

**Project Title:**

**Book Store WEB APP**

**Course:**

**Software Engineering**

**Submitted By:**

* **Muhammad Ibrahim** (Team Lead)
* **M Abdullah Bin Abdul Aziz**
* **Wajhi ur Rehman**

**Instructor:**

**Adnan Jelani**

**Submission Date:**

**22th December 2024**

Table of Contents

[1. Introduction 1](#_Toc185805603)

[Problem Description 1](#_Toc185805604)

[Stakeholders/Users 1](#_Toc185805605)

[Feasibility Study 1](#_Toc185805606)

[Purpose 1](#_Toc185805607)

[Advantages 1](#_Toc185805608)

[Disadvantages 1](#_Toc185805609)

[2. Proposed Solution 2](#_Toc185805610)

[Description of the Proposed Solution 2](#_Toc185805611)

[Description of the Proposed Solution 2](#_Toc185805612)

[Key Features 2](#_Toc185805613)

[Benefits of the Solution 2](#_Toc185805614)

[Front-end Design Sketch 3](#_Toc185805615)

[3. Software Requirement Specification Document (SRS) 5](#_Toc185805616)

[Functional Requirements 5](#_Toc185805617)

[1. User Functional Requirements: 5](#_Toc185805618)

[2. Admin Functional Requirements: 6](#_Toc185805619)

[Non-Functional Requirements 6](#_Toc185805620)

[1. Performance Requirements 6](#_Toc185805621)

[2. Scalability Requirements 6](#_Toc185805622)

[3. Usability Requirements 7](#_Toc185805623)

[4. Security Requirements 7](#_Toc185805624)

[5. Reliability Requirements 7](#_Toc185805625)

[6. Maintainability Requirements 7](#_Toc185805626)

[7. Compatibility Requirements 7](#_Toc185805627)

[8. Accessibility Requirements 7](#_Toc185805628)

[9. Ethical Requirements 7](#_Toc185805629)

[Use Case Diagrams/User Stories 8](#_Toc185805630)

[Assumptions/Constraints 12](#_Toc185805631)

[4. Project Plan 12](#_Toc185805632)

[Timeline and Milestones 12](#_Toc185805633)

[Gantt Chart 13](#_Toc185805634)

[Team Responsibilities 14](#_Toc185805635)

[Technology Stack and Tools 14](#_Toc185805636)

[5. Project Management Tools 14](#_Toc185805637)

[Overview of Usage 14](#_Toc185805638)

[Task Assignment and Sprint Details 14](#_Toc185805639)

[6. Testing 16](#_Toc185805640)

[Test Cases 16](#_Toc185805641)

[PHP Integration Test for Adding to Cart: 16](#_Toc185805642)

[User Acceptance Testing (UAT) 16](#_Toc185805643)

[Test Plan for UAT: 17](#_Toc185805644)

[Test Plan and Results Summary 17](#_Toc185805645)

[Automated Testing Using Tools 17](#_Toc185805646)

[Summary of testing: 18](#_Toc185805647)

[7. Technical Documentation 18](#_Toc185805648)

[ER Diagram 18](#_Toc185805649)

[8. User Manual 18](#_Toc185805650)

[Explanation of Features 18](#_Toc185805651)

[Usage Steps 18](#_Toc185805652)

[9. Deployment (Optional) 18](#_Toc185805653)

[GitHub: 18](#_Toc185805654)

[10. References 19](#_Toc185805655)

# 1. Introduction

## Problem Description

The traditional method of purchasing books involves physical stores and limited availability, which can be inconvenient for users. This project aims to create an online platform that allows users to browse, search, and purchase books from the comfort of their homes. By offering a seamless online experience, we address issues such as limited accessibility, lack of time, and the need for broader choices.

## Stakeholders/Users

* **Users/Customers:** Book enthusiasts who need an easy way to browse and purchase books or Students and researchers seeking academic resources.
* **Admin:** Responsible for managing the book inventory, orders, and user accounts. Monitor sales, revenue, and promotions.

## Feasibility Study

### Purpose

The application is needed to provide an easy-to-use platform where users can browse, purchase, and review books. It aims to bridge the gap between users and bookstores by providing detailed book information and a smooth purchasing process.

### Advantages

* **Accessibility:** Users can shop from anywhere, anytime.
* **Efficiency:** Automates inventory management and order processing.
* **Cost-Effective:** Reduces costs associated with physical stores.
* **Environmental Impact:** Promotes eco-friendly practices by reducing paper-based catalogs.

### Disadvantages

* Initial development costs and time.
* Dependency on a reliable internet connection for access.
* Requires technical knowledge for management.
* Potential cybersecurity threats.

# 2. Proposed Solution

## Description of the Proposed Solution

The Online Book Store is a web application that provides a user-friendly interface for customers to browse, search, and purchase books. It includes features like book reviews, ratings, and author details to enhance the user experience. Admins can manage inventory, monitor orders, and track sales.

## Description of the Proposed Solution

The Online Book Store is a web application that provides a user-friendly interface for customers to browse, search, and purchase books. It includes features like book reviews, ratings, and author details to enhance the user experience. Admins can manage inventory, monitor orders, and track sales.

## Key Features

* **Book Browsing and Searching**:
  + Categories like genre, author, and price range.
  + Advanced search by title, author, or ISBN.
  + Filters for price, rating, and publication year.
* **Book Details**:
  + Summaries, descriptions, and cover images.
  + Author bio and user reviews.
* **Shopping Cart**:
  + Add, update, or remove items.
  + Save for later options.
* **Checkout Process**:
  + Address input, payment method selection, and order confirmation.
* **User Account**:
  + Sign up, log in, and manage profiles.
  + View order history and Wishlist.
* **Admin Features**:
  + Inventory, order, and user management.
  + Promotion management.

