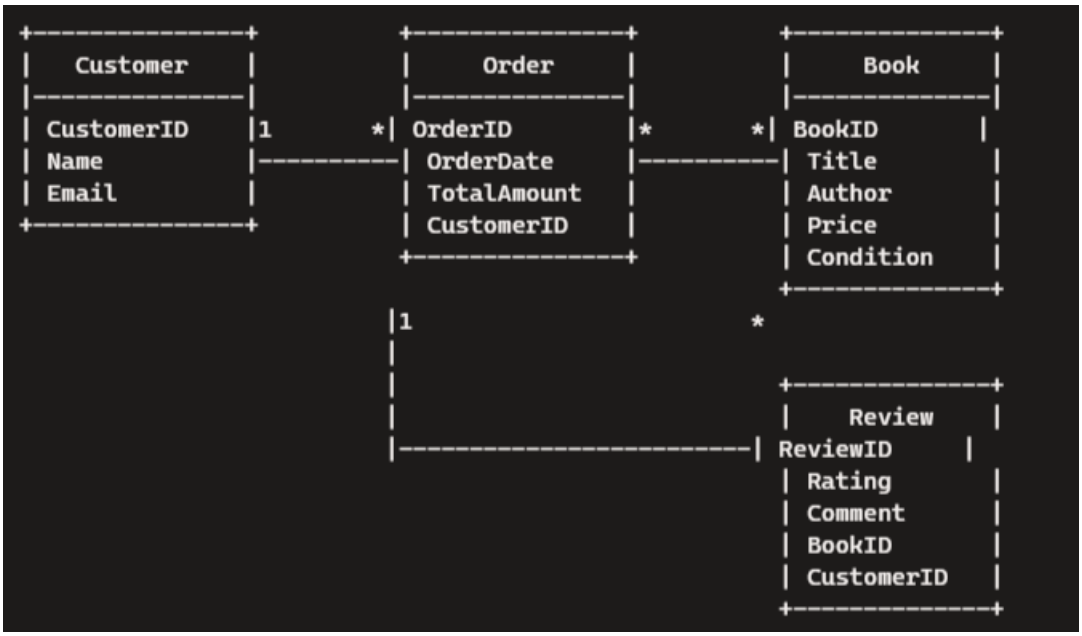
2. **An Entity-Relationship (ER)** diagram is a visual representation of the relationships between entities in a database. It is used in database design to illustrate the structure and relationships of data in an organized way. Here are the key components of an ER diagram:

1. **Entities**: These are objects or concepts that can have data stored about them. Each entity is represented by a rectangle. For example, in a book-selling application, entities might include "Customer," "Book," and "Order."

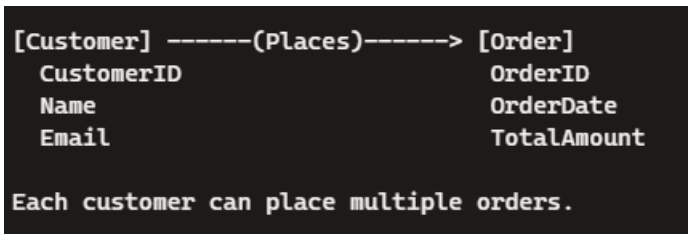
2. **Attributes**: These are the data we want to store about the entity, represented by ovals connected to their respective entities. For example, a "Book" entity might have attributes like Title, Author, and Price.

3. **Relationships**: These show how entities interact with each other and are represented by diamonds. For example, the relationship between "Customer" and "Order" might be "Places."

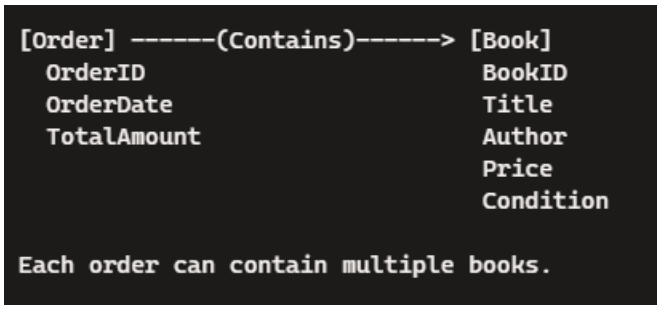
4. **Primary Key**: A unique identifier for each entity, usually underlined within the entity box. For example, "BookID" could be the primary key for the "Book" entity.



