Library Management System

Lecturer:

D4017_Jude Joseph Lamug Martinez, MCS

Arranged by :

2702350775 Suwandi The

Object Oriented Programming

Computer Science Faculty

Binus International University

Table Of Contents

Abstract

This report is built to document my Object Oriented Programming final project about making a Library Management System in Java. The purposes of this program are to allow the library administrators to manage book inventories, while users can borrow, return, purchase, and view their borrowed books.

Chapter 1 - Project Specification

1.1 Project title

Project title: Library Management System

1.2 Project Description

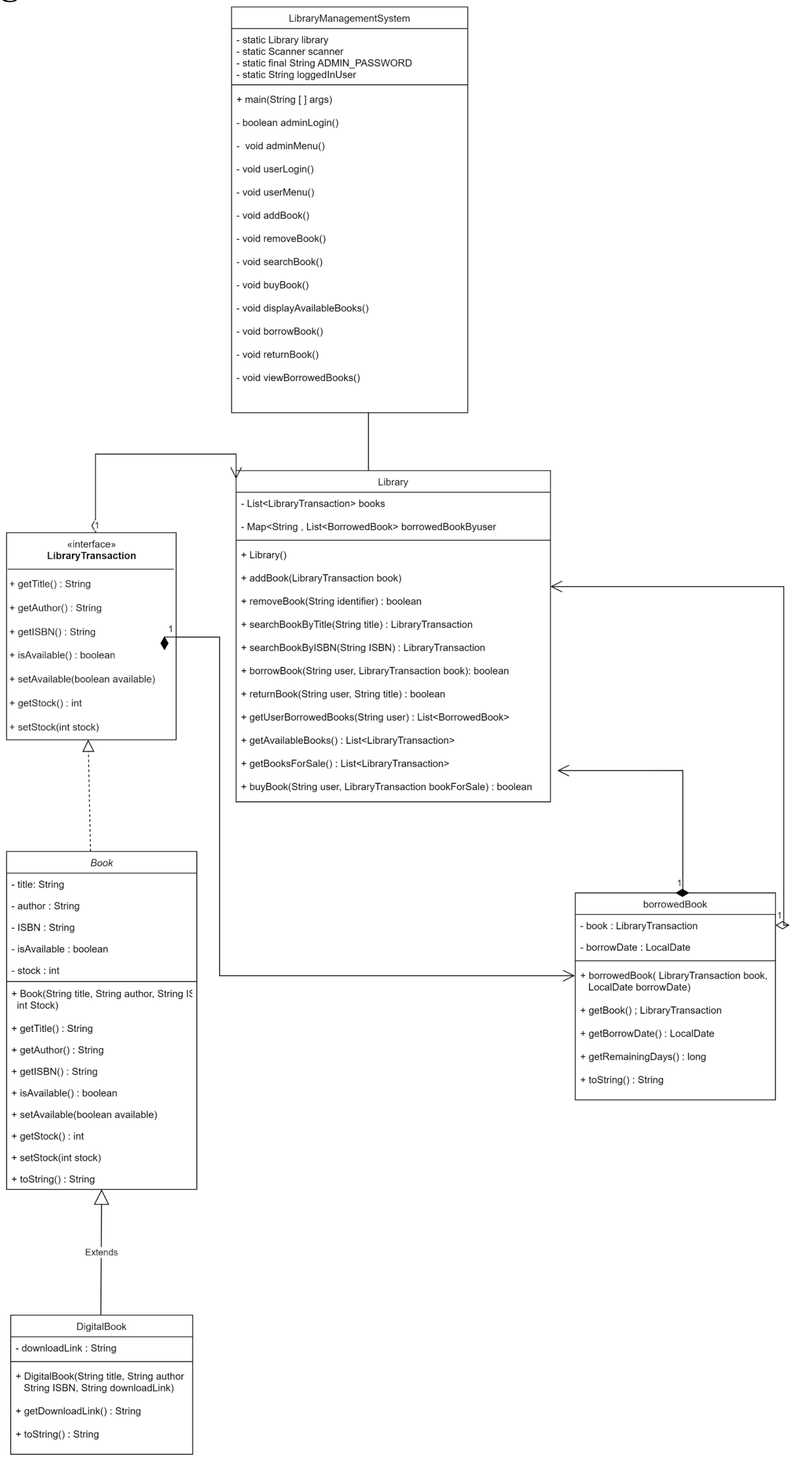
This Library Management System is an object-oriented program designed to streamline and manage the library operations, incorporating the functionalities for handling both physical and digital books. The system distinguished between administrators and users roles, offering specialized features for each.The system allows the administrators to manage the library inventories, while the users can borrow, return, purchase, and view their borrowed books. The key functionalities of this system include:

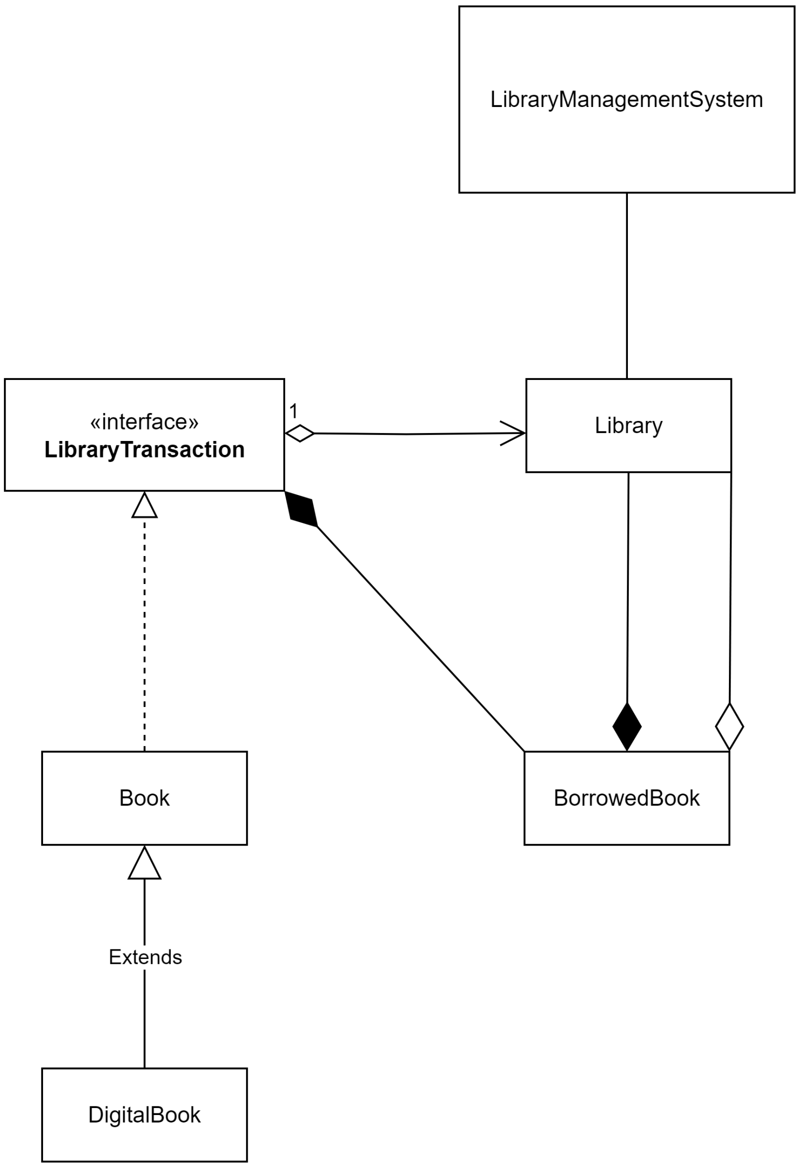
- **Administrators login for managing the library inventory**
    - Adding both physical and digital books
    - Searching books by title or ISBN
    - Removing books by title or ISBN

