Project Title: BookStore Management System

The BookStore Management System (BMS) is a web-based application designed to manage bookstore operations efficiently. The system is modular and follows the MVC architecture for flexibility and scalability, making it compatible with the ASP.NET Core MVC framework.

Core Modules:

1. Book Management – Handles creation, updates, and management of books.
2. Order Processing – Facilitates order submission, processing.
3. Customer Management – Manages customer profiles and interactions.
4. Inventory Management – Manages stock levels.
5. User Management – Manages user authentication, authorization, and profiles.

Assumptions:

1. The application will be deployed locally during development using a relational database (e.g., MySQL or MS SQL).
2. Security mechanisms will include role-based authentication.
3. ORM framework (Entity Framework for .NET) will handle database interactions.
4. No containerization will be used for local deployment.

Module-Level Design:

3.1 Book Management Module Purpose: Handles operations for book lifecycle management.

- **Controller:**
    - AdminController
        - createBook(bookData)
        - updateBook(bookId, bookData)
        - deleteBook(bookId)
- **Model:**
    - Entity: Book
    - Attributes:
        - bookId (PK)

        - title (VARCHAR)
        - author (VARCHAR)
        - price (DECIMAL)
        - publicationDate (DATE)
        - isbn (VARCHAR)

3.2 Order Processing Module Purpose: Facilitates order management.

- **Controller:**
    - UserController
        - submitOrder(orderData)
        - getOrderDetails(orderId)
- **Model:**
    - Entity: Order
    - Attributes:
        - orderId (PK)
        - customerId (FK)
        - orderAmount (DECIMAL)
        - orderStatus (ENUM)
        - orderDate (DATE)
        - deliveryDate (DATE)

3.3 Customer Management Module Purpose: Manages customer profiles and data.

- **Controller:**
    - AdminController
        - addCustomer(customerData)
        - updateCustomer(customerId, customerData)
        - getCustomerDetails(customerId)
- **Model:**
    - Entity: Customer
    - Attributes:

        - customerId (PK)
        - name (VARCHAR)
        - email (VARCHAR)
        - phone (VARCHAR)
        - address (VARCHAR)

3.4 Inventory Management Module Purpose: Manages stock levels and inventory tracking.

- **Controller:**
    - AdminController
        - updateStock(bookId, quantity)
        - getStockDetails(bookId)
- **Model:**
    - Entity: Inventory
    - Attributes:
        - inventoryId (PK)
        - bookId (FK)
        - quantity (INT)

3.5 User Management Module Purpose: Manages authentication and role-based access control.

- **Controller:**
    - AccountController
        - registerUser(userData)
        - loginUser(username, password)
- **Service:**
- **Model:**
    - Entity: User
    - Attributes:
        - userId (PK)
        - username (VARCHAR)

        - password (VARCHAR, Encrypted)
        - role (ENUM)

Admin Controller:

- **Manage Users:**
    - Controller: AdminController
        - manageUsers()
- **Manage Book Stock:**
    - Controller: AdminController
        - manageBookStock()
- **View All Orders:**
    - Controller: AdminController
        - viewAllOrders()

User Controller:

- **Manage Profile:**
    - Controller: UserController
        - manageProfile(userId)
- **All Orders by User:**
    - Controller: UserController
        - getAllOrdersByUser(userId)

Database Schema:

4.1 Table Definitions:

1. **Book Table**

```
CREATE TABLE Book (
 bookId INT PRIMARY KEY AUTO_INCREMENT,
 title VARCHAR(100),
 author VARCHAR(100),
 price DECIMAL(10, 2),
 publicationDate DATE,
```

```
 isbn VARCHAR(20)
);
```

2. **Order Table**

```
CREATE TABLE Order (
 orderId INT PRIMARY KEY AUTO_INCREMENT,
 customerId INT,
 orderAmount DECIMAL(10, 2),
 orderStatus ENUM('PENDING', 'SHIPPED', 'DELIVERED', 'CANCELLED'),
 orderDate DATE,
 deliveryDate DATE,
 FOREIGN KEY (customerId) REFERENCES Customer(customerId)
);
```

3. **Customer Table**

```
CREATE TABLE Customer (
 customerId INT PRIMARY KEY AUTO_INCREMENT,
 name VARCHAR(100),
 email VARCHAR(100),
 phone VARCHAR(15),
 address TEXT
);
```

4. **Inventory Table**

```
CREATE TABLE Inventory (
 inventoryId INT PRIMARY KEY AUTO_INCREMENT,
 bookId INT,
 quantity INT,
 FOREIGN KEY (bookId) REFERENCES Book(bookId)
);
```

5. **User Table**

```
CREATE TABLE User (
 userId INT PRIMARY KEY AUTO_INCREMENT,
 username VARCHAR(50) UNIQUE,
 password VARCHAR(255),
 role ENUM('ADMIN', 'USER')
);
```

6. **CardItem Table**

```
CREATE TABLE CardItem (
 cardItemId INT PRIMARY KEY AUTO_INCREMENT,
 userId INT,
 bookId INT,
 quantity INT,
 FOREIGN KEY (userId) REFERENCES User(userId),
 FOREIGN KEY (bookId) REFERENCES Book(bookId)
);
```

7. **OrderItems Table**

```
CREATE TABLE OrderItems (
 orderItemId INT PRIMARY KEY AUTO_INCREMENT,
 orderId INT,
 bookId INT,
 quantity INT,
 price DECIMAL(10, 2),
 FOREIGN KEY (orderId) REFERENCES Order(orderId),
 FOREIGN KEY (bookId) REFERENCES Book(bookId)
);
```

Conclusion: This document provides a comprehensive low-level design for the BookStore Management System, ensuring modularity, security, and compatibility for development in ASP.NET Core MVC.