Exploring System Architecture with Design Patterns and UML Diagrams

An in-depth analysis of setup, commands, assumptions, limitations, and future improvements

System Architecture Overview

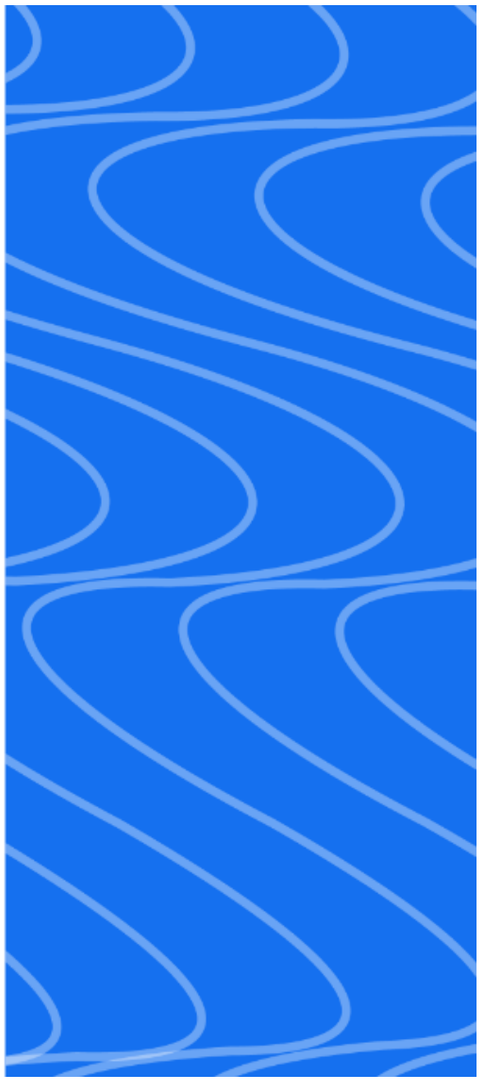
System Architecture Overview

The Library Management System (LMS) is structured around the Model-View-Controller (MVC) pattern.

Key components include the Model for managing library data, the Controller for user interactions, and the absence of a direct View due to the command-line interface.

Design patterns like Singleton, Factory Method, Decorator, Observer, Command, and Facade enhance system functionality.

The system aims for efficient book management, user notifications, and extensible features for adding new item types.



Design Patterns in Action

01

Singleton Pattern

Ensures only one instance of the library exists for consistent data management.

02

Factory Method Pattern

Facilitates the creation of new library items like books and magazines without altering core code.

03

Decorator Pattern

Dynamically enhances book properties, allowing for special collections like rare books.

04

Observer Pattern

Notifies users of new book availability, enhancing user experience.

05

Command Pattern

Enables flexible management of user actions like book checkout with undo functionality.

UML Diagrams for Patterns

Singleton Pattern UML

Illustrates the structure where only one instance of the library exists.

Factory Method Pattern UML

Demonstrates the creation of library items like books and magazines using a factory method.

Decorator Pattern UML

Shows how the RareBookDecorator dynamically adds "rare" status to books.

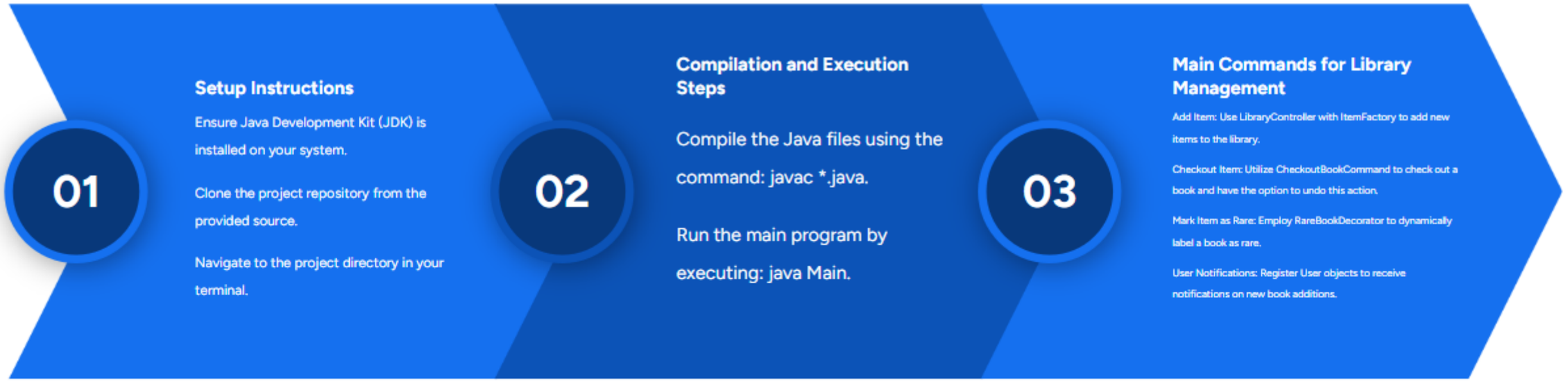
Observer Pattern UML

Displays the relationship between User and BookAvailabilityNotifier for user notifications.

Command Pattern UML

Represents the CheckoutBookCommand for encapsulating checkout actions.

Setup and Commands



Assumptions and Limitations

01

Single Library Instance

The system operates under the assumption of a single library instance for all items.

02

Simplified Interface

The absence of a graphical user interface (GUI) simplifies the system's interaction design.

03

Limited Item Types

Currently, the system only supports Book and Magazine item types, with potential for easy expansion using the ItemFactory.

04

No Database Integration

Data is stored in memory, leading to data loss upon application closure.

05

Command-Line Interface

The main class functions as a command-line interface, limiting user interaction possibilities.

Conclusion and Future Improvements

01

Conclusion

The Library Management System project effectively showcases the implementation of various design patterns, enhancing code flexibility and maintainability.

02

Future Improvements

Integration of a database to persist data beyond application sessions.

Expansion of item types beyond books and magazines to enrich the library's offerings.

Development of a graphical user interface (GUI) for a more user-friendly experience.

Design Patterns in Action

01

Singleton Pattern

Ensures only one instance of the library exists for consistent data management.

02

Factory Method Pattern

Facilitates the creation of new library items like books and magazines without altering core code.

03

Decorator Pattern

Dynamically enhances book properties, allowing for special collections like rare books.

04

Observer Pattern

Notifies users of new book availability, enhancing user experience.

05

Command Pattern

Enables flexible management of user actions like book checkout with undo functionality.