- **Users login for performing library transactions**
    - Purchase books by title or ISBN
    - Borrow books by title or ISBN
    - Return books by title or ISBN
    - View borrowed books.

Chapter 2. Solution Design

1.1 Class diagram


LibraryManagementSystem
- static Library library
- static Scanner scanner
- static final String ADMIN_PASSWORD
- static String loggedInUser
+ main(String [ ] args)
- boolean adminLogin()
- void adminMenu()
- void userLogin()
- void userMenu()
- void addBook()
- void removeBook()
- void searchBook()
- void buyBook()
- void displayAvailableBooks()
- void borrowBook()
- void returnBook()
- void viewBorrowedBooks()
Library
- List<LibraryTransaction> books
- Map<String , List<BorrowedBook> borrowedBookByuser
+ Library()
+ addBook(LibraryTransaction book)
+ removeBook(String identifier) : boolean
+ searchBookByTitle(String title) : LibraryTransaction
+ searchBookByISBN(String ISBN) : LibraryTransaction
+ borrowBook(String user, LibraryTransaction book): boolean
+ returnBook(String user, String title) : boolean
+ getUserBorrowedBooks(String user) : List<BorrowedBook>
+ getAvailableBooks() : List<LibraryTransaction>
+ getBooksForSale() : List<LibraryTransaction>
+ buyBook(String user, LibraryTransaction bookForSale) : boolean
1
«interface»
LibraryTransaction
+ getTitle() : String
+ getAuthor() : String
+ getISBN() : String
+ isAvailable() : boolean
+ setAvailable(boolean available)
+ getStock() : int
+ setStock(int stock)
1
Book
- title: String
- author : String
- ISBN : String
- isAvailable : boolean
- stock : int
+ Book(String title, String author, String IS
int Stock)
+ getTitle() : String
+ getAuthor() : String
+ getISBN() : String
+ isAvailable() : boolean
+ setAvailable(boolean available)
+ getStock() : int
+ setStock(int stock)
+ toString() : String
Extends
DigitalBook
- downloadLink : String
+ DigitalBook(String title, String author
String ISBN, String downloadLink)
+ getDownloadLink() : String
+ toString() : String
1
1
borrowedBook
- book : LibraryTransaction
- borrowDate : LocalDate
+ borrowedBook(LibraryTransaction book,
LocalDate borrowDate)
+ getBook() ; LibraryTransaction
+ getBorrowDate() : LocalDate
+ getRemainingDays() : long
+ toString() : String

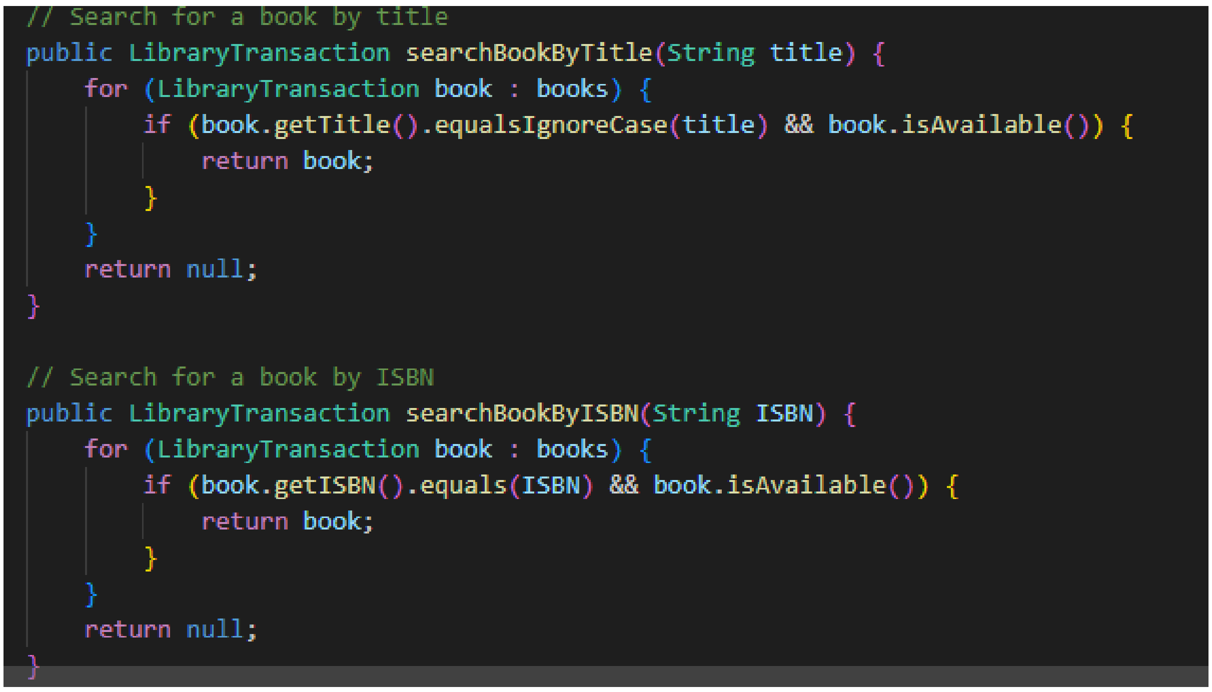

LibraryManagementSystem
«interface»
LibraryTransaction
1
Library
Book
BorrowedBook
Extends
DigitalBook