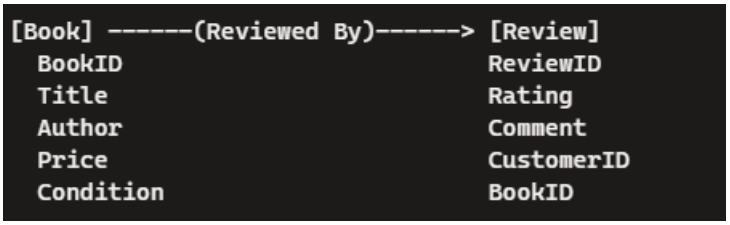
Customer Module



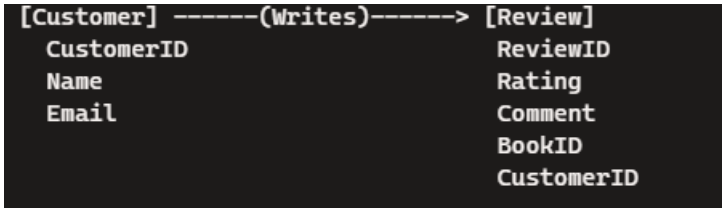
Order Module



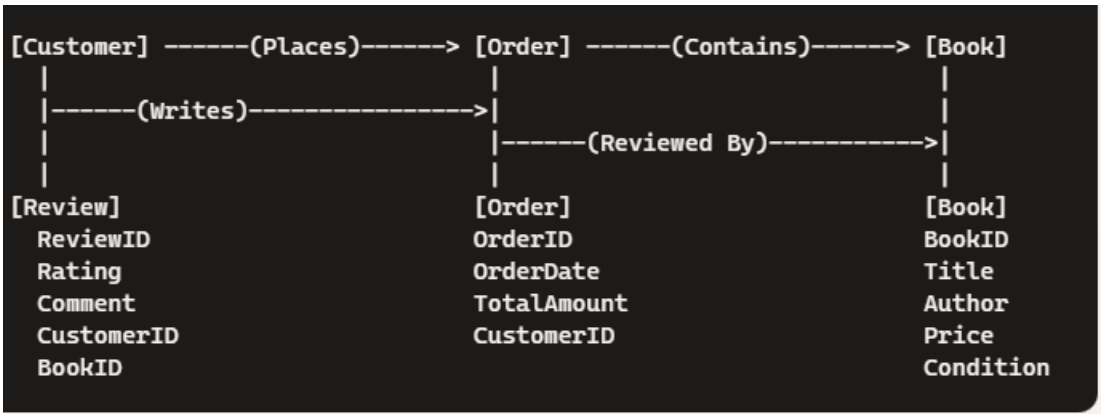
Book Module



Review Module



Combined Relationship Module



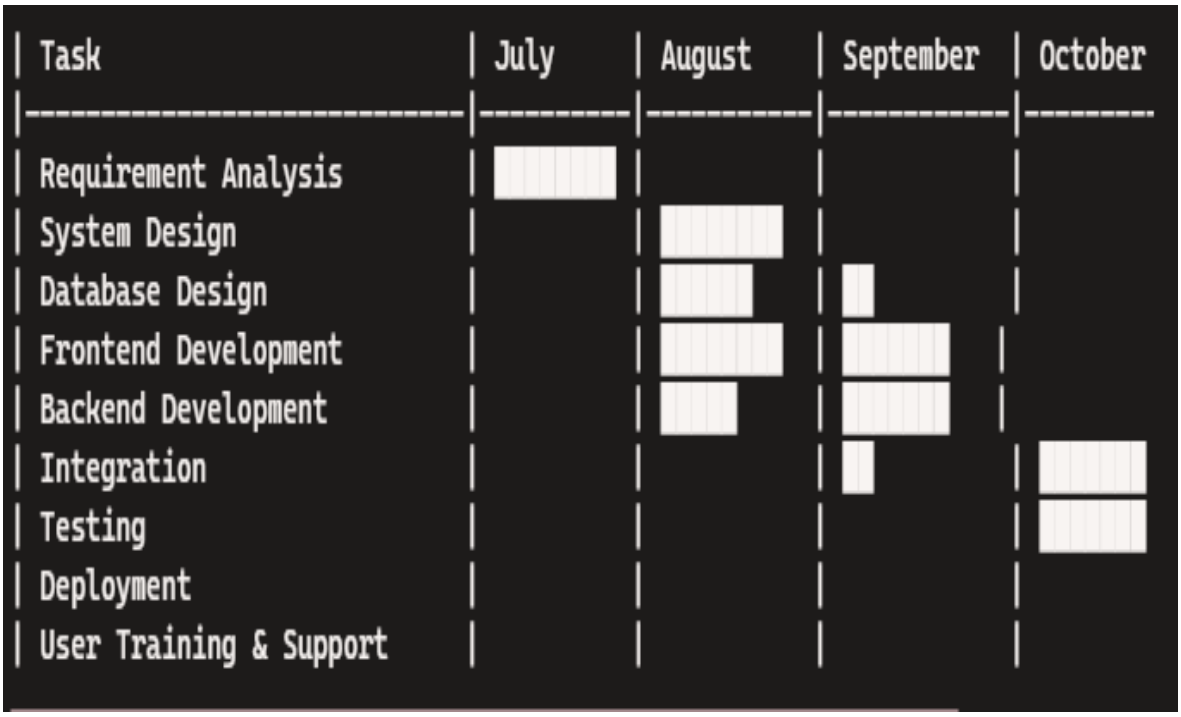
Description

- **Customer to Order**: A customer can place multiple orders.
- **Order to Book**: An order can contain multiple books.
- **Book to Review**: A book can have multiple reviews.
- **Customer to Review**: A customer can write multiple reviews for different books.

3 Gantt- Chart

Description of Tasks

- **Requirement Analysis (July)**: Gathering and analyzing requirements from stakeholders.
- **System Design (August)**: Designing the architecture and components of the system.
- **Database Design (August - September)**: Creating the database schema and structure.
- **Frontend Development (August - September)**: Developing the user interface of the application.
- **Backend Development (August - September)**: Developing the server-side logic and APIs.
- **Integration (September - October)**: Integrating the frontend with the backend.
- **Testing (October - November)**: Performing various tests to ensure functionality and performance.



5. Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) is a graphical representation of the flow of data through a system. It is used to visualize how information moves from input to output within a system and the processes that transform data. DFDs are particularly useful in system analysis and design to understand how data is processed, where it comes from, and where it goes.

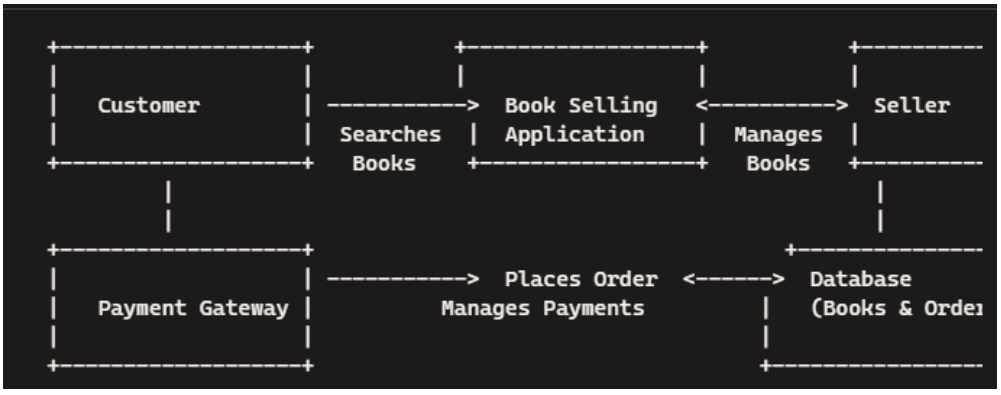
Key Components of a DFD:

1. **External Entities**: Represent sources or destinations of data outside the system. These are usually shown as rectangles. Examples include customers, suppliers, or external databases.
2. **Processes**: Represent activities that transform data. Processes are shown as circles or rounded rectangles. Each process has a name that describes what it does, such as "Process Order" or "Update Inventory."
3. **Data Stores**: Represent places where data is stored within the system. These are shown as open-ended rectangles (like a cylinder). Examples include databases or file systems.
4. **Data Flows**: Represent the movement of data between entities, processes, and data stores. Data flows are shown as arrows indicating the direction of data movement.

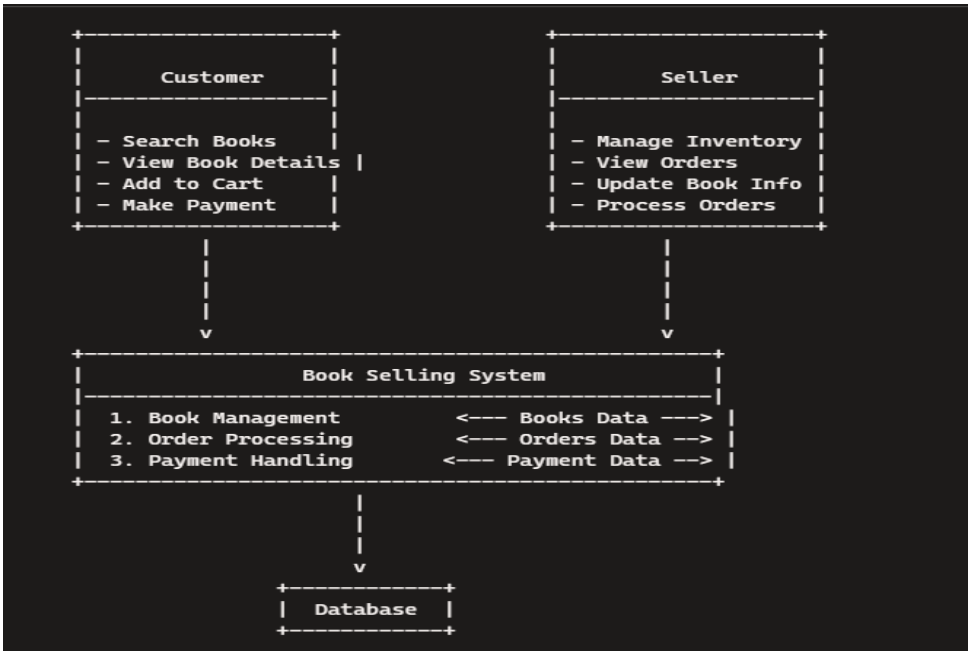
Benefits of Using DFDs:

- **Clarity**: Simplifies complex systems by breaking them down into manageable parts.
- **Communication**: Helps stakeholders understand the system's workings at various levels of detail.
- **Problem Identification**: Facilitates the identification of inefficiencies or bottlenecks in the system.

Zero Level



First Level DFD



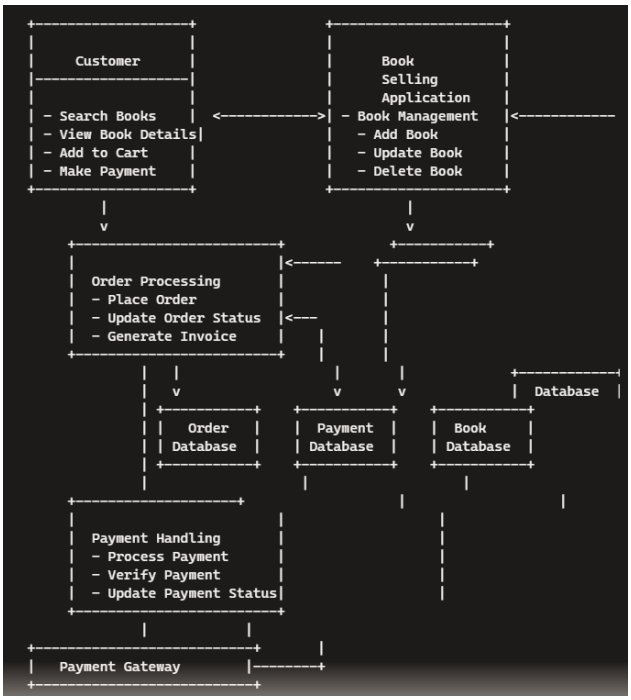
Second Level DFD

Key Components:

- **Customer**: Searches for books, views book details, adds books to the cart, and makes payments.
- **Seller**: Manages inventory, views orders, updates book information, and processes orders.
- **Book Selling Application**: Central system handling book management, order processing, and payment handling.
- **Database**: Stores book information, orders, and payment details.
- **Payment Gateway**: External system for processing payments.

Main Processes:

1. **Book Management**: Involves adding, updating, and deleting book records.
2. **Order Processing**: Manages order placement, status updates, and invoice generation.
3. **Payment Handling**: Processes and verifies payments, updating their status.