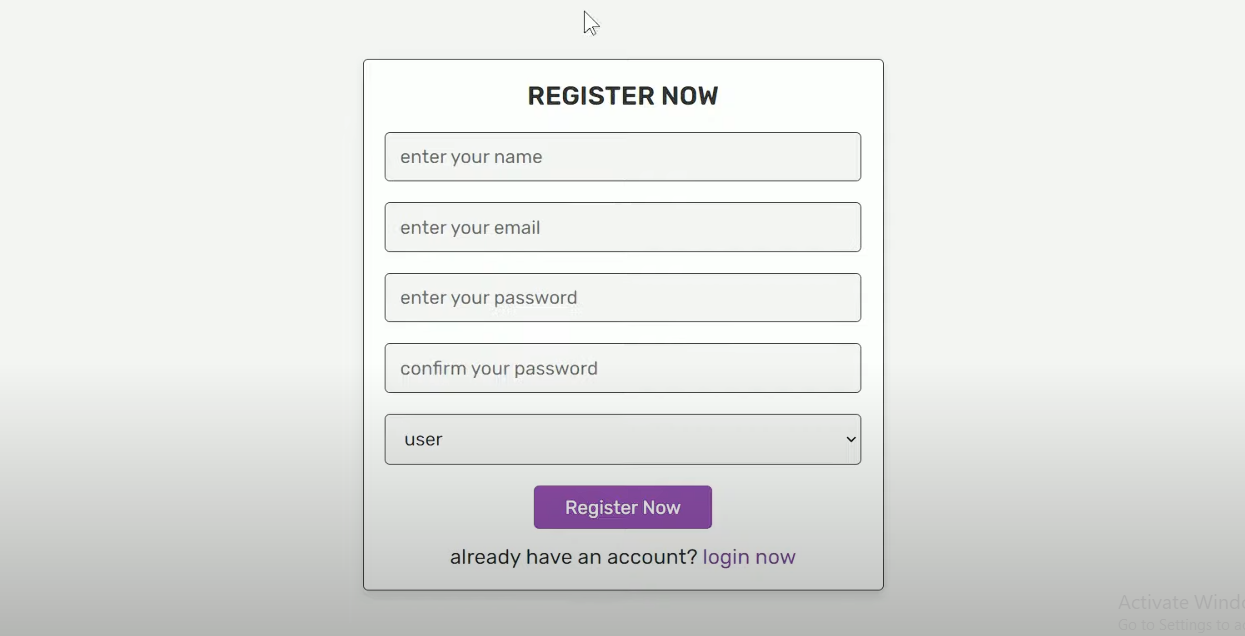
## Benefits of the Solution

* **Efficiency:** Streamlines book purchasing.
* **User-Friendly:** Intuitive interface for all users.
* **Scalability:** Can handle increasing inventory and users.
* **Security:** Ensures secure transactions.

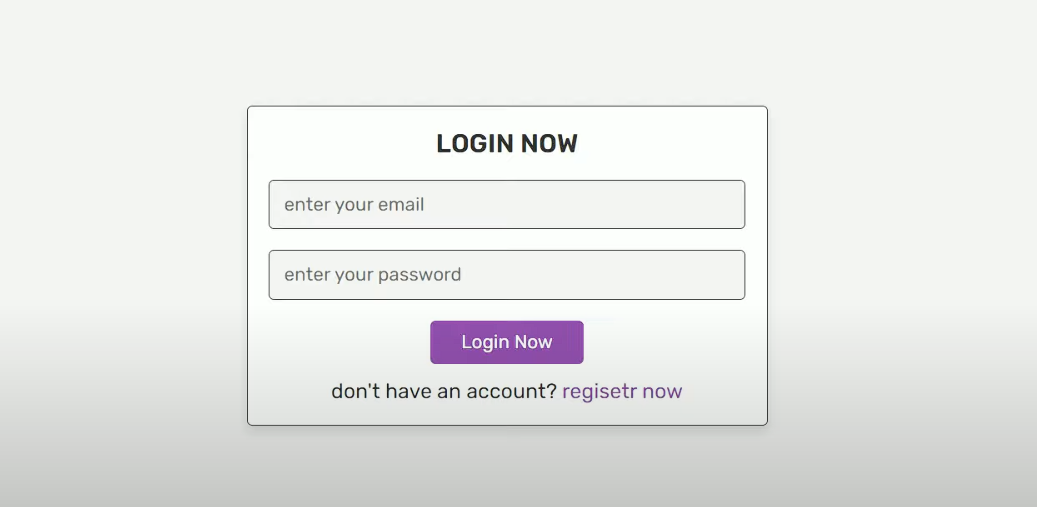
## Front-end Design Sketch

The project will include following pages:

