Library Management System Documentation

User Documentation

Overview

The Library Management System is a software application designed to automate and manage the operations of a library. It provides functionalities for adding, editing, and removing books, authors, and patrons. Users can search for books by title, author, or ISBN, borrow and return books, and manage patron records.

Class Descriptions

- **LibraryCLI**: This class provides a command-line interface for interacting with the library system.

- **Library**: Represents the library itself and contains methods for managing books, authors, and patrons.

- **Book**: Represents a book in the library system, including its title, author, ISBN, publisher, and number of copies.

- **Author**: Represents an author of a book, including their name and date of birth.

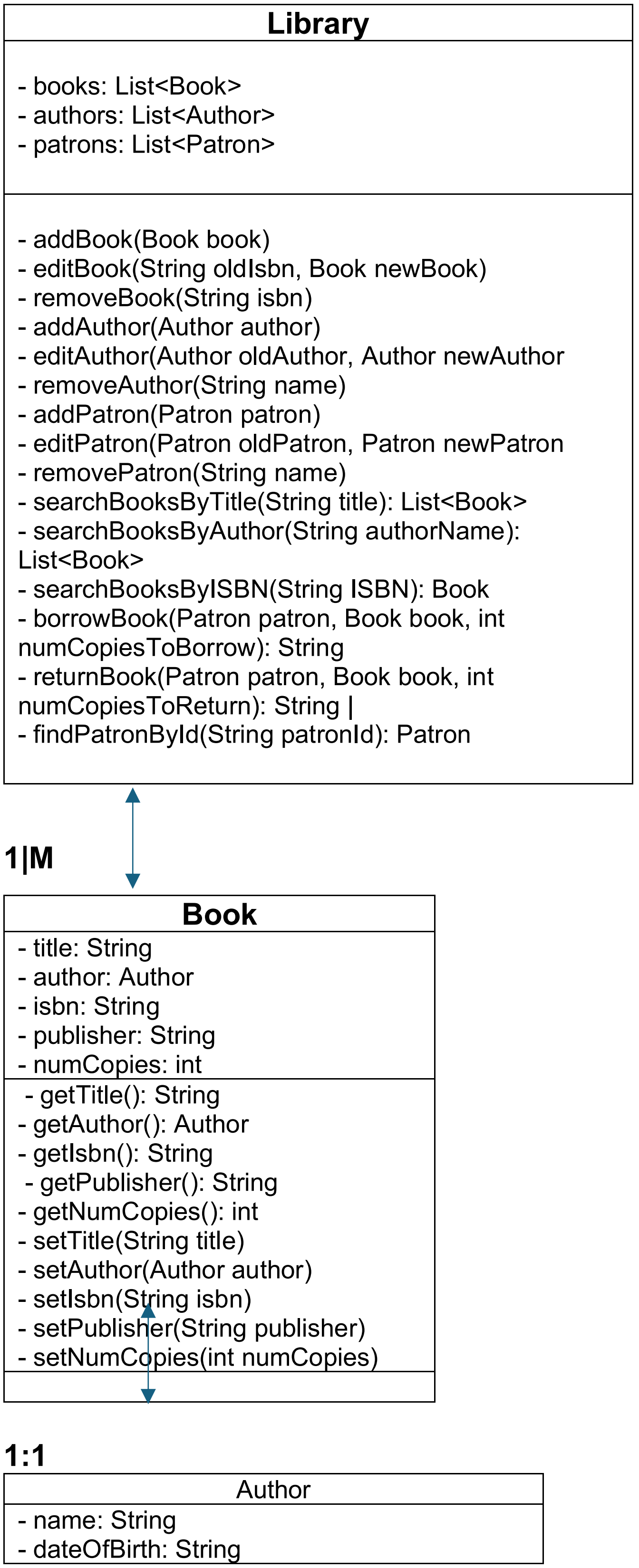
- **Patron**: Represents a library patron, including their name, address, and phone number.

Getting Started

To access the Library Management System:

- Open the command-line interface.
- Compile and run the **Main.java** file – or type java Main and hit enter.
- Follow the instructions displayed in the console to interact with the system.