1.2 Implementation and how it works

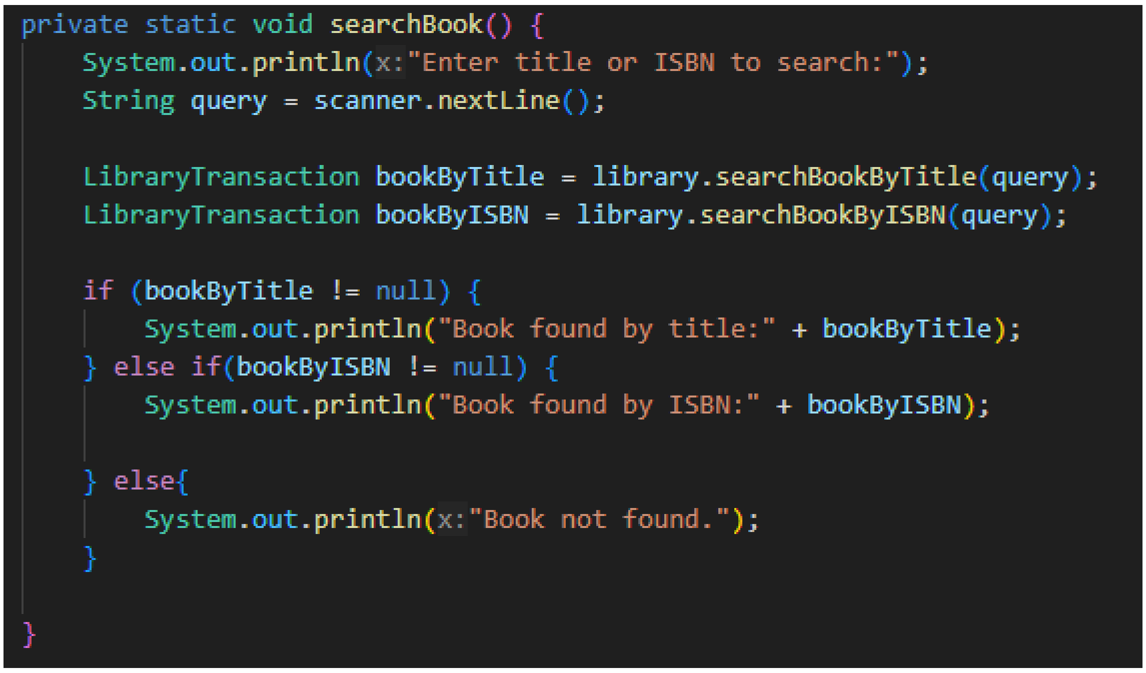
1.Algorithms:

Searching algorithm : the system uses a linear search algorithm.

complexity: O(n) , where n is the number of books.



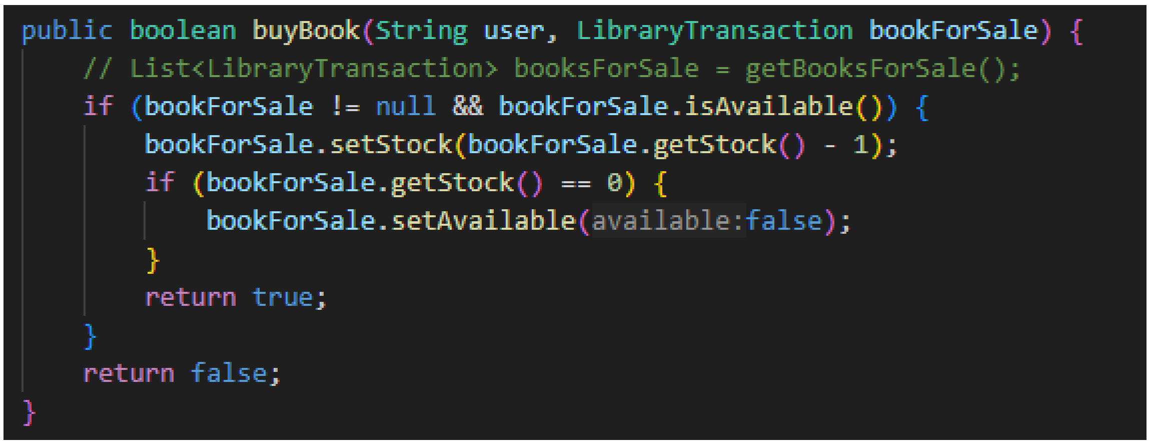
```java
// Search for a book by title
public LibraryTransaction searchBookByTitle(String title) {
    for (LibraryTransaction book : books) {
        if (book.getTitle().equalsIgnoreCase(title) && book.isAvailable()) {
            return book;
        }
    }
    return null;
}

// Search for a book by ISBN
public LibraryTransaction searchBookByISBN(String ISBN) {
    for (LibraryTransaction book : books) {
        if (book.getISBN().equals(ISBN) && book.isAvailable()) {
            return book;
        }
    }
    return null;
}
```



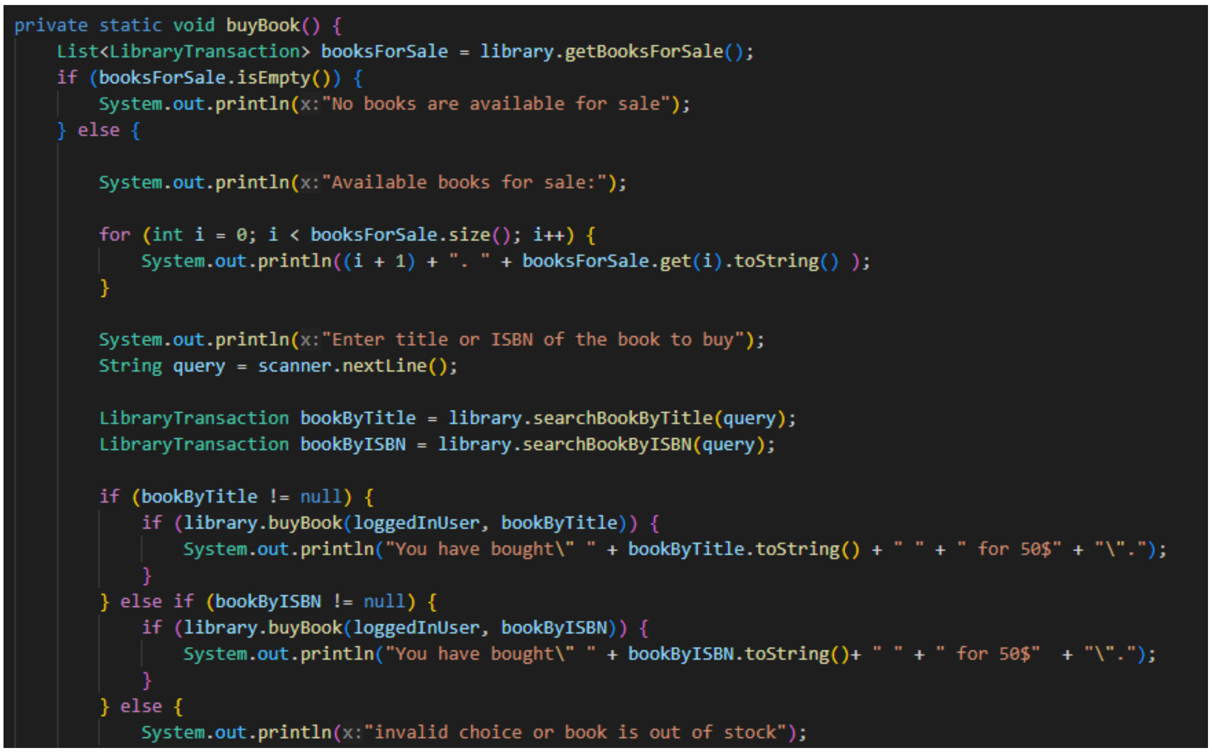
```java
private static void searchBook() {
    System.out.println(x:"Enter title or ISBN to search:");
    String query = scanner.nextLine();

    LibraryTransaction bookByTitle = library.searchBookByTitle(query);
    LibraryTransaction bookByISBN = library.searchBookByISBN(query);

    if (bookByTitle != null) {
        System.out.println("Book found by title:" + bookByTitle);
    } else if(bookByISBN != null) {
        System.out.println("Book found by ISBN:" + bookByISBN);

    } else{
        System.out.println(x:"Book not found.");
    }

}
```