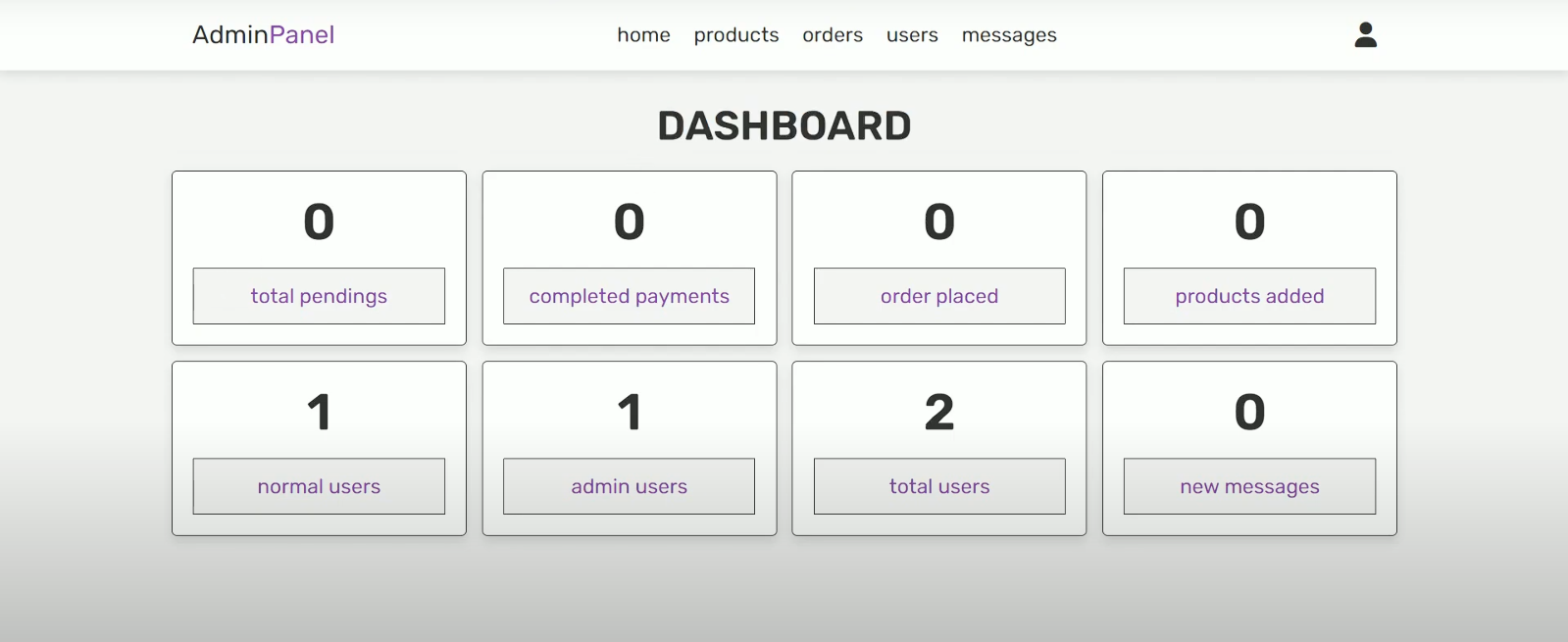
**Register Form:**



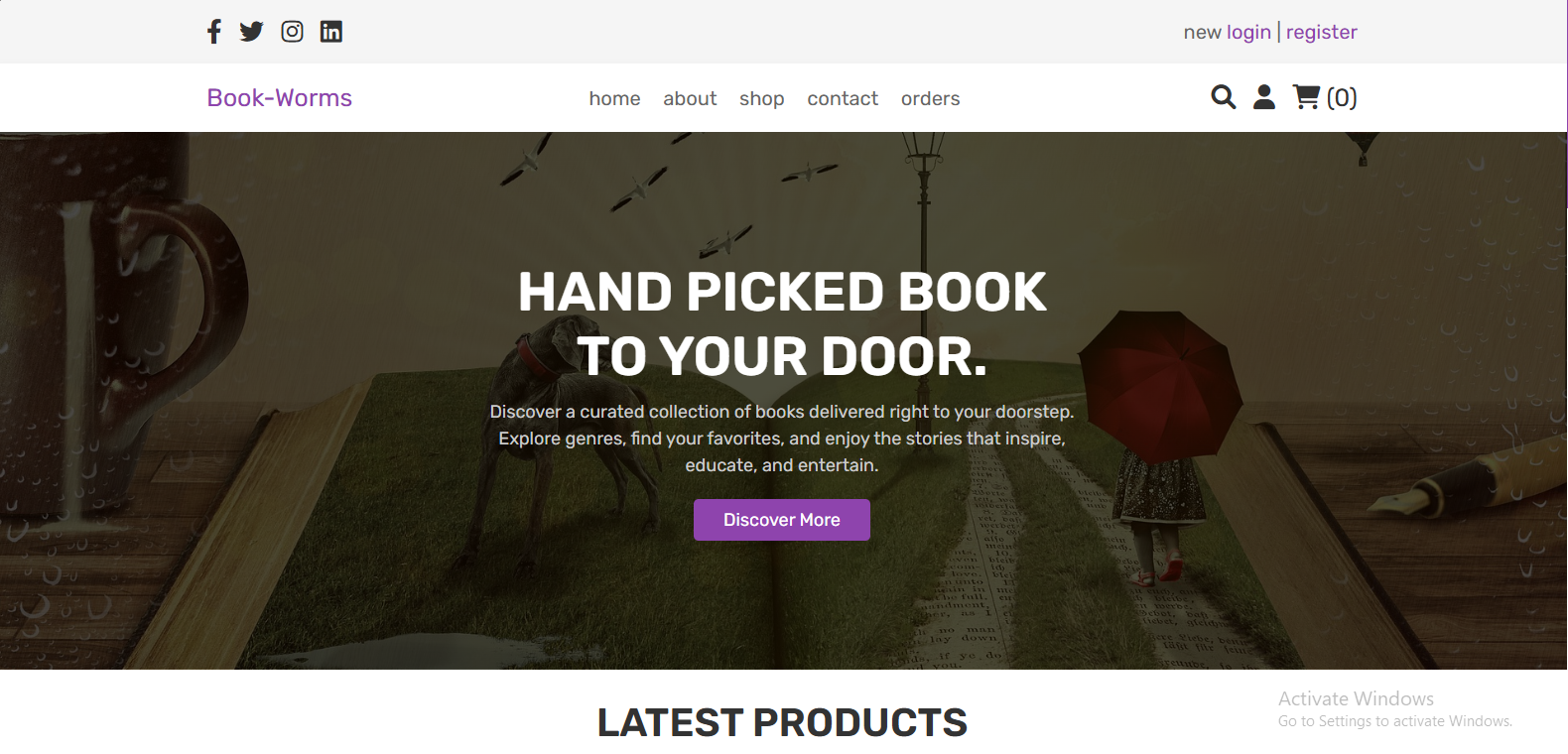
**Login Page:**



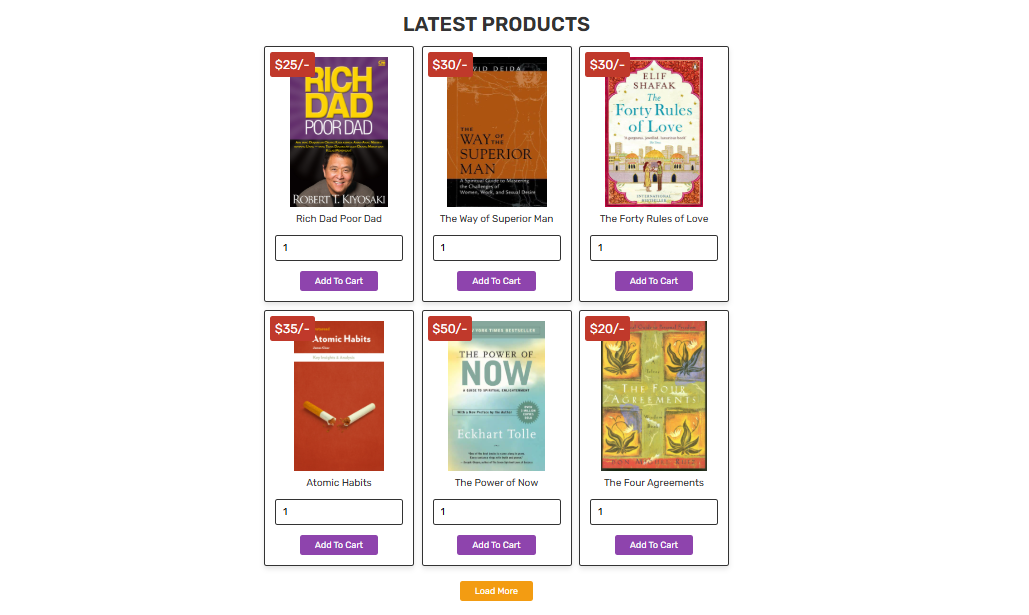
**Admin Panel Page:**



**User Home Page:**

****

**Products Catalog:**



# 3. Software Requirement Specification Document (SRS)

## Functional Requirements

### User Functional Requirements:

* **User Registration:**
* Users can create accounts by providing essential details like name, email, and password.
* **User Login**:
* Registered users can log in to access their personalized dashboard.
* **Book Browsing**:
* Users can browse books by categories like genre, author, or price range.
* **Advanced Search**:
* Users can search for books by title, author, or ISBN.
* **Filters**:
* Users can filter results based on price, rating, and publication year.
* **Book Details**:
* Users can view book summaries, author bios, and detailed descriptions.
* **Shopping Cart**:
* Users can add books to a cart, update quantities, or remove items.
* **Wishlist**:
* Users can add books to a Wishlist for future reference.
* **Checkout Process**:
* Users can proceed to checkout by providing their address and selecting a payment method.
* **Order History**:
* Users can view their past orders, including details like status and items purchased.
* **Contact Support**:
* Users can submit inquiries or feedback via the contact form.

### Admin Functional Requirements:

* **Admin Login**:
* Admins can securely log into the admin dashboard.
* **Manage Products**:
* Admins can add, edit, or delete books, including uploading cover images.
* **Inventory Management**:
* Admins can assign categories, tags, and track stock levels.
* **Order Management**:
* Admins can view, update, and manage orders, including their statuses (e.g., pending, shipped).
* **User Management**:
* Admins can view, block, or delete user accounts.
* **Contact Management**:
* Admins can review and respond to user inquiries or feedback.
* **Promotion Management**:
* Admins can create and manage discounts, offers, or featured books.
* **Analytics and Reports**:
* Admins can track sales trends, monitor revenue, and view user activity reports.

## Non-Functional Requirements

### 1. Performance Requirements

* The system should respond to user activity, for example, pages loading, adding items to cart within few seconds for normal operation.
* It should support multiple users at the same time without considerable performance degradation.
* Database queries should be executed efficiently enough to avoid any kind of problem in operation.

### 2. Scalability Requirements

* The system will be designed to support the growth factor, including new features, books, categories, and user accounts.
* It will support scaling horizontally (adding servers) or vertically (upgrading servers) as traffic increases.

### 3. Usability Requirements

* The application will provide an user-friendly interface.
* Users will be able to navigate the website easily on both desktop and mobile devices.
