Online Book Store

Project Description:

The project aims to develop an online book store system where users can search, browse, and purchase books. The system will have features for both users and administrators. Users can search for books, add them to their shopping carts, manage their carts, and place orders. Administrators can manage the inventory by adding, updating, or deleting books, authors, categories, and subcategories.

Functional Requirements:

1. User Management:
   - Users can register for an account.
   - Users can log in to their accounts.
   - Users can update their profile information.
   - Users can reset their passwords if forgotten.

2. Search and Browse:

- Users can search for books by title, author, or category.

- Users can browse through available books.

- Users can view detailed information about each book, including its price, description, and availability.

3. Shopping Cart:

- Users can add books to their shopping carts.

- Users can view the contents of their shopping carts.

- Users can update the quantity of books in their carts.

- Users can remove books from their carts.

4. Order Management:

- Users can place orders for the books in their shopping carts.

- Users can view their order history.

- Administrators can view all orders placed in the system.

5. Admin Panel:

- Administrators can add new books to the inventory.
- Administrators can update existing book information.
- Administrators can delete books from the inventory.
- Administrators can manage authors, categories, and subcategories.

Non-functional Requirements:

1. Performance:

- The system should be responsive and handle multiple user requests simultaneously.
- Book search and browsing operations should be fast, even with a large inventory.

2. Security:

- User authentication and authorization should be implemented to ensure that only authorized users can access certain features.

- User passwords should be securely stored using encryption techniques.

3. Scalability:

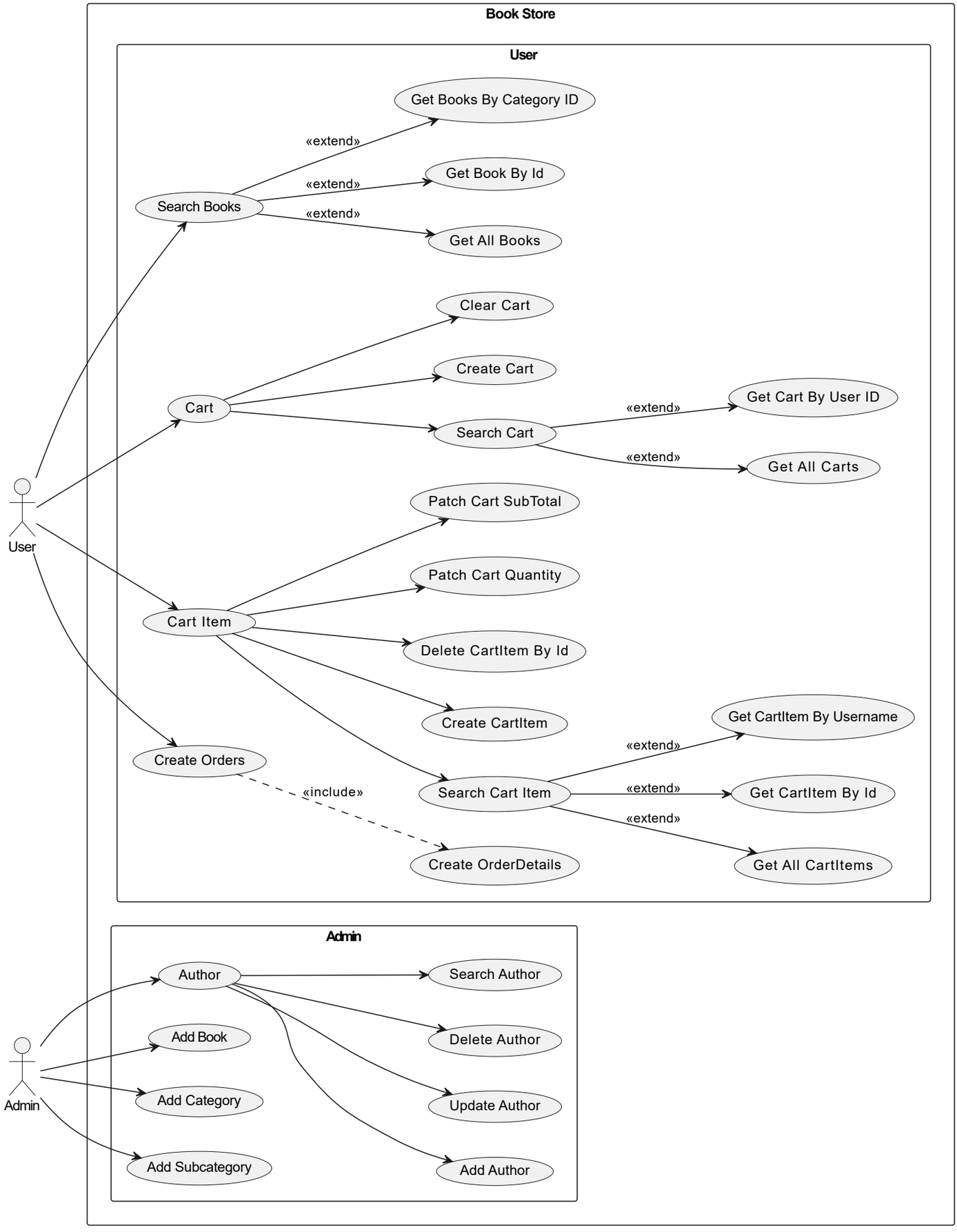
- The system should be designed to handle a growing number of users and books without significant performance degradation.