Buying a book algorithm:

- **firstly, the system search for the book by title or ISBN**
- **the system check the availability**
- **decrement the stock is available**
- **Complexity : O(n) for searching, O(1) for buying**



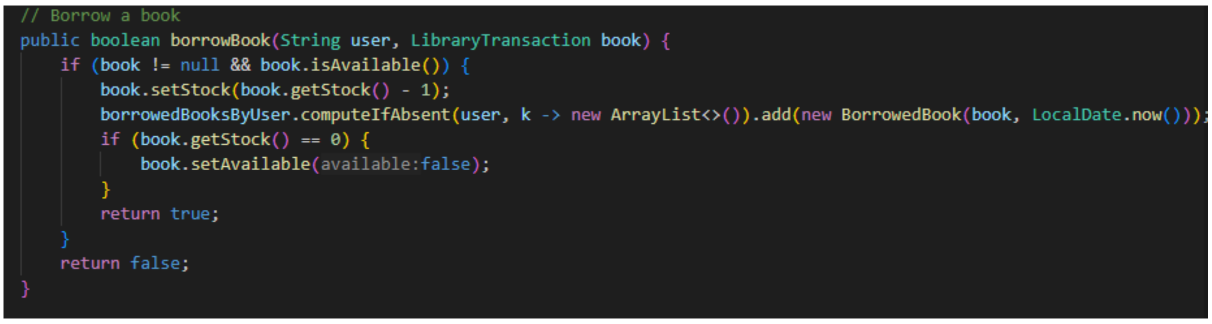
```java
public boolean buyBook(String user, LibraryTransaction bookForSale) {
    // List<LibraryTransaction> booksForSale = getBooksForSale();
    if (bookForSale != null && bookForSale.isAvailable()) {
        bookForSale.setStock(bookForSale.getStock() - 1);
        if (bookForSale.getStock() == 0) {
            bookForSale.setAvailable(available:false);
        }
        return true;
    }
    return false;
}
```



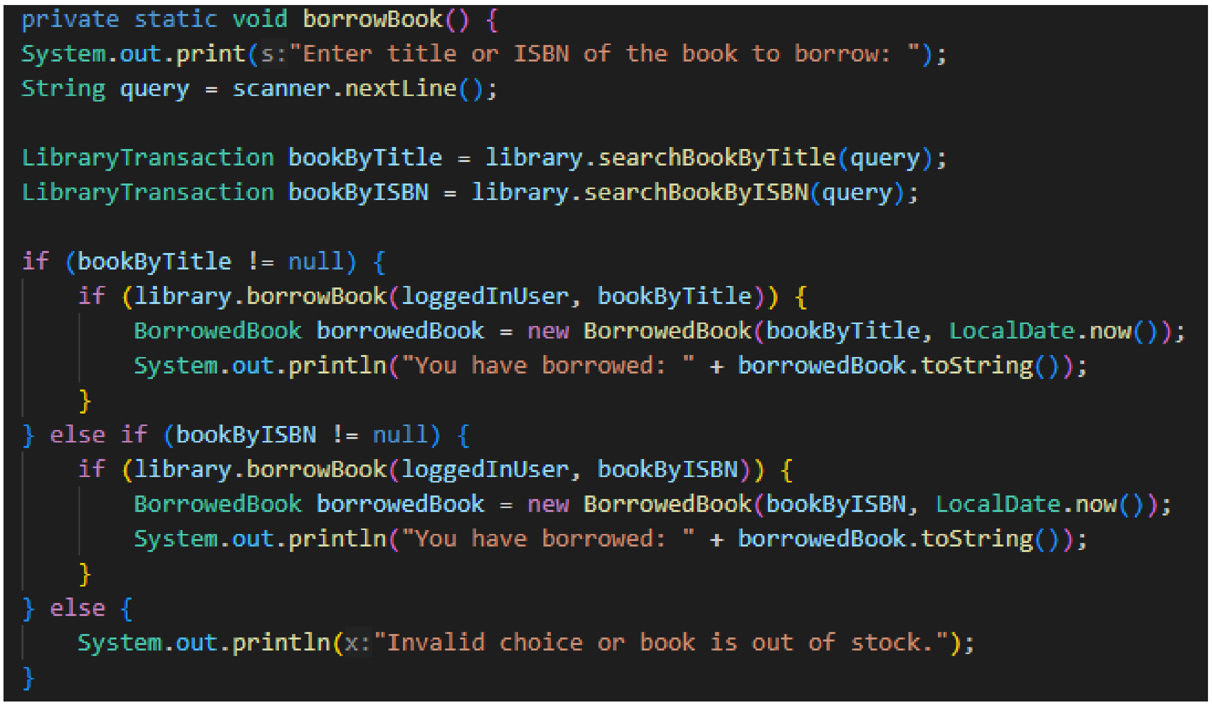
```java
private static void buyBook() {
    List<LibraryTransaction> booksForSale = library.getBooksForSale();
    if (booksForSale.isEmpty()) {
        System.out.println(x:"No books are available for sale");
    } else {

        System.out.println(x:"Available books for sale:");

        for (int i = 0; i < booksForSale.size(); i++) {
            System.out.println((i + 1) + ". " + booksForSale.get(i).toString() );
        }

        System.out.println(x:"Enter title or ISBN of the book to buy");
        String query = scanner.nextLine();

        LibraryTransaction bookByTitle = library.searchBookByTitle(query);
        LibraryTransaction bookByISBN = library.searchBookByISBN(query);

        if (bookByTitle != null) {
            if (library.buyBook(loggedInUser, bookByTitle)) {
                System.out.println("You have bought\" " + bookByTitle.toString() + " " + " for 50$" + "\".");
            }
        } else if (bookByISBN != null) {
            if (library.buyBook(loggedInUser, bookByISBN)) {
                System.out.println("You have bought\" " + bookByISBN.toString()+ " " + " for 50$"  + "\".");
            }
        } else {
            System.out.println(x:"invalid choice or book is out of stock");
```