CHAPTER-3

SYSTEM DESIGN

3.1: Basic Modules

An Old book selling Web Application typically includes two main modules: the customer module and the operator module. Here's a brief overview of the features and functions of each module:

Customer Module:

The **customer module** is the user-facing part of the Old book selling Web Application , where customers can browse books, select their desired items, and complete the purchasing process. The key features of the customer module include:

1. **Book Search**: Customers can search for books by keywords like title, author, genre, or ISBN. Results show book descriptions, price, availability, and ratings. Filters include price range, publication date, and format (paperback, hardcover, eBook, audiobook).
2. **Book Selection**: Customers can choose the format and add books to their cart. For eBooks, they can preview or sample chapters before buying.

3. **Shopping Cart and Checkout**: Customers review their cart with details such as quantity, format, price, and total cost. They can update items, apply promo codes, and enter shipping or download info before checkout.
4. **Payment Processing**: Customers pay through a secure gateway. Options include credit/debit cards, net banking, and mobile wallets.
5. **Order Confirmation**: After payment, the system sends an order confirmation and e-receipt. For physical books, it includes shipping details and an estimated delivery date.

Operator Module

1. **Inventory and Book Management**: Operators manage the book catalog, including adding, removing, or updating listings. They can set titles, authors, genres, ISBNs, descriptions, prices, stock availability, and formats. Promotional pricing and discounts can also be set.
2. **Order and Booking Management**: Operators view and manage customer orders, track status, process payments, and update order statuses. They handle cancellations, exchanges, refunds, and can modify orders if needed.
3. **Customer Management**: Operators view customer profiles, order history, payment details, and personal info. They assist with returns, refunds, and inquiries. Order confirmations, shipping updates, and promotions can be sent directly to customers.
4. **Reports and Analytics**: The module tracks and analyzes sales, inventory, and customer behavior. Operators generate reports on sales trends, popular genres, revenue, and engagement to optimize inventory and marketing.
5. **Shipping and Fulfillment Management**: Operators manage the shipping process, update statuses, generate shipping labels, and track deliveries. This ensures timely and accurate fulfillment.

3.4 Security Issues

An online book selling system faces several security threats, including unauthorized access, payment fraud, database breaches, denial-of-service (DoS) attacks, and insider threats. Mitigating these risks involves implementing strong authentication mechanisms like multi-factor authentication (MFA), secure payment gateways with encryption, strict access controls, DDoS protection tools, and monitoring user activity. Regular security audits, penetration testing, and compliance with regulations such as PCI-DSS further enhance the system's security and ensure safe transactions.

CHAPTER-4

IMPLEMENTATION AND TESTING

4.1 Implementation Approaches (Installation Procedure)

The implementation of an **Old book selling Web Application** involves several key steps, ranging from software installation and configuration to testing and deployment. Here are the main steps involved in the installation and setup procedure for the system:

1. **Software Installation:**

The first step is to install the necessary software and tools required for developing the online book selling system. This includes selecting and installing the programming language (e.g., PHP, Python, or JavaScript), web server software (e.g., Apache or Nginx), database management system (DBMS) (e.g., MySQL or PostgreSQL), and development tools such as a text editor or integrated development environment (IDE) for writing code.

2. **Database Setup:**

The next step involves setting up the database management system (DBMS) and creating the necessary database tables and relationships. The database will store key information such as book catalogs (titles, authors, prices, descriptions, etc.), customer accounts, order history, payment details, and shipping information. Database tables should be designed to handle these relationships efficiently, and appropriate constraints should be put in place to ensure data integrity.

3. **Web Server Configuration:**

The web server needs to be configured to handle HTTP requests and serve web pages to users. This includes configuring the web server (e.g., Apache or Nginx) to work with the programming language and web framework used in developing the online book selling system. Configuration steps may also include setting up server security, caching mechanisms, and other optimizations to improve system performance.

4. **Application Development:**

Once the development environment is set up, the development team can begin building the online book selling system. This involves writing code for essential features such as book search, product listings, shopping cart functionality, order processing, and payment gateway integration. The system should also include necessary security features such as encryption (for sensitive data like passwords and payment details), input validation, and access control to protect user data and transactions.

5. Testing:

Once the core features of the system have been developed, thorough testing is essential to ensure that the application functions as intended. This includes:

- **Functional Testing:** Verifying that all features (such as book search, checkout, and payment processing) work correctly.
- **Integration Testing:** Ensuring that different components of the system (such as the front-end, back-end, and database) work together seamlessly.
- **Performance Testing:** Evaluating how the system handles high traffic, large numbers of simultaneous users, and database queries to ensure the system performs well under load.
- **Security Testing:** Ensuring that sensitive data is properly encrypted, that there are no vulnerabilities in the system (e.g., SQL injection or cross-site scripting), and that the system is secure against potential attacks.

6. Deployment:

Once testing is complete and the system is stable, the online book selling system can be deployed to a production environment. This involves setting up the domain name, SSL certificates (for secure HTTPS connections), and choosing a hosting provider (e.g., AWS, DigitalOcean, or other cloud platforms). Additional deployment steps may include configuring email servers for customer notifications, integrating analytics tools, and setting up regular backups to protect customer and business data.

Source code

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

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Contact Us | BooksMart</title>
    <link rel="icon" href="bmlogo.png" type="image/png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="contact_us.css">
</head>
<body>


    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-dark fixed-top" style="background-color: #d2d3db;">


        <div class="container-fluid">
            <!-- Logo and Name -->
            <a class="navbar-brand d-flex align-items-center" href="#">
                <img src="bmlogo.png" alt="Logo" class="rounded-circle me-2" width="40" height="40"> <!-- Circular Logo -->
                <strong>BooksMart</strong>
            </a>


            <!-- Toggler Button for Mobile View -->
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>




    <!-- Contact Us Form with Background Image -->
    <div class="container contact-container">
        <h2 class="text-center text-white">Contact Us</h2>
```

```
        <form id="contact-form" class="shadow-lg p-4 bg-light bg-opacity-50 rounded">
            <div class="mb-3">
                <label for="name" class="form-label text-white">Name</label>
                <input type="text" class="form-control" id="name" name="name" placeholder="Your name" required>
            </div>
            <div class="mb-3">
                <label for="email" class="form-label text-white">Email</label>
                <input type="email" class="form-control" id="email" name="email" placeholder="Your email" required>
            </div>
            <div class="mb-3">
                <label for="subject" class="form-label text-white">Subject</label>
                <input type="text" class="form-control" id="subject" name="subject" placeholder="Subject" required>
            </div>
            <div class="mb-3">
                <label for="message" class="form-label text-white">Message</label>
                <textarea class="form-control" id="message" name="message" rows="5" placeholder="Write your message here..." required></textarea>
            </div>
            <button type="submit" class="btn btn-primary w-100">Send Message</button>
            <div id="status" class="text-center mt-3 text-white"></div>
        </form>
    </div>

    <!-- Bootstrap and JavaScript dependencies -->
    <script src="https://cdn.jsdelivr.net/npm/@emailjs/browser@3/dist/email.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"></script>
    <script src="contact_us.js"></script>
</body>
</html>
```

Home.php

```
<?php
session_start();
if (!isset($_SESSION['username'])) {
    header("Location: login.html"); // Redirect to login page if not logged in
    exit;
}
$username = $_SESSION['username'];
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BooksMart-Homepage</title>
    <link rel="icon" href="bmlogo.png" type="image/png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="homepage.css">
    <style>
        /* Logo */
        .logo {
            width: 40px;
            height: 40px;
            border-radius: 50%;
        }

        /* Company name */
        .company-name {
            font-weight: bold;
```

```
    font-size: 1.5rem;
}

/* Centered Navbar styling */
.navbar-center {
    flex-grow: 1;
    display: flex;
    justify-content: center;
    align-items: center;
}

.navbar-nav .nav-link {
    padding-left: 20px;
    padding-right: 20px;
    transition: color 0.3s ease;
}

.navbar-nav .nav-link:hover {
    color: #00bfff; /* Tint color on hover */
}

/* Increase space between 'Sell' and the search bar */
.search-bar {
    width: 250px;
    margin-left: 40px; /* Increased margin to move search bar further away from 'Sell' */
}

/* Carousel styling */
.carousel-item img {
    width: 100%;
```

```
            height: 100vh; /* Full viewport height */
            object-fit: cover;
        }

        .carousel-caption {
            bottom: 20%;
            text-align: center;
        }

        .carousel-caption h1 {
            font-size: 3rem;
            font-style: italic;
            color: #fff;
            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.6);
        }
    </style>
</head>
<body>