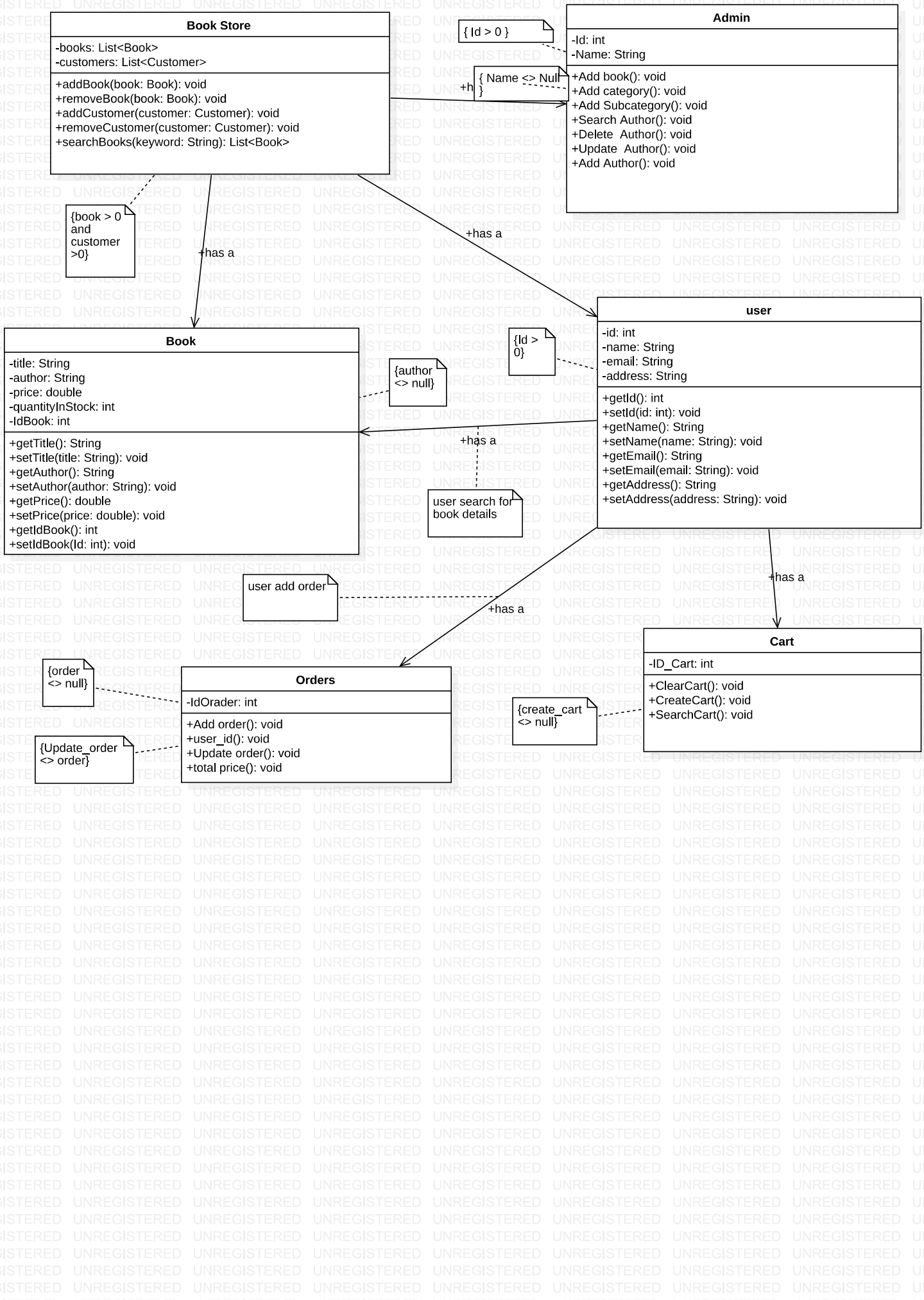
- It should be easy to scale the system by adding more servers or resources as needed.