Borrowing algorithm:

1. **The system search for a book by title or ISBN**
2. **The system check for the availability of the book**
3. **Decrement the stock if available**
4. **add the book to the users' borrowed book list**



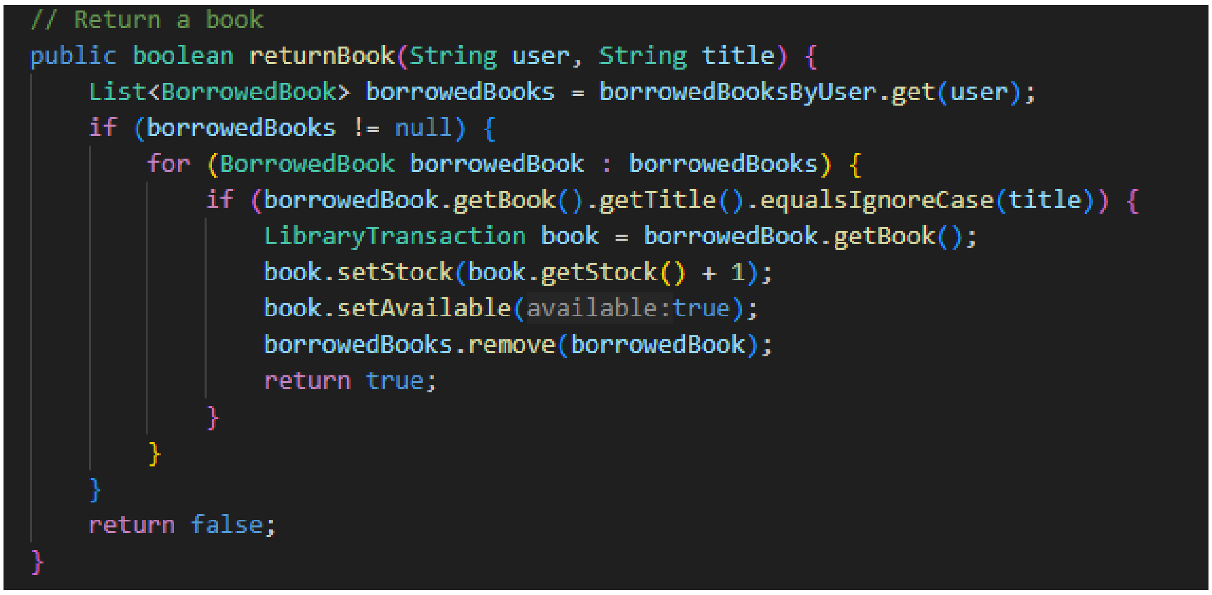
```java
// Borrow a book
public boolean borrowBook(String user, LibraryTransaction book) {
    if (book != null && book.isAvailable()) {
        book.setStock(book.getStock() - 1);
        borrowedBooksByUser.computeIfAbsent(user, k -> new ArrayList<>()).add(new BorrowedBook(book, LocalDate.now()));
        if (book.getStock() == 0) {
            book.setAvailable(available:false);
        }
        return true;
    }
    return false;
}
```



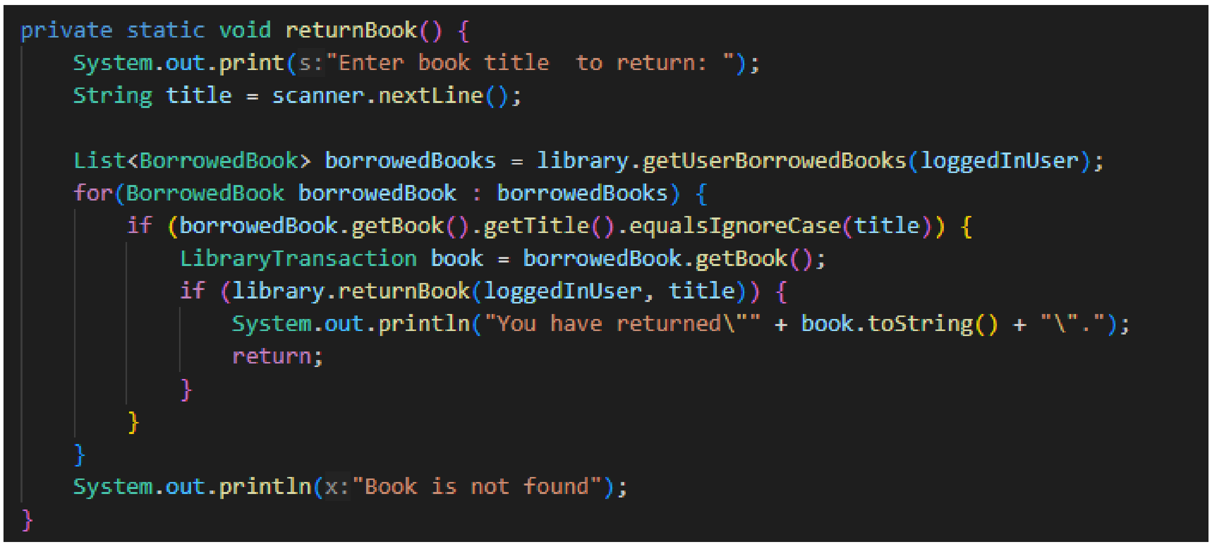
```java
private static void borrowBook() {
System.out.print(s:"Enter title or ISBN of the book to borrow: ");
String query = scanner.nextLine();

LibraryTransaction bookByTitle = library.searchBookByTitle(query);
LibraryTransaction bookByISBN = library.searchBookByISBN(query);

if (bookByTitle != null) {
    if (library.borrowBook(loggedInUser, bookByTitle)) {
        BorrowedBook borrowedBook = new BorrowedBook(bookByTitle, LocalDate.now());
        System.out.println("You have borrowed: " + borrowedBook.toString());
    }
} else if (bookByISBN != null) {
    if (library.borrowBook(loggedInUser, bookByISBN)) {
        BorrowedBook borrowedBook = new BorrowedBook(bookByISBN, LocalDate.now());
        System.out.println("You have borrowed: " + borrowedBook.toString());
    }
} else {
    System.out.println(x:"Invalid choice or book is out of stock.");
}
```