    <!-- Navbar -->
    <nav class="navbar navbar-expand-lg navbar-light bg-light fixed-top">
        <div class="container-fluid">
            <!-- Logo and Company Name -->
            <a class="navbar-brand d-flex align-items-center" href="#">
                <img src="bmlogo.png" alt="Logo" class="logo">
                <span class="company-name ms-2">BooksMart</span>
            </a>

            <!-- Toggler for small screens -->
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
```

```
        <span class="navbar-toggler-icon"></span>
      </button>

      <!-- Centered menu and search bar -->
      <div class="collapse navbar-collapse" id="navbarNav">
        <div class="navbar-center">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link" href="home.php">Home</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="contact_us.html">Contact Us</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="sellings.php">Sell</a>
            </li>
          </ul>

          <!-- Search bar with more space between 'Sell' -->
          <form class="d-flex" action="searching.php" method="GET" id="search-form">
            <input class="form-control search-bar" type="search" placeholder="Search" name="query" aria-label="Search">
          </form>
        </div>

        <!-- Sign Out button on the far right -->
        <form class="d-flex ms-auto" action="logout.php" method="post">
          <button class="btn btn-outline-danger" type="submit">Sign Out</button>
        </form>
      </div>
    </div>
```

```
</nav>

<!-- Carousel -->
<div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="carousel" data-bs-pause="false">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img src="img1.jpg" class="d-block w-100" alt="Slide 1">
      <div class="carousel-caption d-none d-md-block">
        <h1>Find your favorite classics and rare finds</h1>
      </div>
    </div>
    <div class="carousel-item">
      <img src="img2.jpg" class="d-block w-100" alt="Slide 2">
      <div class="carousel-caption d-none d-md-block">
        <h1>Find your favorite classics and rare finds</h1>
      </div>
    </div>
    <div class="carousel-item">
      <img src="img3.jpg" class="d-block w-100" alt="Slide 3">
      <div class="carousel-caption d-none d-md-block">
        <h1>Find your favorite classics and rare finds</h1>
      </div>
    </div>
    <div class="carousel-item">
      <img src="img4.jpg" class="d-block w-100" alt="Slide 4">
      <div class="carousel-caption d-none d-md-block">
        <h1>Find your favorite classics and rare finds</h1>
      </div>
    </div>
    <div class="carousel-item">
```

```
            <img src="img6.jpg" class="d-block w-100" alt="Slide 5">
            <div class="carousel-caption d-none d-md-block">
              <h1>Find your favorite classics and rare finds</h1>
            </div>
          </div>
        </div>
        <button class="carousel-control-prev" type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide="prev">
          <span class="carousel-control-prev-icon" aria-hidden="true"></span>
          <span class="visually-hidden">Previous</span>
        </button>
        <button class="carousel-control-next" type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide="next">
          <span class="carousel-control-next-icon" aria-hidden="true"></span>
          <span class="visually-hidden">Next</span>
        </button>
      </div>

      <div class="container my-5">
        <div class="row">
          <div class="col-md-4">
            <div class="card">
              <img src="x.jpg" class="card-img-top" alt="Book 1">
              <div class="card-body">
                <h5 class="card-title">Students</h5>
                <p class="card-text">Unlock academic success with affordable textbooks just a click away!</p>
                <a href="student_db.php" class="btn btn-primary">View Details</a>
              </div>
            </div>
          </div>
          <div class="col-md-4">
```

```
            <div class="card">
                <img src="y.jpg" class="card-img-top" alt="Book 2">
                <div class="card-body">
                    <h5 class="card-title">Professionals</h5>
                    <p class="card-text">Upgrade your expertise with rare and valuable professional reads!</p>
                    <a href="professional_db.php" class="btn btn-primary">View Details</a>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <img src="z.jpg" class="card-img-top" alt="Book 3">
                <div class="card-body">
                    <h5 class="card-title">Casuals</h5>
                    <p class="card-text">Discover hidden gems and timeless classics at unbeatable prices!</p>
                    <a href="casuals_db.php" class="btn btn-primary">View Details</a>
                </div>
            </div>
        </div>
    </div>
</div>

<footer class="footer mt-auto py-3 bg-dark text-white text-center">
    <div class="container">
        <span>&copy; 2024 BooksMart. All rights reserved.</span>
    </div>
</footer>
    $(document).ready(function() {
        // Initialize typeahead on the search input
        $('.search-bar').typeahead({
```

```
            source: function(query, process) {
                return $.ajax({
                    url: 'search_books.php',
                    type: 'GET',
                    data: { query: query },
                    dataType: 'json',
                    success: function(data) {
                        return process(data);
                    }
                });
            },
            afterSelect: function(item) {
                $('#search-form').submit();
            }
        });
    });
  </script>
</body>
</html>
```

Payment.php

```
<?php
include 'db_connection.php';

// Retrieve id from URL
$id = isset($_GET['id']) ? intval($_GET['id']) : 0;

// Fetch the book details from the database
```

```
$sql = "SELECT book_name, price, rent_or_sell, qr_code_image, no_of_days, id, user_type FROM sellings WHERE id = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $id);
$stmt->execute();
$stmt->bind_result($bookName, $price, $rentOrSell, $qrCodeBlob, $noOfDays, $bookId, $userType);
$stmt->fetch();
$stmt->close();
$conn->close();



// Convert the QR code binary data to base64 if it's not null
$qrCodeImage = $qrCodeBlob ? base64_encode($qrCodeBlob) : null;
?>



<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Payment</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
        body {
            background-image: url('payment.jpg');
            background-size: cover;
            background-repeat: no-repeat;
            background-attachment: fixed;
        }
        .container {
```

```
    background-color: rgba(255, 255, 255, 0.9);
    padding: 30px;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
    margin-top: 50px;
    max-width: 600px;
}
.btn-primary {
    width: 100%;
    font-size: 18px;
    font-weight: bold;
}
/* Styles for enlarged QR code */
.qr-code-modal {
    display: none;
    position: fixed;
    z-index: 1000;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    overflow: auto;
    background-color: rgba(0, 0, 0, 0.8);
    padding-top: 60px;
}
.qr-code-modal img {
    margin: auto;
    display: block;
    max-width: 90%;
    max-height: 90%;
```

```
        }
        .modal-content {
            background-color: #fefefe;
            margin: 5% auto;
            padding: 20px;
            border: 1px solid #888;
            width: 80%;
        }
    </style>
</head>
<body>
<nav class="navbar navbar-expand-lg navbar-light bg-light py-3 shadow-sm w-100">
    <div class="container-fluid">
        <a class="navbar-brand d-flex align-items-center" href="home.php">  <img src="bmlogo.png" alt="Logo" class="rounded-circle" style="width: 40px; height: 40px; margin-right: 10px;">
            <span class="fw-bold" style="font-size: 24px;">BooksMart</span>
        </a>


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


        <div class="collapse navbar-collapse justify-content-end">
            <a class="btn btn-outline-danger px-4" href="logout.php">Sign Out</a>
        </div>
    </div>
</nav>
         <button type="submit" class="btn btn-primary">Submit Payment</button>
    </form>
</div>
```

```
<!-- QR Code Modal -->
<div id="qrCodeModal" class="qr-code-modal" onclick="closeQrCodeModal()">
    <div class="modal-content">
        <span onclick="closeQrCodeModal()" style="cursor: pointer; float: right;">&times;</span>
        <?php if ($qrCodeImage): ?>
            <img src="<?php echo $qrCodeImage; ?>" alt="QR Code" style="max-width: 50%; max-height: 50%;"/>
        <?php endif; ?>
    </div>
</div>

<!-- Success Modal -->
<div class="modal fade" id="successModal" tabindex="-1" aria-labelledby="successModalLabel" aria-hidden="true">
    <div class="modal-dialog">

    function openQrCodeModal() {
        document.getElementById('qrCodeModal').style.display = "block";
    }

    function closeQrCodeModal() {
        document.getElementById('qrCodeModal').style.display = "none";
    }

    function resetForm() {
        document.getElementById('paymentForm').reset(); // Reset the form fields
        toggleQrCode(false); // Hide the QR code image if it was shown
    }

    // Check if the success parameter is present in the URL
```

```
    const urlParams = new URLSearchParams(window.location.search);
    if (urlParams.has('success')) {
        var successModal = new bootstrap.Modal(document.getElementById('successModal'));
        successModal.show();
    }
</script>
</body>
</html>
```

payment.php

```
<?php
session_start();
if (!isset($_SESSION['username'])) {
    header("Location: login.html"); // Redirect to login page if not logged in
    exit;
}
?>
<?php
// Include the database connection file
include 'db_connection.php';

