NADA GAMAL

Quantum Bookstore Design

JULY 8, 2025

FAWRY RISE JOURNEY

Introduction

The goal of this project is to design an extensible online bookstore system that supports various book types and allows common operations such as adding, buying, and removing books from inventory. The system was designed using object-oriented principles and design patterns to promote clean separation of concerns and future scalability.

Design Overview

The system is centered around an abstract class `Book`, from which all specific book types inherit. The store supports three book types:

- **PaperBook** – can be bought, shipped, and has stock
- **EBook** – can be bought and sent via email
- **DemoBook** – not for sale, read-only

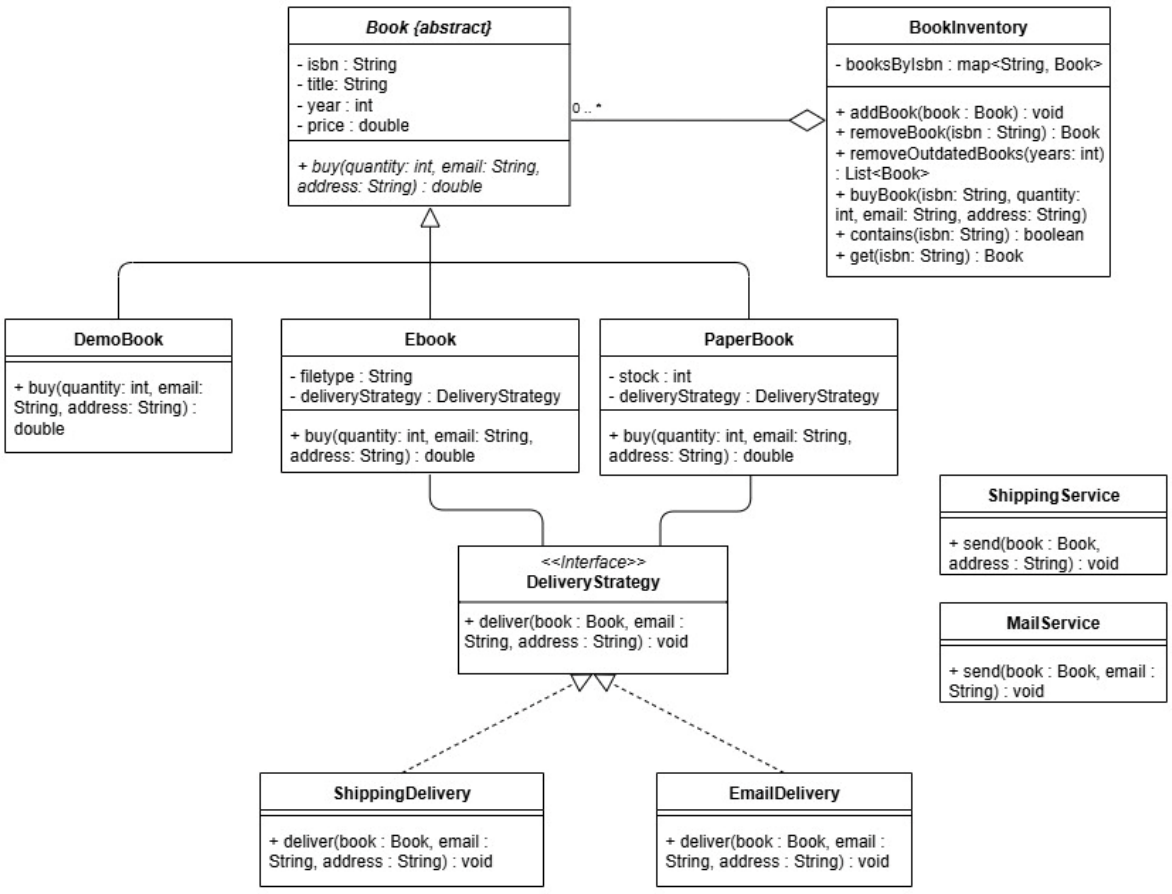
Books are stored in a centralized `BookInventory` class, which manages adding books, removing outdated ones, and processing purchases. Behavior of delivering a purchased book (e.g. shipping or emailing) is delegated to strategy classes via the `DeliveryStrategy` interface. The design emphasizes extensibility – new types of books or delivery methods can be added with minimal changes.

Class Descriptions

Class	Responsibility
`Book`	Base class for all books. Holds shared fields(ISBN, title, year, price) and defines the abstract `buy(…)` method
`PaperBook`	Represents a physical book. Tracks stock and uses a delivery strategy to ship upon purchase.
`EBook`	Represents a digital book. Has a file type and uses email delivery strategy to send upon purchase.
`DemoBook`	Represents a non-purchasable showcase book. Overrides `buy(…)` to prevent purchase.
`BookInventory`	Manages all books. Provides methods to add, remove, and buy books. Uses a map by ISBN.
`DeliveryStrategy`	Defines the contract for delivery behavior. Allows injecting different delivery methods.
`ShippingDelivery`	Implements delivery strategy for physical delivery.
`EmailDelivery`	Implements delivery strategy for digital delivery.

`MailService`	Static utility class that simulates sending an eBook via email
`ShippingService`	Static utility class that simulates shipping a paper book to a given address.
`BookstoreTest`	Test class to demonstrate adding, removing, and purchasing books. Validates correct behaviors and edge cases.

UML Class Diagram



Sample Output

```
--- Removing outdated books (20 years) ---
Quantum Book Store: Removed - Object-Oriented Programming

--- Buying available books ---
Quantum Book Store: Total paid for Paperbook: 91.98
Quantum Book Store: Total paid for EBook: 30.0

--- Trying to buy a demo book ---
This demo book is not for sale.

--- Buying more than available stock ---
Not enough stock available.

--- Buying a book not in inventory ---
Book with ISBN ID999 not found.
```

Conclusion

This design fulfills the requirements of a flexible and extensible online bookstore. It separates responsibilities clearly, uses polymorphism to model book behavior, and delegates delivery logic using the strategy pattern. The system can be easily extended in the future by adding need book types or delivery methods without modifying existing logic.