Returning book algorithm:

1. **Locate the book in the users' borrowed books list**
2. **Increment the stock**
3. **remove the book from the list**



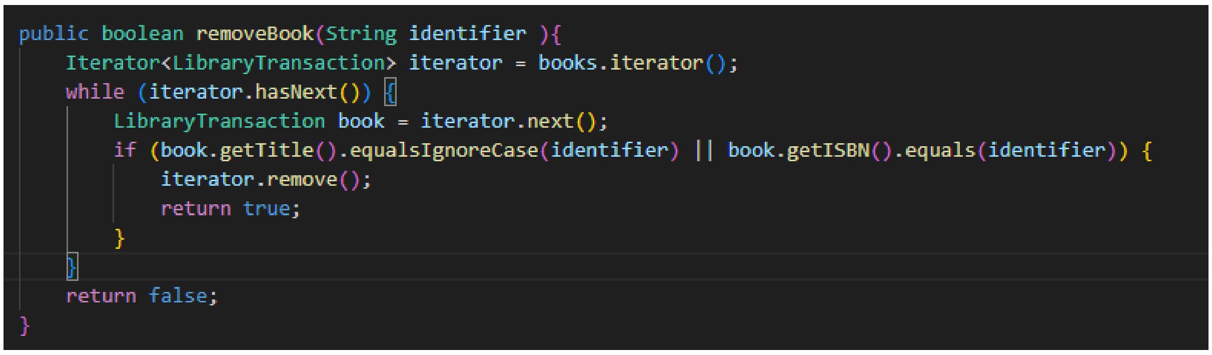
```java
// Return a book
public boolean returnBook(String user, String title) {
    List<BorrowedBook> borrowedBooks = borrowedBooksByUser.get(user);
    if (borrowedBooks != null) {
        for (BorrowedBook borrowedBook : borrowedBooks) {
            if (borrowedBook.getBook().getTitle().equalsIgnoreCase(title)) {
                LibraryTransaction book = borrowedBook.getBook();
                book.setStock(book.getStock() + 1);
                book.setAvailable(available:true);
                borrowedBooks.remove(borrowedBook);
                return true;
            }
        }
    }
    return false;
}
```



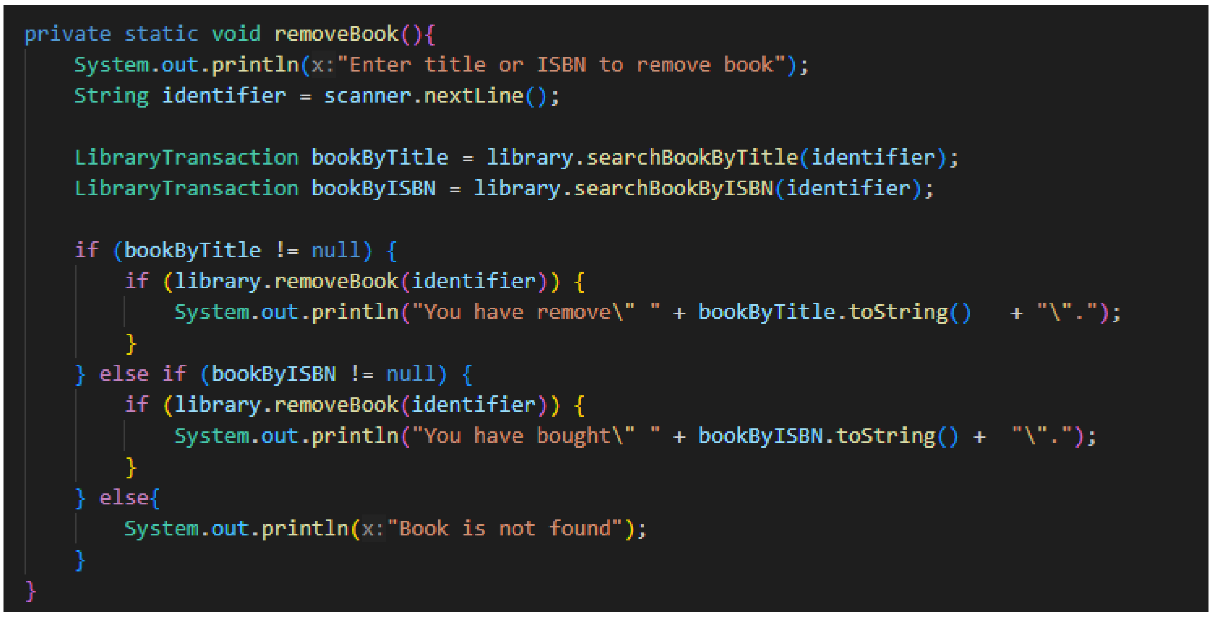
```java
private static void returnBook() {
    System.out.print(s:"Enter book title  to return: ");
    String title = scanner.nextLine();

    List<BorrowedBook> borrowedBooks = library.getUserBorrowedBooks(loggedInUser);
    for(BorrowedBook borrowedBook : borrowedBooks) {
        if (borrowedBook.getBook().getTitle().equalsIgnoreCase(title)) {
            LibraryTransaction book = borrowedBook.getBook();
            if (library.returnBook(loggedInUser, title)) {
                System.out.println("You have returned\"" + book.toString() + "\".");
                return;
            }
        }
    }
    System.out.println(x:"Book is not found");
}
```

Removing algorithm:

1. **The system search for the book by title or ISBN**
2. **The system check for the availability**
3. **Iterate over the list of book stored in the library**
4. **the system check for matching title or ISBN**
5. **if a match is found , the system will remove the book**



```java
public boolean removeBook(String identifier ){
    Iterator<LibraryTransaction> iterator = books.iterator();
    while (iterator.hasNext()) {
        LibraryTransaction book = iterator.next();
        if (book.getTitle().equalsIgnoreCase(identifier) || book.getISBN().equals(identifier)) {
            iterator.remove();
            return true;
        }
    }
    return false;
}
```



```java
private static void removeBook(){
    System.out.println(x:"Enter title or ISBN to remove book");
    String identifier = scanner.nextLine();

    LibraryTransaction bookByTitle = library.searchBookByTitle(identifier);
    LibraryTransaction bookByISBN = library.searchBookByISBN(identifier);

    if (bookByTitle != null) {
        if (library.removeBook(identifier)) {
            System.out.println("You have remove\" " + bookByTitle.toString()   + "\".");
        }
    } else if (bookByISBN != null) {
        if (library.removeBook(identifier)) {
            System.out.println("You have bought\" " + bookByISBN.toString() +  "\".");
        }
    } else{
        System.out.println(x:"Book is not found");
    }
}
```

2. Solution scheme:

Class structure:

Book Class: represents physical books in the library

DigitalBook Class : inherits from "Book" and represents the digital version of book with a download link

BorrowedBook Class: represents a record of a borrowed book, including the book details and the remaining days.