// Get the id and user_type from the URL
$id = isset($_GET['id']) ? $_GET['id'] : '';
$user_type = isset($_GET['user_type']) ? $_GET['user_type'] : '';

if ($id != '' && $user_type != '') {
    // Fetch all details for the selected book based on both id and user_type
    $sql = "SELECT * FROM sellings WHERE user_type = ? AND id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("ss", $user_type, $id);
    $stmt->execute();
```

```
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $book = $result->fetch_assoc();
    } else {
        echo "No book found with that id and user type.";
        exit;
    }
} else {
    echo "Invalid id or user type.";
    exit;
}

$conn->close();
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Book Details</title>
    <link rel="icon" href="bmlogo.png" type="image/png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <style>
        body {
            background-image: url('https://img.freepik.com/premium-photo/stack-books-stationery-background-school-board_147376-4875.jpg');
            background-size: cover;
            background-repeat: no-repeat;
            background-attachment: fixed;
        }
```

```
        .navbar {
            background-color: #007bff;


<?php
session_start();
if (!isset($_SESSION['username'])) {
    header("Location: login.html"); // Redirect to login page if not logged in
    exit;
}
$username = $_SESSION['username'];
?>
styling */
        .navbar-center {
            flex-grow: 1;
            display: flex;
            justify-content: center;
            align-items: center;
        }

        .navbar-nav .nav-link {
            padding-left: 20px;
            padding-right: 20px;
            transition: color 0.3s ease;

            <!-- Toggler for small screens -->
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>

            <!-- Centered menu and search bar -->
            <div class="collapse navbar-collapse" id="navbarNav">
                <div class="navbar-center">
                    <ul class="navbar-nav">
                        <li class="nav-item">
                            <a class="nav-link" href="home.php">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" href="contact_us.html">Contact Us</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" href="sellings.php">Sell</a>
                        </li>
                    </ul>

                    <!-- Search bar with more space between 'Sell' -->
```

```
                <form class="d-flex" action="searching.php" method="GET" id="search-form">
                    <input class="form-control search-bar" type="search" placeholder="Search" name="query" aria-
label="Search">
                </form>
            </div>

            <!-- Sign Out button on the far right -->
            <form class="d-flex ms-auto" action="logout.php" method="post">
                <button class="btn btn-outline-danger" type="submit">Sign Out</button>
            </form>
        </div>
    </div>
</nav>

<!-- Carousel -->
<div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="carousel" data-bs-pause="false">
    <div class="carousel-inner">
        <div class="carousel-item active">
            <img src="img1.jpg" class="d-block w-100" alt="Slide 1">
            <div class="carousel-caption d-none d-md-block">
                <h1>Find your favorite classics and rare finds</h1>
            </div>
        </div>
        <div class="carousel-item">
            <img src="img2.jpg" class="d-block w-100" alt="Slide 2">
            <div class="carousel-caption d-none d-md-block">
                <h1>Find your favorite classics and rare finds</h1>
            </div>
                <a href="student_db.php" class="btn btn-primary">View Details</a>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <img src="y.jpg" class="card-img-top" alt="Book 2">
                <div class="card-body">
                    <h5 class="card-title">Professionals</h5>
                    <p class="card-text">Upgrade your expertise with rare and valuable professional reads!</p>
                    <a href="professional_db.php" class="btn btn-primary">View Details</a>

<!-- Bootstrap and JS Scripts -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script                    src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-3-typeahead/4.0.2/bootstrap3-
typeahead.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
<script>
    $(document).ready(function() {
```

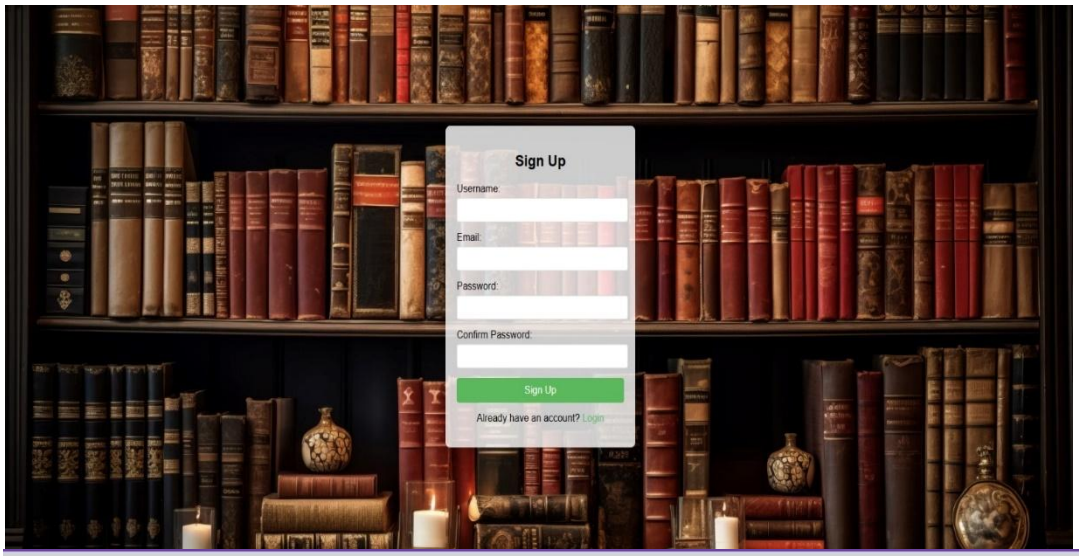
```
            // Initialize typeahead on the search input
            $('.search-bar').typeahead({
                source: function(query, process) {
                    return $.ajax({
                        url: 'search_books.php',
                        type: 'GET',
                        data: { query: query },
                        dataType: 'json',
                        success: function(data) {
                            return process(data);
                        }
                    });
                },
                afterSelect: function(item) {
                    $('#search-form').submit();
                }
            });
        });
    </script>
</body>
</html>

<?php
include 'db_connection.php';

// Retrieve id from URL
$id = isset($_GET['id']) ? intval($_GET['id']) : 0;

// Fetch the book details from the database
$sql = "SELECT book_name, price, rent_or_sell, qr_code_image, no_of_days, id, user_type FROM sellings WHERE id = ?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $id);
$stmt->execute();
$stmt->bind_result($bookName, $price, $rentOrSell, $qrCodeBlob, $noOfDays, $bookId, $userType);
$stmt->fetch();
$stmt->close();
$conn->close();


// Convert the QR code binary data to base64 if it's not null
$qrCodeImage = $qrCodeBlob ? base64_encode($qrCodeBlob) : null;
?>


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

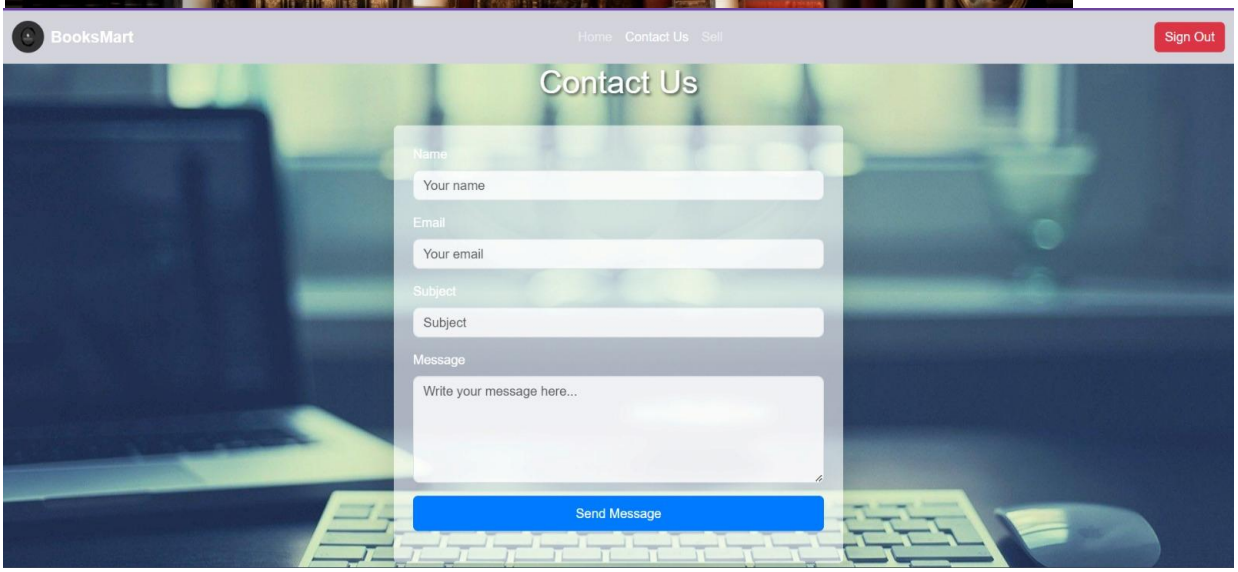
```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Payment</title>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
<style>
    body {
        background-image: url('payment.jpg');
        background-size: cover;
        background-repeat: no-repeat;
        background-attachment: fixed;
    }
    .container {
        background-color: rgba(255, 255, 255, 0.9);
        padding: 30px;
        border-radius: 8px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
        margin-top: 50px;
        max-width: 600px;
    }
    .btn-primary {
        width: 100%;
        font-size: 18px;
        font-weight: bold;
    }
    /* Styles for enlarged QR code */
    .qr-code-modal {
        display: none;
        position: fixed;
        z-index: 1000;
        left: 0;
        top: 0;
        width: 100%;
        height: 100%;
        overflow: auto;
        background-color: rgba(0, 0, 0, 0.8);
        padding-top: 60px;
    }
    .qr-code-modal img {
        margin: auto;
        display: block;
        max-width: 90%;
        max-height: 90%;
    }
    .modal-content {
        background-color: #fefefe;
        margin: 5% auto;
        padding: 20px;
        border: 1px solid #888;
```