* All forms and processes (e.g., checkout, registration) will be simple, clear, and validated with user-friendly error messages.

### 4. Security Requirements

* All sensitive information (passwords, personal information) will be encrypted (hashed passwords).
* The system will prevent unauthorized access via user authentication and session management.
* The system will protect against common web threats such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

### 5. Reliability Requirements

* The system will have maximum uptime with high availability for users.
* Daily database backups will be made to avoid data loss.
* In the event of a crash, the system will recover in few minutes with redundant servers or backup plans.

### 6. Maintainability Requirements

* The codebase will be based on clean coding practices with comments for readability and easier debugging.
* The system will facilitate easy updates to features, UI, and backend without major disruptions.

### 7. Compatibility Requirements

* The website will be compatible with modern web browsers (e.g., Chrome, Firefox, Safari, Edge).
* It will work smoothly on other operating systems such as Windows, macOS, Android, iOS.
* Responsive design is guaranteed for multiple screen sizes (desktop, tablet, mobile).

### 8. Accessibility Requirements

* The application will be accessible to everyone including those with disabilities under web accessibility guidelines (e.g. WCAG 2.1)
* Features like high contrast mode, readable font and navigation using the keyboard should be there.

### 9. Ethical Requirements

* The application will not collect or make use of user data without consent.

## Use Case Diagrams/User Stories

1. **Admin Use Cases:**

* **Book Management:** Add, Edit and Delete book entries. Upload book cover images. Assign categories and tags.
* **Order Management:** View all orders with statuses (pending, shipped, delivered). Update order status.
* **User Management:** View and manage user accounts. Block or delete accounts if necessary.
* **Reports and Analysis:** Track sales trends such as most purchased books, revenue reports.
* **Promotion Management:** Highlight featured or newly added books. Email for new arrivals, or email for discounts.

1. **Admin User Stories:**

Following are few user stories;

* + **Book Management**:

As an admin, I want to add, edit, and delete book entries and upload cover images so that I can keep the inventory up-to-date and visually appealing for users.

* + **Order Management**:

As an admin, I want to view and update order statuses (e.g., pending, shipped, delivered) so that I can efficiently manage customer orders and ensure timely delivery.

* + **User Management**:

As an admin, I want to manage user accounts by viewing, blocking, or deleting them if necessary to maintain a secure and user-friendly platform.

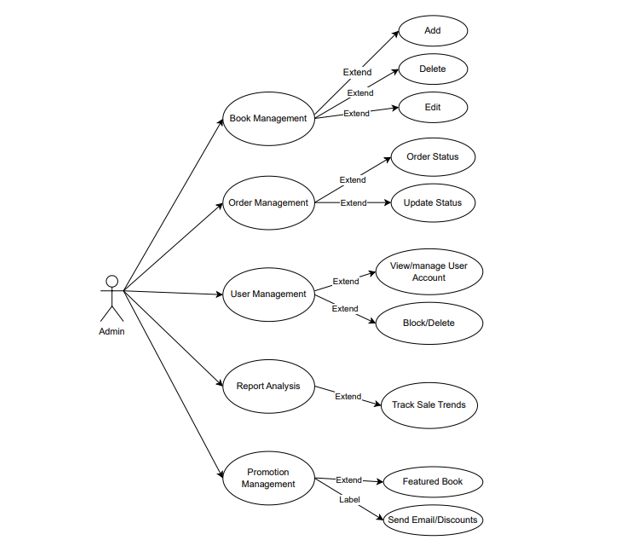
* + **Reports and Analysis**:

As an admin, I want to track sales trends and generate revenue reports so that I can make data-driven decisions to improve the store's performance.