Library Class: manages the collection of books and the operations within the library

LibraryTransaction interface: defines a contract for any transaction involving a book

LibraryManagementSystem Class: provides the user interface and the interaction

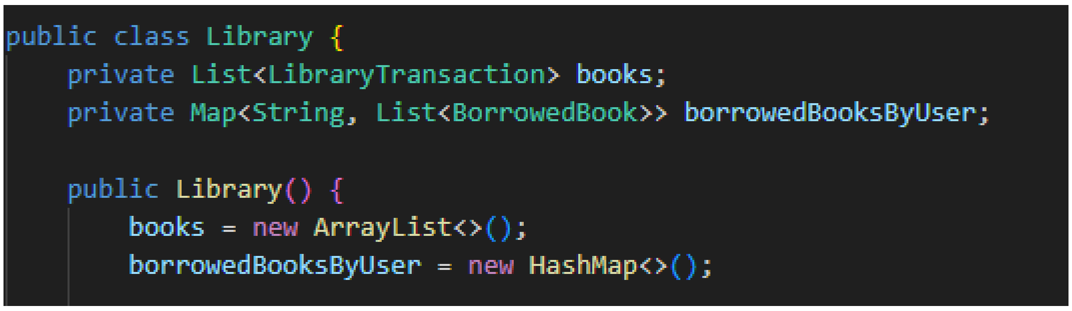
Data Structures used:

1. **ArrayList**

   Usage: to store the collection of books in the library

2. **HashMap**

   Usage: to map users to their list of borrowed books

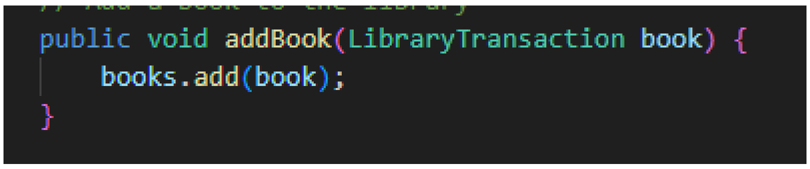


```
public class Library {
    private List<LibraryTransaction> books;
    private Map<String, List<BorrowedBook>> borrowedBooksByUser;

    public Library() {
        books = new ArrayList<>();
        borrowedBooksByUser = new HashMap<>();
```

Implementation:

1. **Adding a book**

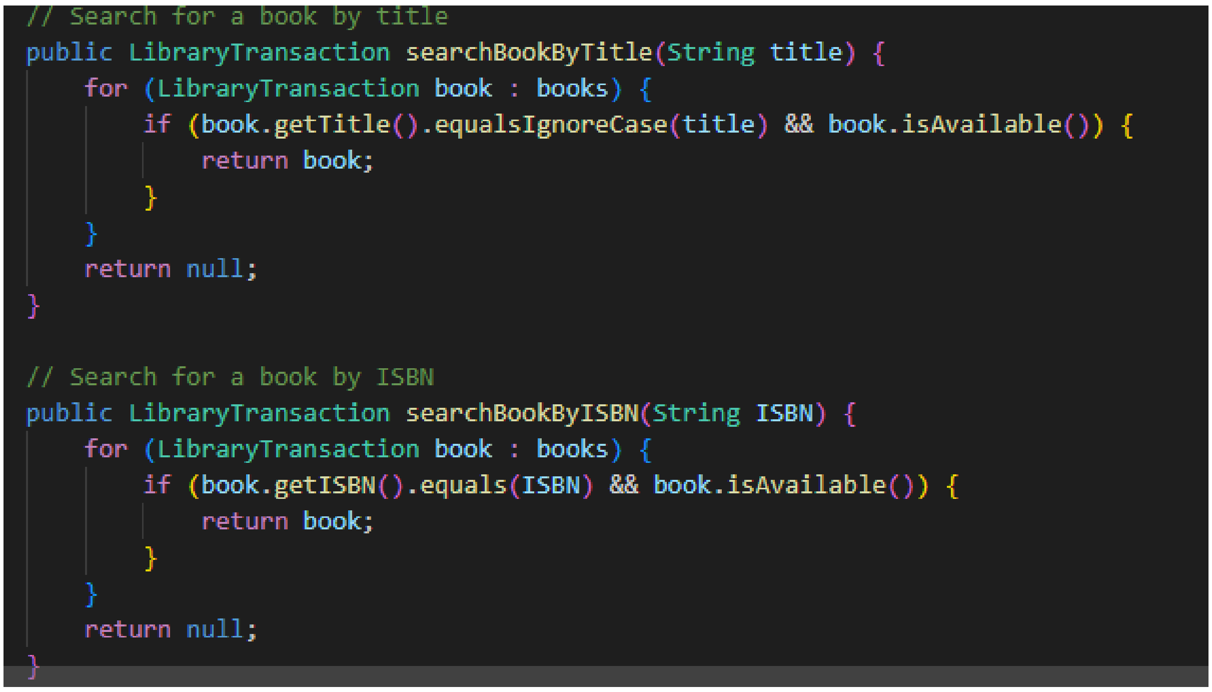
   add a book to the ArrayList of books



```java
public void addBook(LibraryTransaction book) {
    books.add(book);
}
```

2. **Searching for a book**

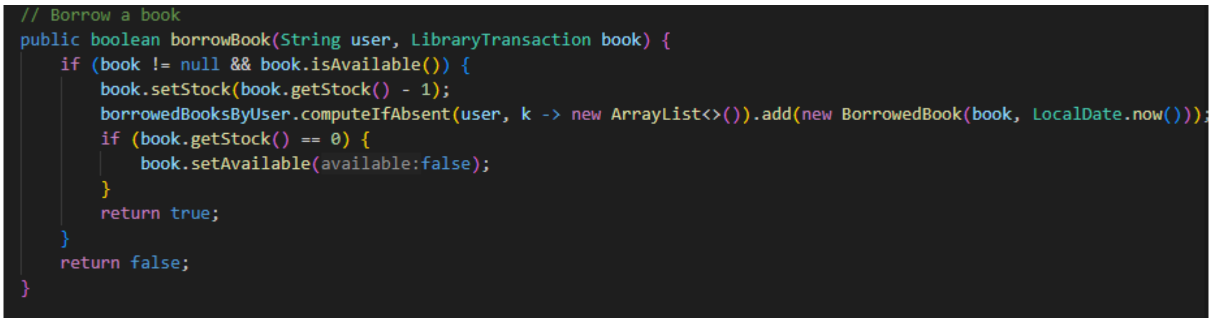
   The searching methods iterates over the ArrayList to find a book by title or ISBN



```java
// Search for a book by title
public LibraryTransaction searchBookByTitle(String title) {
    for (LibraryTransaction book : books) {
        if (book.getTitle().equalsIgnoreCase(title) && book.isAvailable()) {
            return book;
        }
    }
    return null;
}

// Search for a book by ISBN
public LibraryTransaction searchBookByISBN(String ISBN) {
    for (LibraryTransaction book : books) {
        if (book.getISBN().equals(ISBN) && book.isAvailable()) {
            return book;
        }
    }
    return null;
}
```