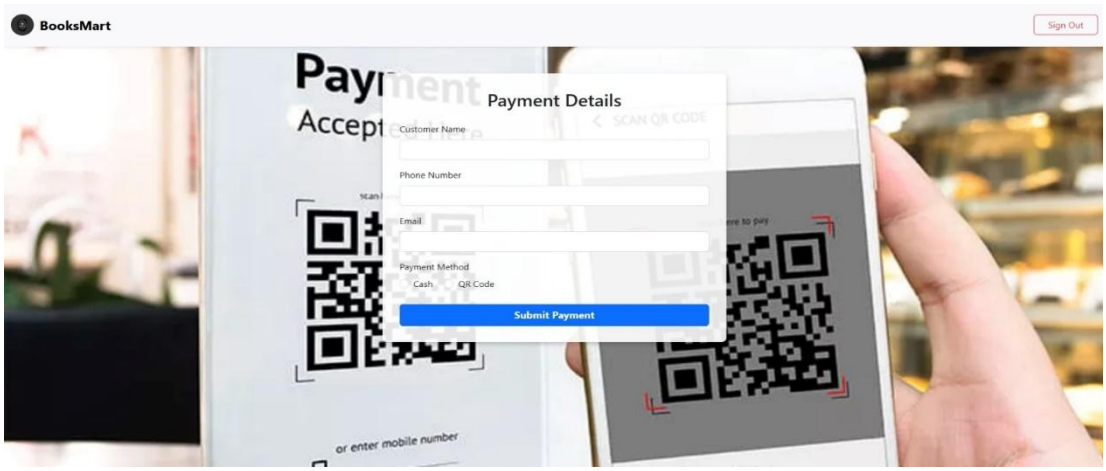
```
            width: 80%;
        }
        <div class="mb-3">
            <label class="form-label">Payment Method</label><br>
            <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="payment_method" id="cash" value="Cash" required onclick="toggleQrCode(false)">
                <label class="form-check-label" for="cash">Cash</label>
            </div>
            <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="payment_method" id="qr_code" value="QR Code" required onclick="toggleQrCode(true)">
                <label class="form-check-label" for="qr_code">QR Code</label>
            </div>
        </div>
        <div class="text-center mt-4">
            <?php if ($qrCodeImage): ?>
                <img src="<?php echo $qrCodeImage; ?>" alt="QR Code" style="width: 150px; height: 150px; cursor: pointer; display: none;" id="qrCodeImage" onclick="openQrCodeModal()"/>
            <?php else: ?>
                <p>No QR code available for this item.</p>
            <?php endif; ?>
        </div>
        <button type="submit" class="btn btn-primary">Submit Payment</button>
    </form>
</div>

<!-- QR Code Modal -->
<div id="qrCodeModal" class="qr-code-modal" onclick="closeQrCodeModal()">
    <div class="modal-content">
        <span onclick="closeQrCodeModal()" style="cursor: pointer; float: right;">&times;</span>
        <?php if ($qrCodeImage): ?>
            <img src="<?php echo $qrCodeImage; ?>" alt="QR Code" style="max-width: 50%; max-height: 50%;"/>
        <?php endif; ?>
    </div>
</div>

<!-- Success Modal -->
<div class="modal fade" id="successModal" tabindex="-1" aria-labelledby="successModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="successModalLabel">Success</h5>
                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
            </div>
            <div class="modal-body">
```

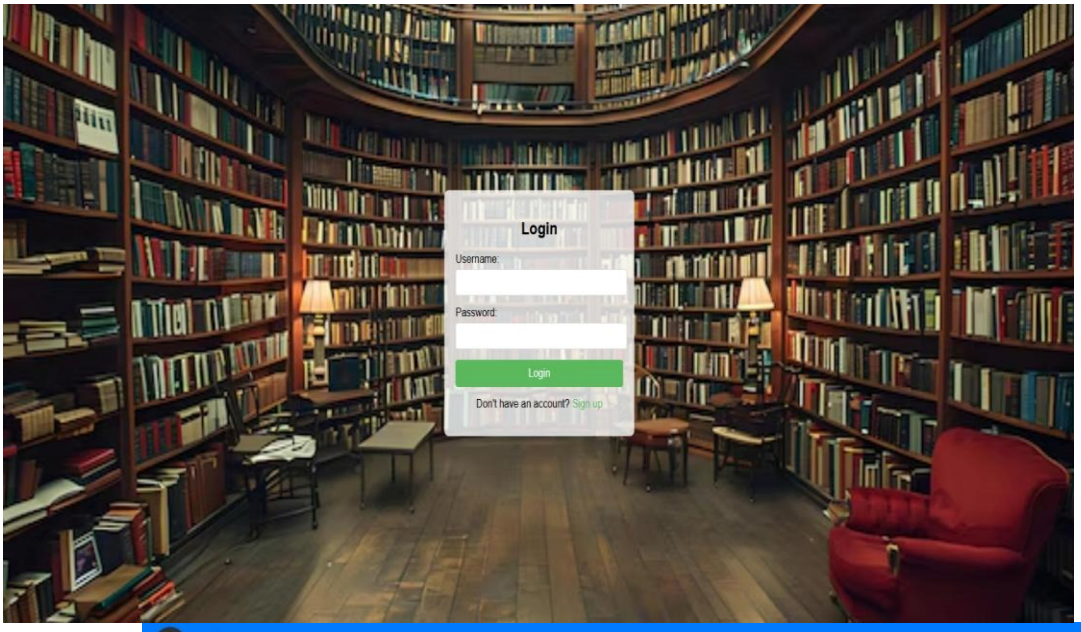
```
            <p>Payment details stored successfully.</p>
          </div>
          <div class="modal-footer">
            <!-- Pass the book id when redirecting to proceed.php -->
            <button type="button" class="btn btn-secondary" data-bs-dismiss="modal" onclick="resetForm();
setTimeout(function(){ window.location.href='proceed.php?id=<?php echo $id; ?>'; }, 300);">Close</button>
          </div>
        </div>
      </div>
    </div>
  </div>
        </div>
      </div>
    </div>
  </div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
<script>
    function toggleQrCode(show) {
        const qrCodeImage = document.getElementById('qrCodeImage');
        qrCodeImage.style.display = show ? 'block' : 'none';
    }

    function openQrCodeModal() {
        document.getElementById('qrCodeModal').style.display = "block";
    }

    function closeQrCodeModal() {
        document.getElementById('qrCodeModal').style.display = "none";
    }

    function resetForm() {
        document.getElementById('paymentForm').reset(); // Reset the form fields
        toggleQrCode(false); // Hide the QR code image if it was shown
    }

    // Check if the success parameter is present in the URL
    const urlParams = new URLSearchParams(window.location.search);
    if (urlParams.has('success')) {
        var successModal = new bootstrap.Modal(document.getElementById('successModal'));
        successModal.show();
    }
</script>
</body>
</html>
```