* + **Promotion Management**:

As an admin, I want to highlight featured books and send promotional emails about discounts or new arrivals so that I can engage customers and increase sales.

**Admin Use Case Diagram**

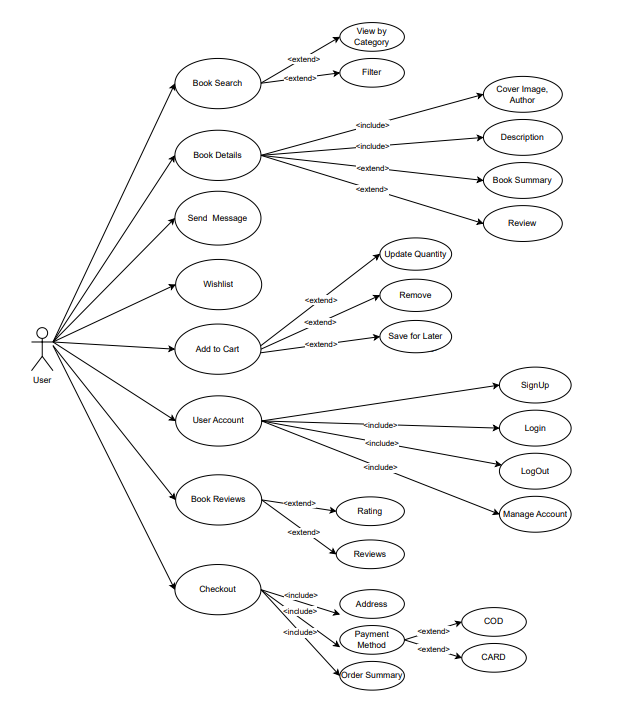
****

1. **User Use Cases:**

* **Book Browsing and Searching** (View books by category (title, author, price range, etc.). Filters (price, rating, publication year).)
* **Shopping Cart** (Add to cart, update quantity in cart, remove items from cart. Save for later option.)
* **Checkout Process** (Address input form and payment method selection (cash on delivery or online payment simulation). Order summary and confirmation.)
* **User Account Management** (Sign up, log in, and logout functionality. Manage profile (update personal information, password). View order history.)
* **Wishlist** (Add books to a Wishlist for future reference.)
* **Book Reviews** (Users can leave ratings and reviews for books.)
* **Send Message:** User can send message to contact admin.

1. **User User Stories:**

* **Book Browsing and Searching**: As a user, I want to browse books by categories and use filters like price, rating, and publication year so that I can quickly find books of interest.
* **Shopping Cart and Wishlist**: As a user, I want to add books to my cart or Wishlist, update quantities, and save items for later so that I can organize my purchases conveniently.
* **Checkout Process**: As a user, I want to provide my address and select a payment method to confirm my order, so that I can complete my purchase efficiently.
* **User Account and Interaction**: As a user, I want to sign up, log in, manage my profile, view my order history, and contact the admin so that I can personalize my experience and resolve issues effectively.



**User Use Case Diagram**

## Assumptions/Constraints

* The application assumes users have an internet connection.
* Payment integration will be simulated without real transactions.
* The application is designed for small-to-medium-sized businesses.