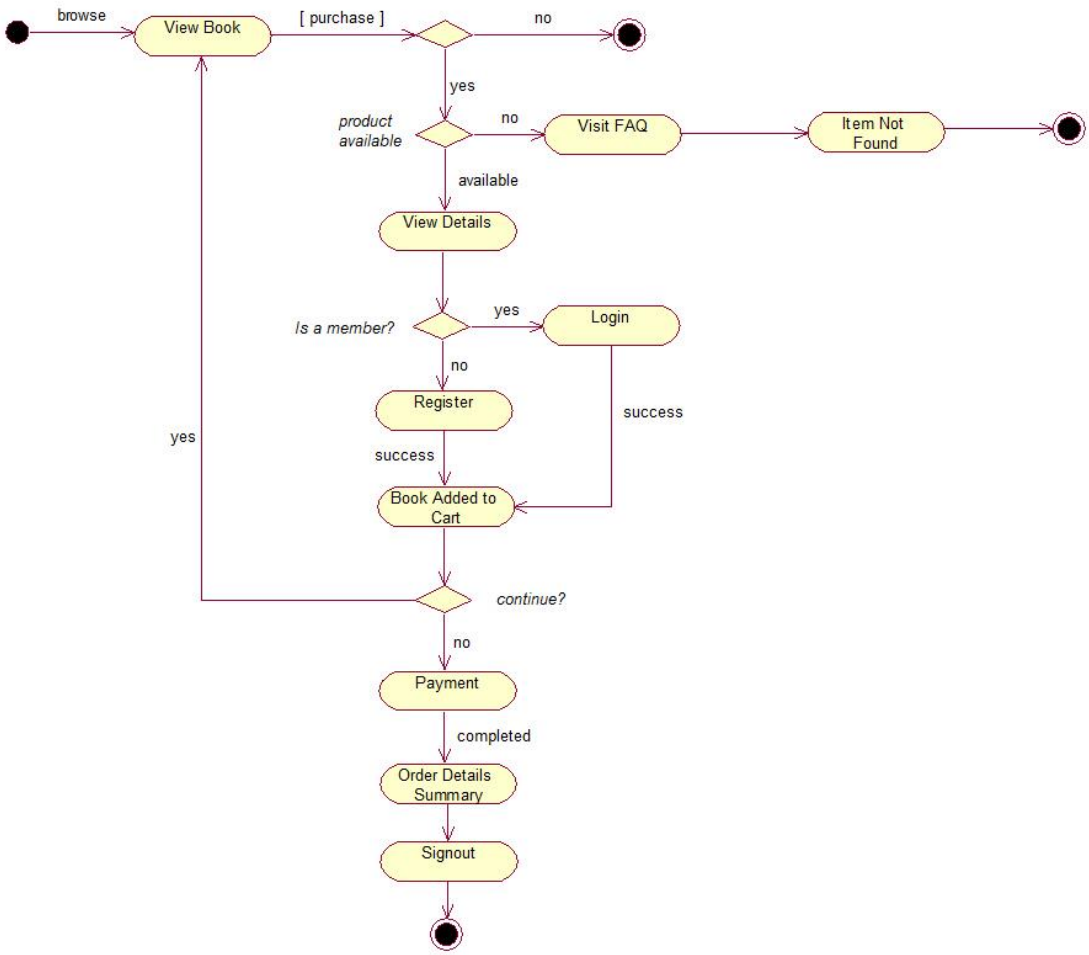
4. Reliability:

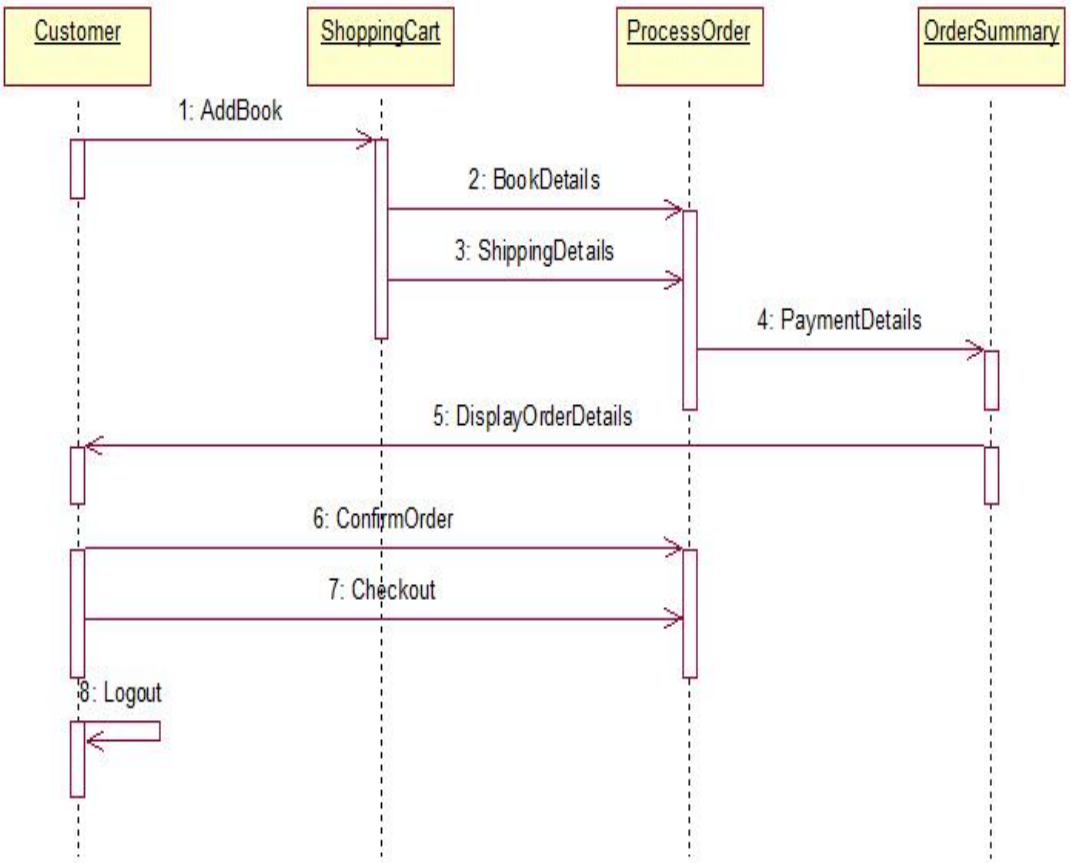
- The system should be highly reliable, with minimal downtime.