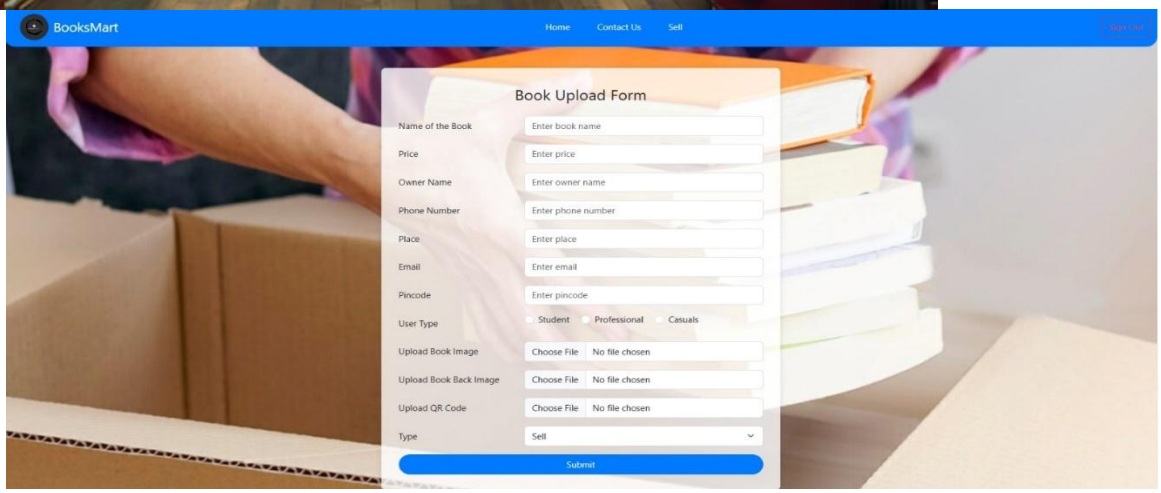
screenshots

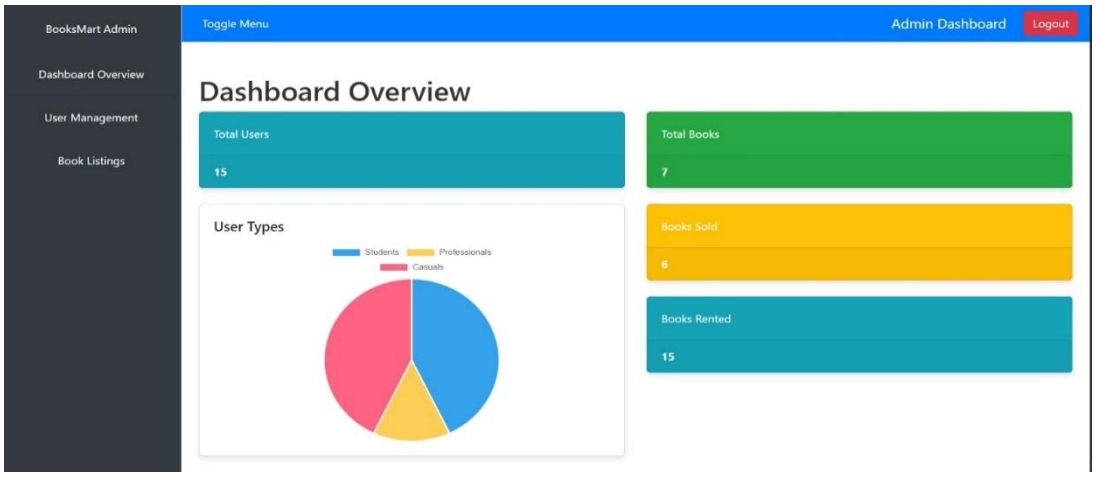







Login
Username:
Password:
Login
Don't have an account? Sign up


BooksMart
Home
Contact Us
Sell
Book Upload Form
Name of the Book
Enter book name
Price
Enter price
Owner Name
Enter owner name
Phone Number
Enter phone number
Place
Enter place
Email
Enter email
Pincode
Enter pincode
User Type
Student
Professional
Casuals
Upload Book Image
Upload Book Back Image
Upload QR Code
Choose File
No file chosen
Type
Sell
Submit


BooksMart Admin
Dashboard Overview
User Management
Book Listings
Toggle Menu
Admin Dashboard
Logout
Dashboard Overview
Total Users
15
Total Books
7
User Types
Students
Professionals
Casuals
Books Sold
6
Books Rented
15

```
desc login;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

Field	Type	Null	Key	Default	Extra
username	varchar(50)	NO		NULL	
email	varchar(50)	NO		NULL	
password	varchar(50)	NO		NULL	

1 of 3,428

Thank You for Your Purchase at BooksMart Inbox x

BooksMart <booksmartbm123@gmail.com> 10:12 PM (0 minutes ago)
to me

Dear Shan Shibu Thomas,

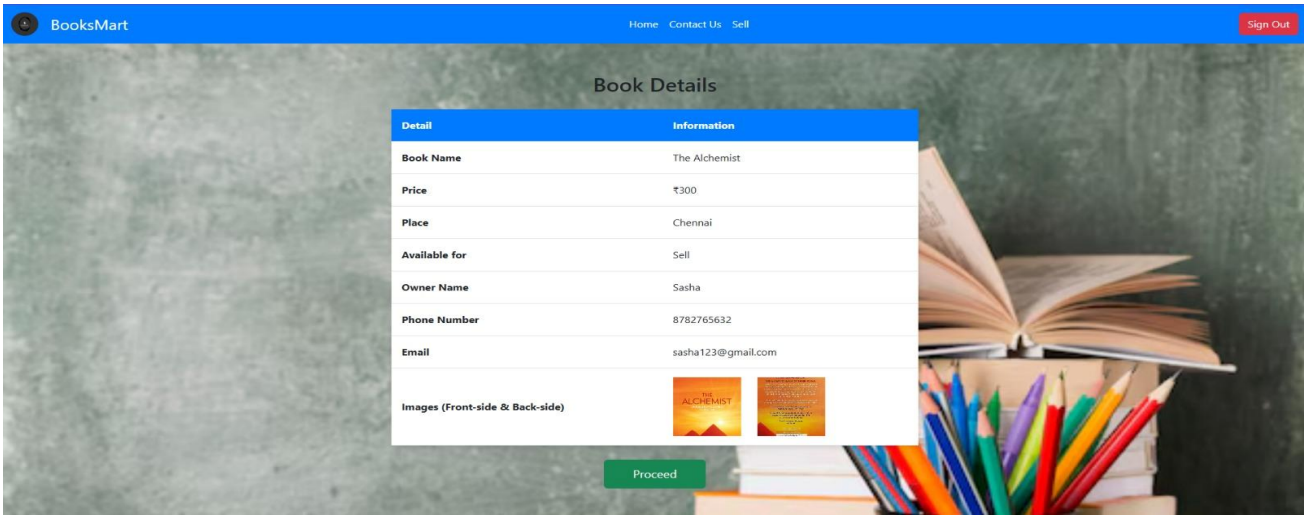
Thank you for purchasing the book 'The Alchemist' for ₹300.
Your payment method: QR Code.

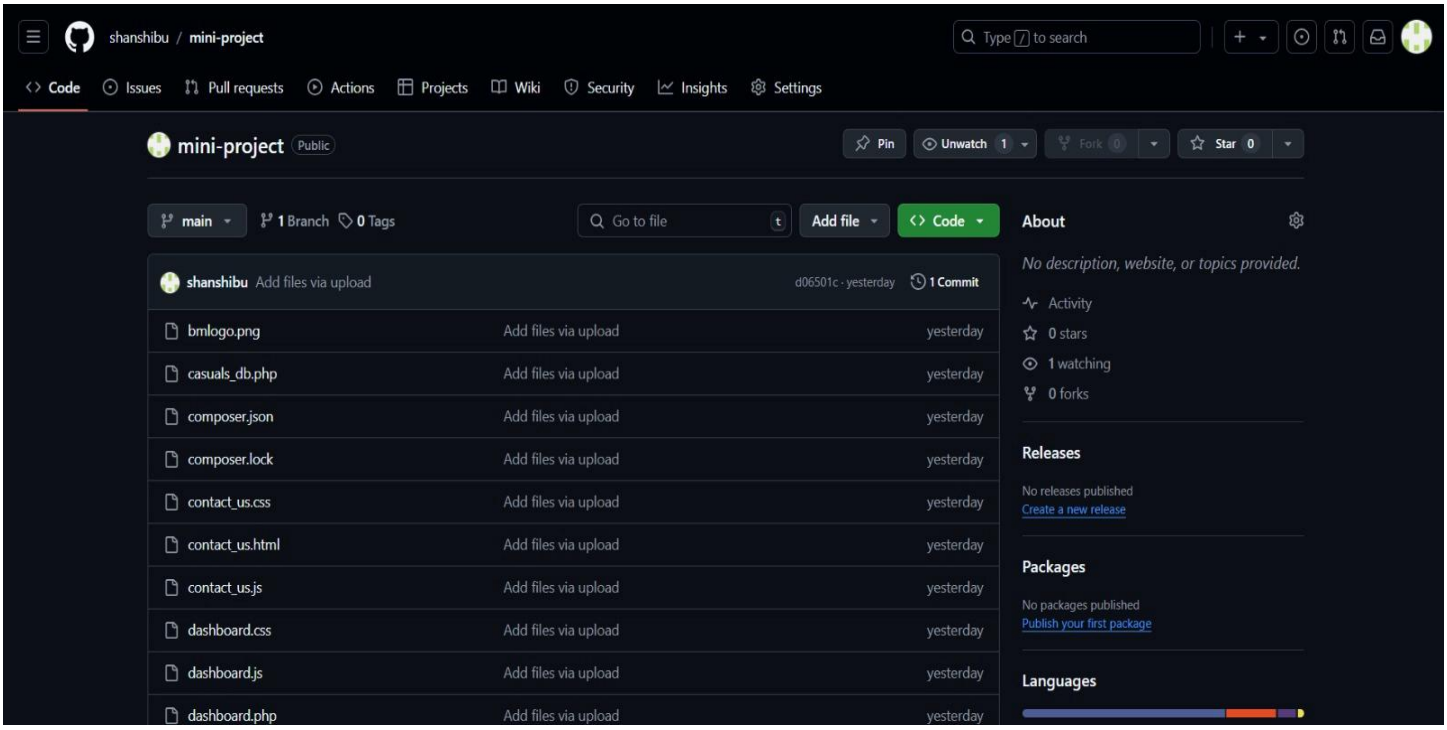
We appreciate your business and hope to see you again soon!

Best regards,
BooksMart Team

Reply Forward

DATABASE