# 4. Project Plan

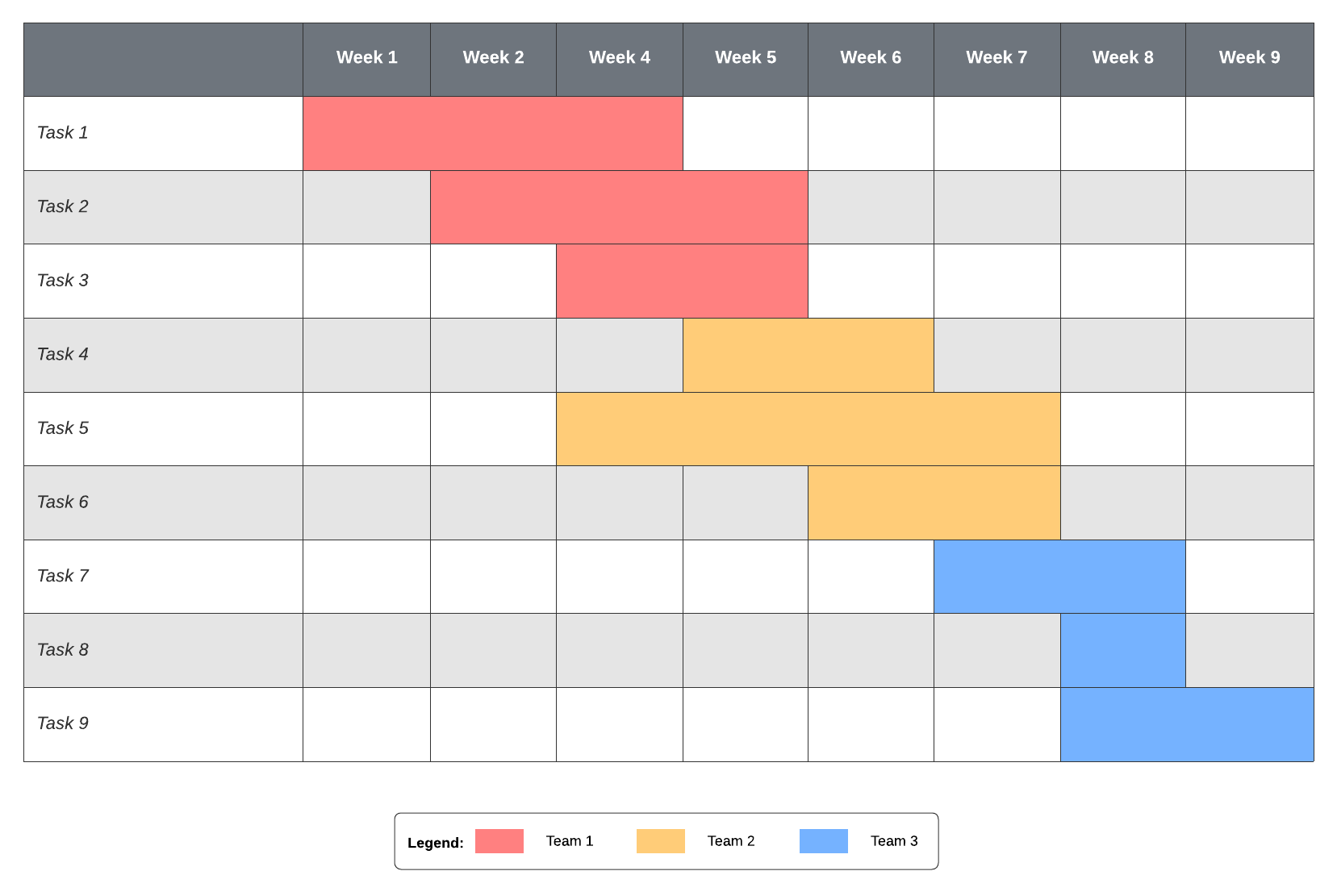
## Timeline and Milestones

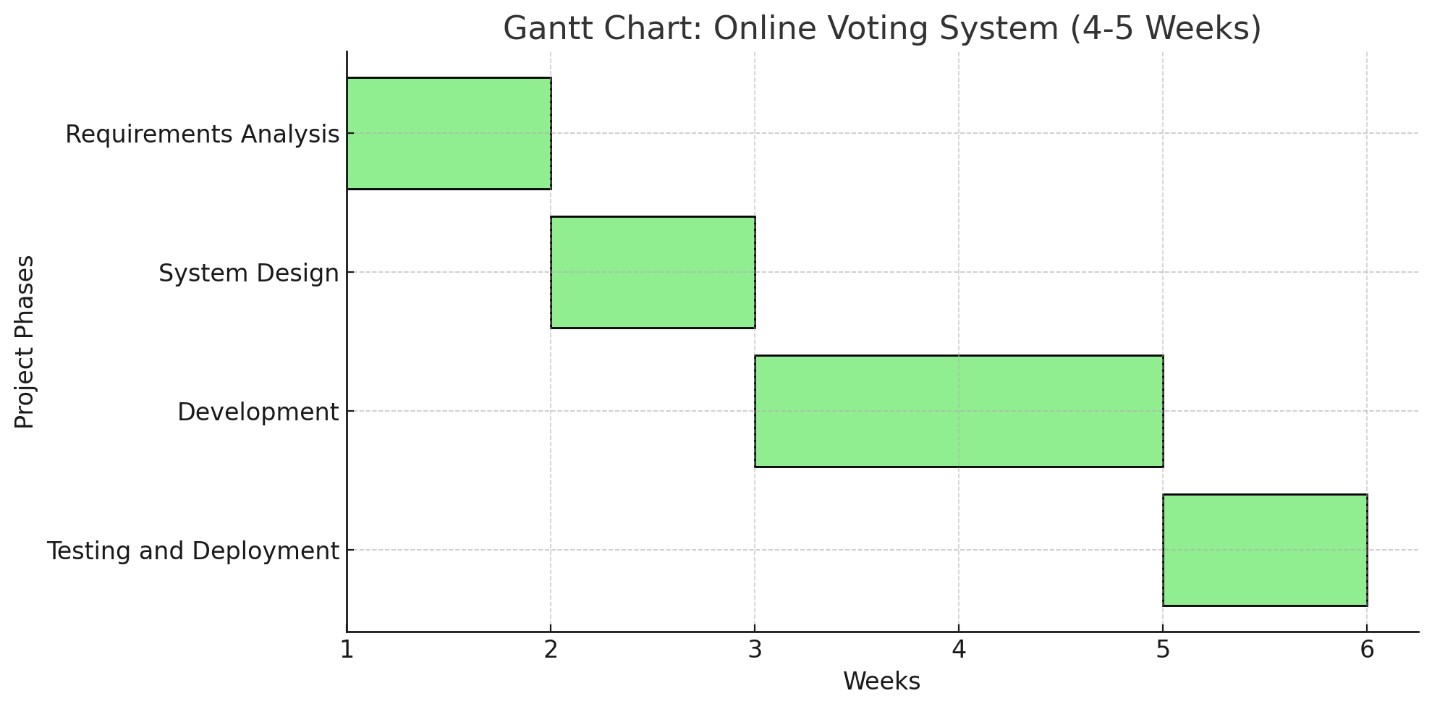
|  |  |  |
| --- | --- | --- |
| **Phases** | **Tasks** | **Weekly Duration** |
| Phase 1 | **UI/UX Design** | 1-2 |
| Phase 2 | **Frontend Implementation** | 3-4 |
| Phase 3 | **Backend Setup** | 5 |
| Phase 4 | **Incremental Integration** | 6-7 |
| Phase 5 | **Testing and Deployment** | 8 |

The project is divided into weekly tasks as under:

* **Weeks 1-2:** UI/UX Design. Focus on designing a responsive and user-friendly interface to enhance the user experience.
* **Weeks 3-4:** Frontend Implementation. Development of the application's frontend using HTML and CSS, ensuring a clean and intuitive design.
* **Week 5:** Backend Setup. Configuring the backend using PHP and integrating it with a MySQL database for data management.
* **Weeks 6-7:** Incremental Integration. Gradual integration of frontend and backend functionalities, testing features incrementally for compatibility.
* **Week 8:** Testing and Deployment. Conducting comprehensive testing to ensure functionality and deploying the completed application.

## Gantt Chart





## Team Responsibilities

* Muhammad Ibrahim: Frontend development (HTML/CSS design).
* M Abdullah Bin Abdul Aziz: Backend development (PHP and database integration).
* Wajhi Ur Rehman: Testing and documentation.