Class Diagram



Library

- books: List<Book>
- authors: List<Author>
- patrons: List<Patron>

- addBook(Book book)
- editBook(String oldIsbn, Book newBook)
- removeBook(String isbn)
- addAuthor(Author author)
- editAuthor(Author oldAuthor, Author newAuthor
- removeAuthor(String name)
- addPatron(Patron patron)
- editPatron(Patron oldPatron, Patron newPatron
- removePatron(String name)
- searchBooksByTitle(String title): List<Book>
- searchBooksByAuthor(String authorName): List<Book>
- searchBooksByISBN(String ISBN): Book
- borrowBook(Patron patron, Book book, int numCopiesToBorrow): String
- returnBook(Patron patron, Book book, int numCopiesToReturn): String |
- findPatronById(String patronId): Patron

1|M

Book

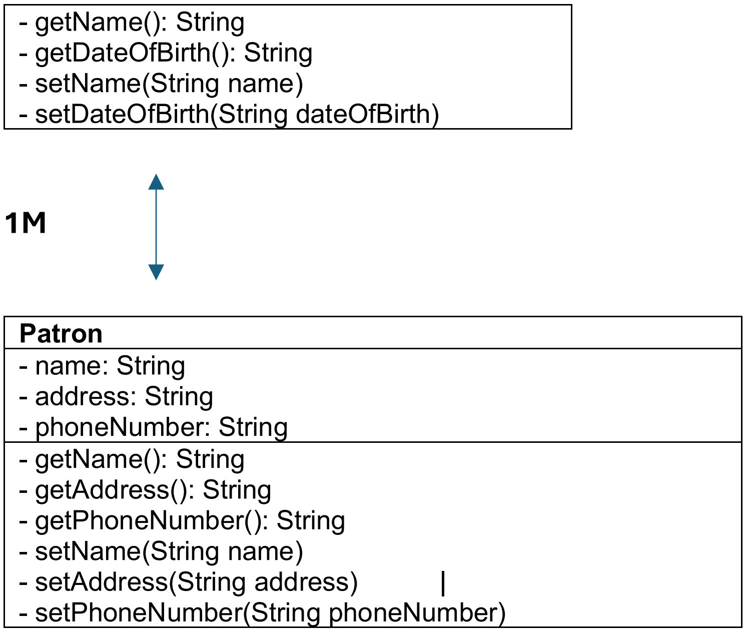
- title: String
- author: Author
- isbn: String
- publisher: String
- numCopies: int

- getTitle(): String
- getAuthor(): Author
- getIsbn(): String
- getPublisher(): String
- getNumCopies(): int
- setTitle(String title)
- setAuthor(Author author)
- setIsbn(String isbn)
- setPublisher(String publisher)
- setNumCopies(int numCopies)

1:1

Author

- name: String
- dateOfBirth: String



- getName(): String
- getDateOfBirth(): String
- setName(String name)
- setDateOfBirth(String dateOfBirth)

1M

Patron

- name: String
- address: String
- phoneNumber: String

- getName(): String
- getAddress(): String
- getPhoneNumber(): String
- setName(String name)
- setAddress(String address) |
- setPhoneNumber(String phoneNumber)

Development Documentation

Source Code Structure

The source code directory structure is organized as follows:

```
Library-Management-System/
        |— Author.java
        |— Book.java
        |— Borrowable.java
        |— Library.java
        |— LibraryCLI.java
        |— Library.java
        |— LibraryCLI.java
        |— Main.java
        |— Patron.java
```

### Build Process
- Compile the project using any Java compiler, e.g., `javac`.
- Run the project by executing the compiled `Main.class` file or type java Main in the terminal.

### Compiler Time Dependencies
- The project has no external dependencies beyond the Java standard library.

### Development Standards
- Followed Java coding standards and best practices.
- Use of meaningful variable and method names.
- Provided comments and documentation for all classes and methods.

### Source Code Repository
- The source code repository is hosted on [https://github.com/janeilchantelle/Library-Management-System.git).
- You can clone the repository using git clone https://github.com/janeilchantelle/Library-Management-System.git.

## Deployment Documentation

### Installation Manual
To deploy the Library Management System:
1. Download the project source code from the repository.
2. Compile the source code using a Java compiler.
3. Run the compiled `Main.class` file to start the application.

### System Requirements
- Java Development Kit (JDK) installed on the system.

### Configuration Settings
- No specific configuration settings are required for deployment.

### Deployment Checklist
- Ensure the JDK is installed and configured correctly.
- Verify that the project compiles without errors.
- Run the application and test its functionality.

### Troubleshooting
- If you encounter any issues during deployment, refer to the development documentation or seek assistance from the project team.