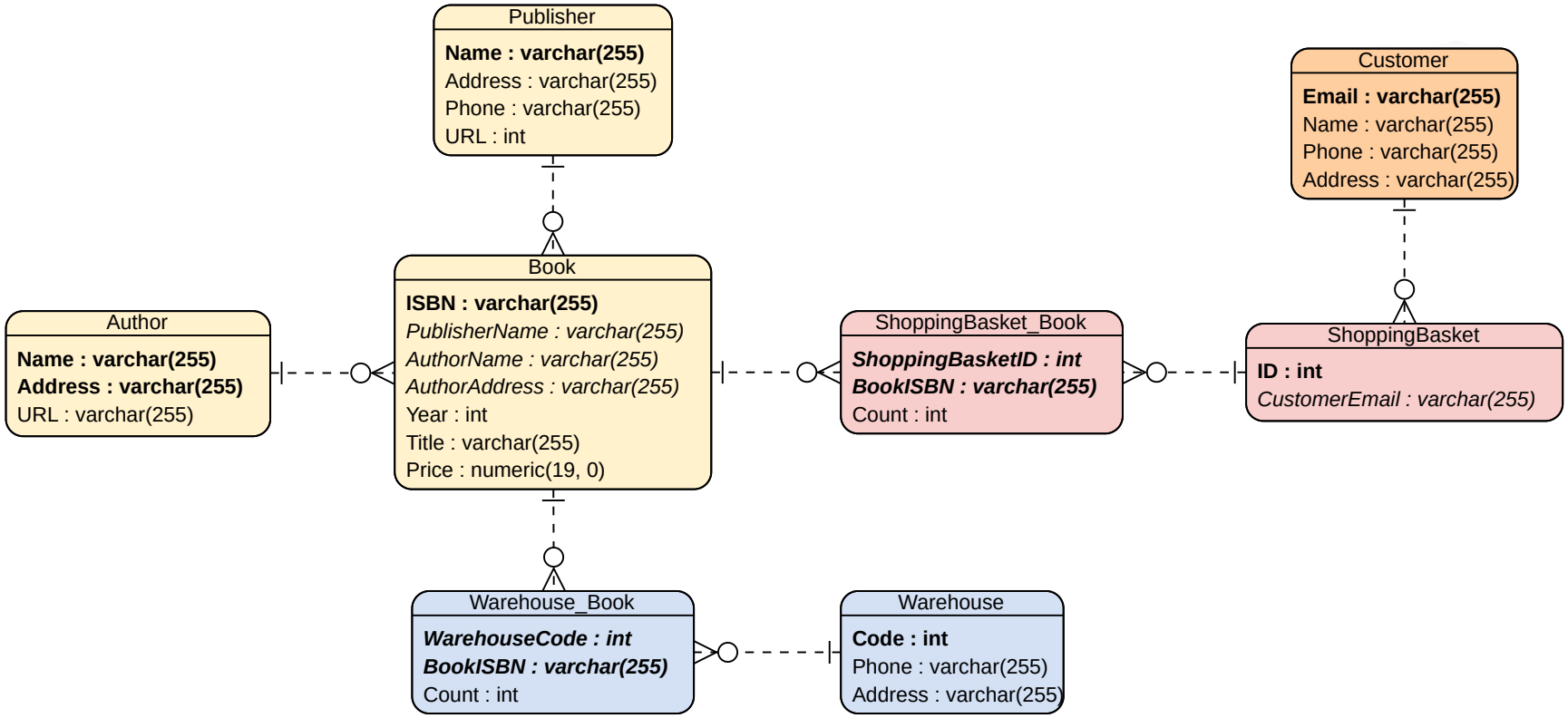
- Data integrity should be maintained to prevent loss of user or order information.