## Technology Stack and Tools

* Frontend: HTML, CSS
* Backend: PHP
* Database: MySQL
* Tools: VS Code, XAMPP, JIRA (for Project Management), Canva (for design mockups)

# 5. Project Management Tools

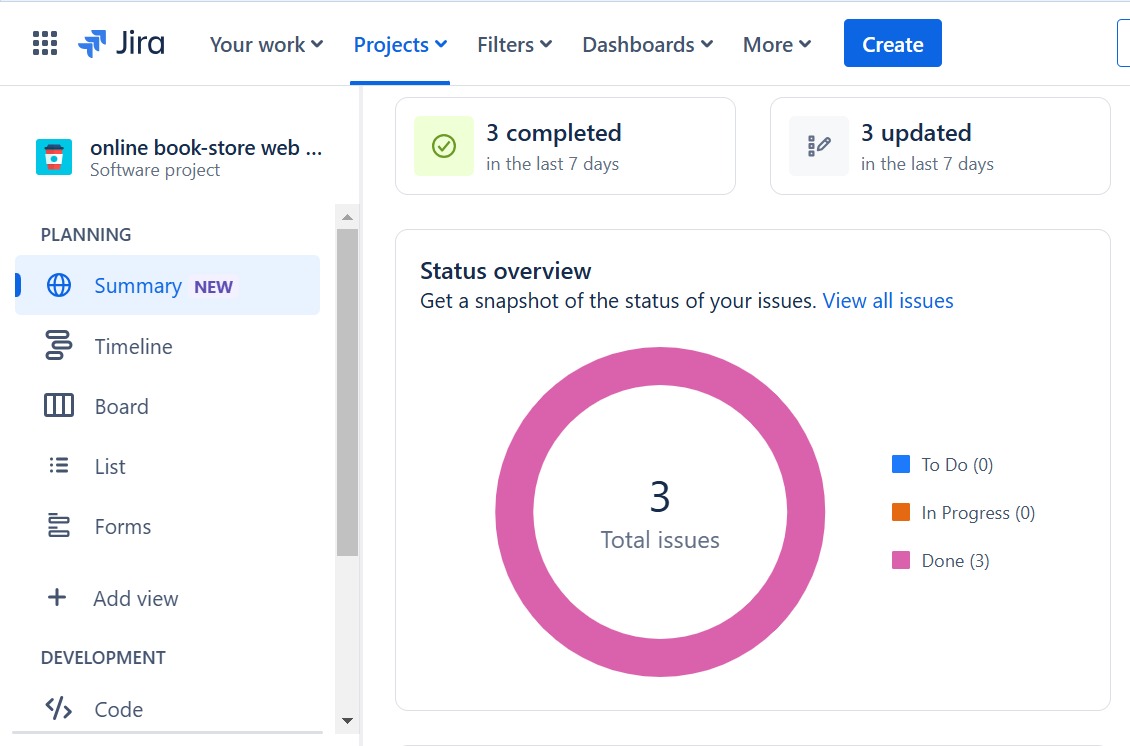
## Overview of Usage

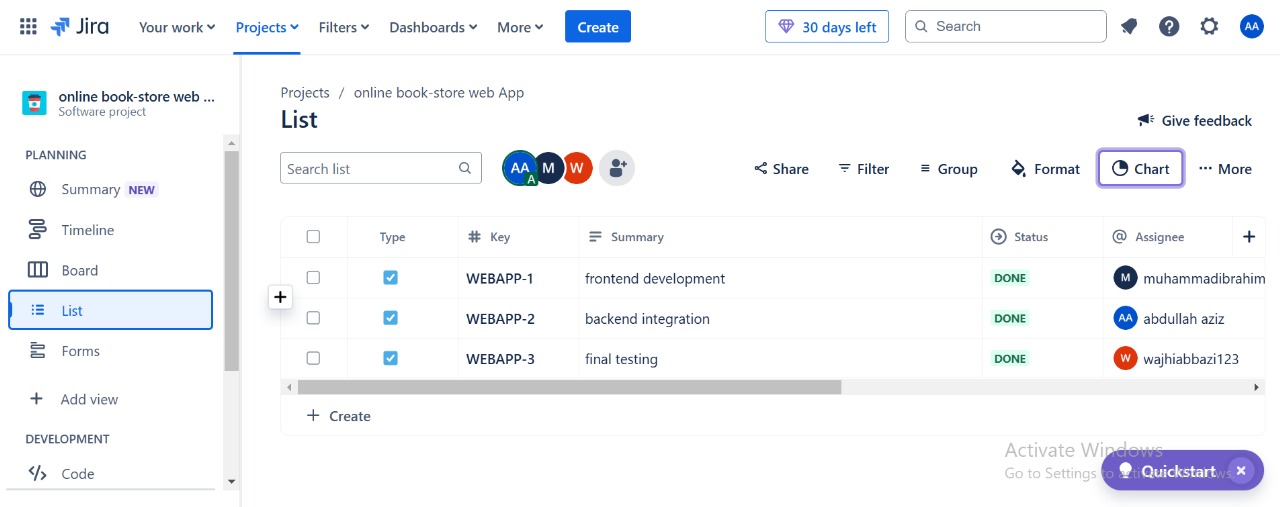
Jira was used to manage tasks and sprints effectively. Tasks were categorized as frontend, backend, or testing.

## Task Assignment and Sprint Details

* Sprint 1: Frontend development.
* Sprint 2: Backend integration.
* Sprint 3: Final testing and deployment.

**Screenshots:**





# 6. Testing

## Test Cases

Integration tests are conducted to check how different modules interact with each other. Following are the details of the tests.

## PHP Integration Test for Adding to Cart:

CODE:

<?php

// Database connection

$conn = new mysqli('localhost', 'root', '', 'online\_bookstore');

// Test data

$user\_id = 1; // Assume a user exists with this ID

$book\_id = 2; // Assume a book exists with this ID

$quantity = 1;

// Add to cart

$query = "INSERT INTO cart (user\_id, book\_id, quantity) VALUES ($user\_id, $book\_id, $quantity)";

$result = $conn->query($query);

// Verify if the item was added to the cart

if ($result) {

$verify\_query = "SELECT \* FROM cart WHERE user\_id = $user\_id AND book\_id = $book\_id";

$verify\_result = $conn->query($verify\_query);

if ($verify\_result->num\_rows > 0) {

echo "Test Case: Add to Cart - PASSED\n";

} else {

echo "Test Case: Add to Cart - FAILED\n";

}

} else {

echo "Test Case: Add to Cart - FAILED: " . $conn->error . "\n";

}

// Clean up test data

$conn->query("DELETE FROM cart WHERE user\_id = $user\_id AND book\_id = $book\_id");

$conn->close();

?>

## User Acceptance Testing (UAT)

User Acceptance Testing (UAT) was also conducted which ensured us that the application has met the needs of end users. This is done by letting a small group of users interact with the system and provide feedback.