```
desc payment;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
customer_name	varchar(100)	NO		NULL	
phone_number	varchar(15)	NO		NULL	
email	varchar(100)	NO		NULL	
transaction_type	varchar(20)	NO		NULL	
created_at	timestamp	NO		current_timestamp()	

```
desc sellings;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
book_name	varchar(255)	NO		NULL	
price	int(11)	NO		NULL	
owner_name	varchar(255)	NO		NULL	
phone_number	varchar(15)	NO		NULL	
place	varchar(255)	NO		NULL	
email	varchar(255)	NO		NULL	
pincode	varchar(10)	NO		NULL	
user_type	varchar(50)	NO		NULL	
book_image	longblob	YES		NULL	
book_back_image	longblob	YES		NULL	
qr_code_image	longblob	YES		NULL	
rent_or_sell	varchar(50)	NO		NULL	
no_of_days	int(11)	YES		NULL	

4.2 Testing approach

1. **Requirements Review**:
   - Understand requirements and clarify gaps.
2. **Functional Testing**:
   - **User Registration and Login**: Test registration, login, and account management.
   - **Book Search and Browsing**: Verify search functionality and accurate book listings.
   - **Book Selection and Checkout**: Test adding books to the cart, checkout process, and applying promo codes.
3. **Usability Testing:**
   - **UI/UX Evaluation**: Assess interface intuitiveness and navigation.
   - **Mobile Responsiveness**: Test on mobile devices.
   - **Error Messages**: Ensure clear and informative error messages.
4. **Performance Testing**:
   - **Load Testing**: Simulate high user loads.
   - **Scalability**: Evaluate under high traffic.
   - **Response Time**: Measure critical process times.
5. **Security Testing**:
   - **Data Security**: Verify secure data handling.
   - **Authentication and Authorization**: Test for unauthorized access.
   - **Vulnerability Assessment**: Check for common security issues.

6. **Compatibility Testing**:
   - **Browser Compatibility**: Test across various browsers.
   - **Operating System Compatibility**: Verify on different OS.
   - **Accessibility Compliance**: Ensure usability for individuals with disabilities.
7. **Integration Testing**:
   - **Third-Party Integrations**: Test with external systems.
   - **Data Consistency**: Verify accurate data synchronization.
8. **Localization Testing**:
   - **Language Translations**: Verify content accuracy.
   - **Currency Conversion**: Ensure prices adjust correctly.
   - **Date and Time Formats**: Test regional formats.
9. **Cross-Browser Testing**:
   - Ensure consistent functionality and appearance across browsers.
10. **User Acceptance Testing (UAT)**:
    - Conduct with real users for usability feedback.
11. **Regression Testing**:
    - Ensure changes do not impact existing functionality.
12. **Performance Monitoring**:

Use tools to monitor and resolve performance issues.

4.1 Unit testing

Unit testing is a crucial aspect of the software development process, particularly for a private bus booking website. It involves verifying that different components work correctly. Key areas for unit testing include user registration and login, ensuring that user data is properly stored and that users can log in with valid credentials while being denied access with invalid ones. Another focus is on bus listings, ensuring that correct data is fetched from the database and accurately displayed, with filtering and sorting options working as expected. The booking process should be tested to verify the selection of buses, seat allocation, passenger details entry, and payment processes, ensuring that bookings are accurately stored in the database. Reservation management must be tested to ensure users can view and manage their bookings, with the functionality to cancel or modify reservations updating the database accordingly. Payment integration tests ensure smooth and secure transactions, accurately recording successful payments and handling failed transactions appropriately. Error handling tests check the website's response to various error scenarios, ensuring that appropriate error messages are displayed and the system handles unexpected situations gracefully. Finally, security and access control tests ensure unauthorized users cannot perform privileged actions and check for vulnerabilities like SQL injection or cross-site scripting (XSS) attacks. This comprehensive approach to unit testing helps ensure a robust and reliable private bus booking website.

4.2 Integrated testing

Integrated testing for an online book selling system ensures that all components and functionalities work seamlessly together. This involves several key areas. First, user registration and authentication must be tested to confirm that new users can register with valid inputs, create accounts, log in, log out, reset passwords, recover accounts, and use multi-factor authentication if available. Next, book listings and search functionality should be verified to ensure books are correctly categorized and searchable by various criteria such as title, author, genre, or publication date. Features like filtering, sorting, and pagination must also be tested to ensure they return relevant results.

CHAPTER- 5

CONCLUSION

5.1 CONCLUSION

In conclusion, the private bus booking website provides a convenient and efficient platform for individuals and groups to easily book private bus transportation. Through its user-friendly interface, extensive bus options, and

reliable booking system, the website offers a seamless experience for customers seeking private transportation solutions. The website's key features, such as real-time availability updates, secure payment options, and customer support, ensure a smooth and hassle-free booking process. With its commitment to customer satisfaction and a wide network of trusted bus operators, the private bus booking website is a reliable and trustworthy platform for all private bus travel needs. Whether it's for corporate events, group outings, or personal trips, the website simplifies the process of reserving private buses and guarantees a comfortable and convenient journey for all passengers.

5.2 Limitations of the System

The limitations and challenges faced by an online book selling system include limited book selection due to partnerships with only a few publishers or authors, resulting in a restricted variety for customers. Additionally, like bus booking websites, online bookstores struggle with availability and real-time stock updates. The system may also be limited to offering books from a small number of sellers or publishers, which restricts variety in pricing, formats (hardcover, paperback, eBook, audiobook), and exclusive content. Moreover, some online bookstores lack responsive or robust customer support, making it difficult to resolve issues such as cancellations, refunds, order tracking, and complaints. To address these challenges, the platform should expand partnerships to increase book variety, implement real-time inventory management to solve stock issues, and establish a comprehensive customer support system with multiple contact channels (chatbots, live chat, email support), along with a self-service FAQ section and ticketing system to enhance the support process and improve the overall customer experience.

5.3 Future Scope of the Project

The future scope of an online book selling system is vast, with numerous opportunities for growth and innovation. As the demand for convenient and personalized online shopping continues to rise, the platform can expand its services beyond book sales to include eBook subscriptions, audiobook rentals, book-related merchandise, and event ticketing for author signings, book launches, or literary festivals, thus becoming a comprehensive one-stop shop for all literary needs. Developing a mobile application would significantly enhance user experience by making the platform more accessible and user-friendly, allowing users to browse books, make purchases, track shipments, and receive notifications about new releases, sales, and promotions directly on their smartphones, thereby improving customer engagement and driving repeat purchases. Additionally, introducing a loyalty program can encourage repeat customers and build long-term relationships by offering discounts, loyalty points, or exclusive

access to new books, motivating users to return for future purchases and enhancing customer retention.

References

- https://getbootstrap.com/docs/5.3/exampls
- https://www.clankart.com
- https://www.w3schools.com/html/default.asp
- https://github.com/shanshibu/mini-project.git