Book Store
User
Get Books By Category ID
«extend»
Get Book By Id
«extend»
Search Books
«extend»
Get All Books
Clear Cart
Create Cart
Cart
Get Cart By User ID
«extend»
Search Cart
«extend»
Get All Carts
User
Patch Cart SubTotal
Patch Cart Quantity
Cart Item
Delete CartItem By Id
Create CartItem
Get CartItem By Username
«extend»
Create Orders
«include»
Search Cart Item
«extend»
Get CartItem By Id
«extend»
Create OrderDetails
Get All CartItems
Admin
Author
Search Author
Add Book
Delete Author
Admin
Add Category
Update Author
Add Subcategory
Add Author


Book Store
-books: List<Book>
-customers: List<Customer>
+addBook(book: Book): void
+removeBook(book: Book): void
+addCustomer(customer: Customer): void
+removeCustomer(customer: Customer): void
+searchBooks(keyword: String): List<Book>
{ Id > 0 }
{ Name <> Null }
Admin
-Id: int
-Name: String
+Add book(): void
+Add category(): void
+Add Subcategory(): void
+Search Author(): void
+Delete Author(): void
+Update Author(): void
+Add Author(): void
{book > 0 and customer >0}
+has a
+has a
user
-id: int
-name: String
-email: String
-address: String
+getId(): int
+setId(id: int): void
+getName(): String
+setName(name: String): void
+getEmail(): String
+setEmail(email: String): void
+getAddress(): String
+setAddress(address: String): void
{Id > 0}
Book
-title: String
-author: String
-price: double
-quantityInStock: int
-IdBook: int
+getTitle(): String
+setTitle(title: String): void
+getAuthor(): String
+setAuthor(author: String): void
+getPrice(): double
+setPrice(price: double): void
+getIdBook(): int
+setIdBook(Id: int): void
{author <> null}
+has a
user search for book details
user add order
+has a
+has a
Cart
-ID_Cart: int
+ClearCart(): void
+CreateCart(): void
+SearchCart(): void
{create_cart <> null}
{order <> null}
Orders
-IdOrader: int
+Add order(): void
+user_id(): void
+Update order(): void
+total price(): void
{Update_order <> order}


browse
View Book
[purchase]
no
yes
product available
no
Visit FAQ
Item Not Found
available
View Details
Is a member?
yes
Login
no
Register
success
success
yes
Book Added to Cart
continue?
no
Payment
completed
Order Details Summary
Signout


Customer
ShoppingCart
ProcessOrder
OrderSummary
1: AddBook
2: BookDetails
3: ShippingDetails
4: PaymentDetails
5: DisplayOrderDetails
6: ConfirmOrder
7: Checkout
8: Logout


Publisher
Name : varchar(255)
Address : varchar(255)
Phone : varchar(255)
URL : int
Customer
Email : varchar(255)
Name : varchar(255)
Phone : varchar(255)
Address : varchar(255)
Book
ISBN : varchar(255)
PublisherName : varchar(255)
AuthorName : varchar(255)
AuthorAddress : varchar(255)
Year : int
Title : varchar(255)
Price : numeric(19, 0)
Author
Name : varchar(255)
Address : varchar(255)
URL : varchar(255)
ShoppingBasket_Book
ShoppingBasketID : int
BookISBN : varchar(255)
Count : int
ShoppingBasket
ID : int
CustomerEmail : varchar(255)
Warehouse_Book
WarehouseCode : int
BookISBN : varchar(255)
Count : int
Warehouse
Code : int
Phone : varchar(255)
Address : varchar(255)