### ****Test Plan for UAT:****

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Test Case Description** | **Expected Result** | **Status** |
| Book Search | Search for books by title | Displays correct results | Passed |
| Add to Cart | Add a book to the shopping cart | |  | | --- | | Book appears in the cart |  |  | | --- | |  | | Passed |
| Checkout | Complete an order | |  | | --- | | Generates an order confirmation page |  |  | | --- | |  | | Passed |
| Admin Inventory Update | Add a new book to the inventory | |  | | --- | | Book appears on the shop page |  |  | | --- | |  | | Passed |

### Test Plan and Results Summary

* Login functionality: Passed.
* Book search and checkout: Passed.
* Admin panel operations: Passed.

## Automated Testing Using Tools

CODE:

Automated testing tools can also be used to streamline the testing process. Automated testing was also conducted which ensured us that the application has met the needs of end users.

use PHPUnit\Framework\TestCase;

class UserTest extends TestCase {

public function testUserRegistration() {

// Simulate input data

$email = "test\_user@example.com";

$password = "password123";

// Hash password

$hashed\_password = password\_hash($password, PASSWORD\_BCRYPT);

// Expected behavior

$this->assertNotEmpty($hashed\_password, "Password hashing failed.");

$this->assertTrue(password\_verify($password, $hashed\_password), "Password verification failed.");

}

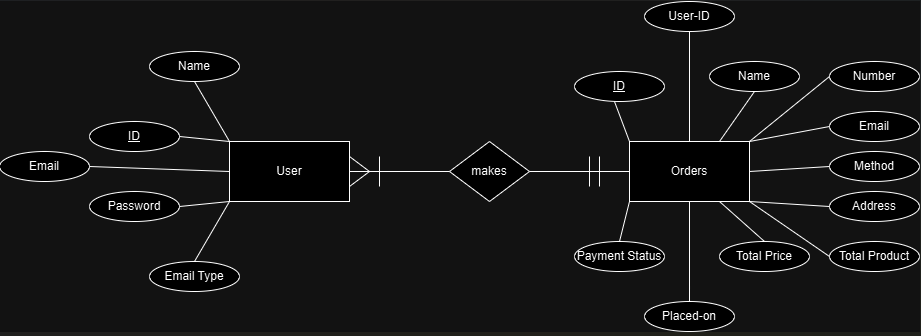
}

## Summary of testing:

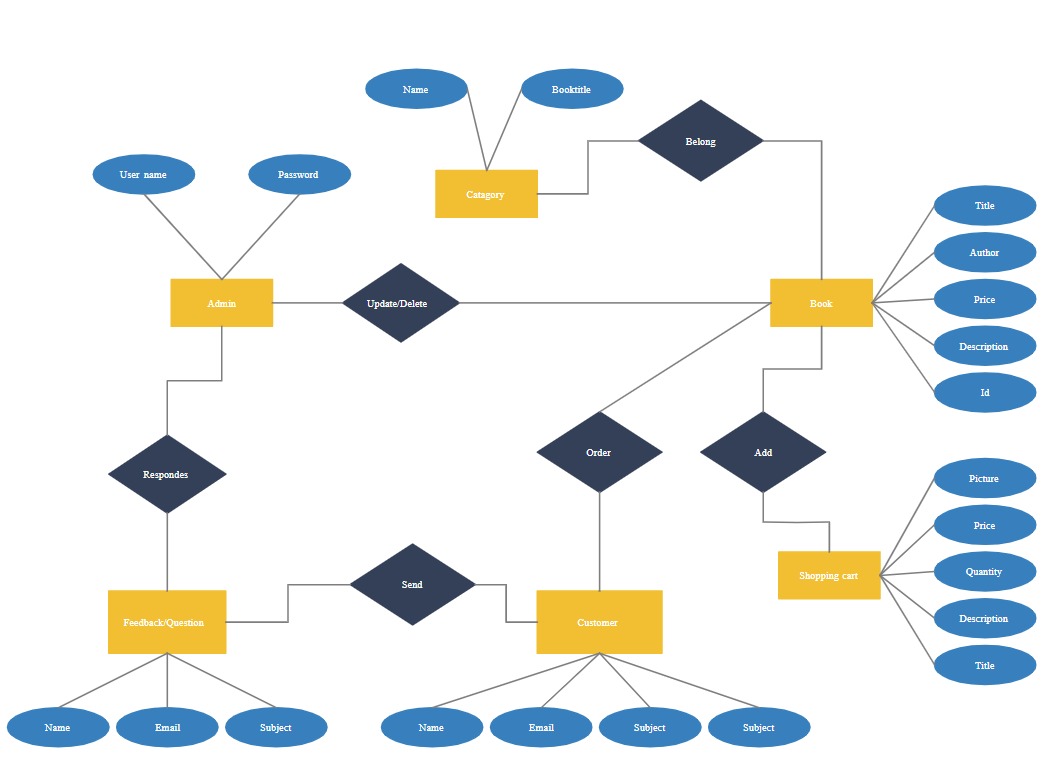
* **Unit Testing:**
* Validates individual features like login and registration.
* **Integration Testing:**
* Tests interactions between modules, like adding to cart and checkout.
* **User Testing:**
* Collects feedback from real users to identify usability issues.
* **Automated Testing:**
* Uses tools like PHPUnit to automate repetitive test cases.

# 7. Technical Documentation

## ER Diagram



Following is the generic ER diagram for this project:



# 8. User Manual

## Explanation of Features

1. Register and log in to your account.
2. Browse books using filters.
3. Add books to your cart or Wishlist.
4. Complete the checkout process securely.

## Usage Steps

1. Access the homepage.
2. Use the search bar or categories to find books.
3. Add items to the cart and proceed to checkout.

# 9. Deployment (Optional)

Deployment could not be completed due to time constraints but will be completed in the future.

GitHub:  
[GitHub Repository for Online Book Store](https://github.com/ibrahiim-1/Online-BookStore-Web-APP)

# 10. References

1. W3Schools - PHP Tutorials
2. MySQL Documentation
3. YouTube – MySQL integration