3. **Borrowing a book**

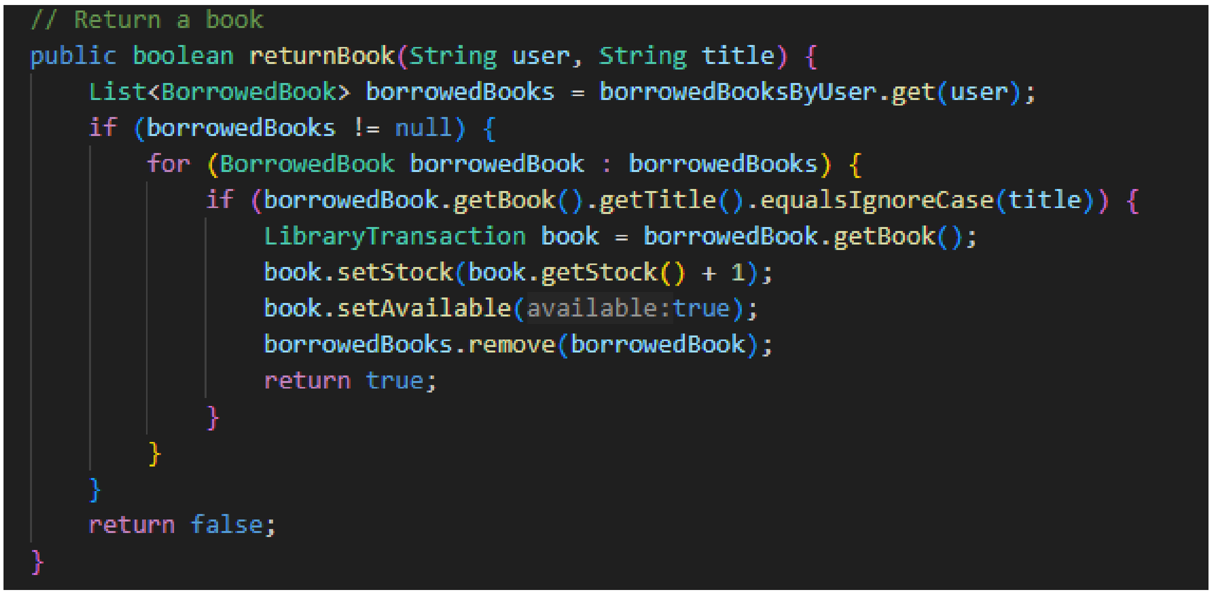
   The borrowBook method updates the stock and availability of the book and records the borrowed book for the user in the HashMap



```java
// Borrow a book
public boolean borrowBook(String user, LibraryTransaction book) {
    if (book != null && book.isAvailable()) {
        book.setStock(book.getStock() - 1);
        borrowedBooksByUser.computeIfAbsent(user, k -> new ArrayList<>()).add(new BorrowedBook(book, LocalDate.now()));
        if (book.getStock() == 0) {
            book.setAvailable(available:false);
        }
        return true;
    }
    return false;
}
```

4. **Returning a book**

   The returnBook method finds the borrowed books for the users' list , updates the stock and availability, and removes the book from the list



```java
// Return a book
public boolean returnBook(String user, String title) {
    List<BorrowedBook> borrowedBooks = borrowedBooksByUser.get(user);
    if (borrowedBooks != null) {
        for (BorrowedBook borrowedBook : borrowedBooks) {
            if (borrowedBook.getBook().getTitle().equalsIgnoreCase(title)) {
                LibraryTransaction book = borrowedBook.getBook();
                book.setStock(book.getStock() + 1);
                book.setAvailable(available:true);
                borrowedBooks.remove(borrowedBook);
                return true;
            }
        }
    }
    return false;
}
```

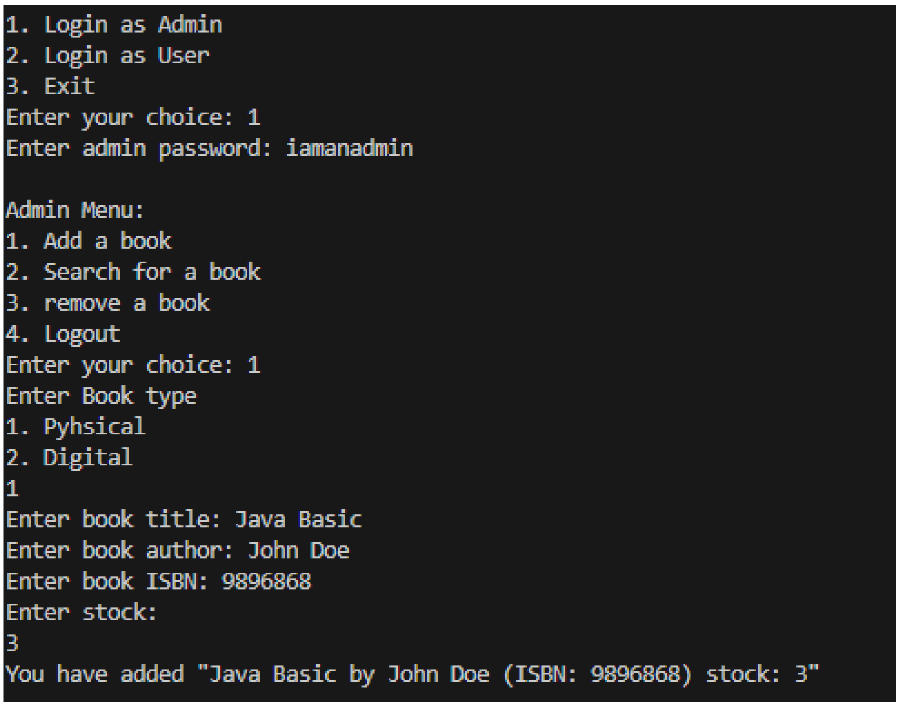
Chapter 3. Evidence of working program

1. **Admin**

   the system will prompt the user to input a password if they want to login as an admin.

   admin adding:

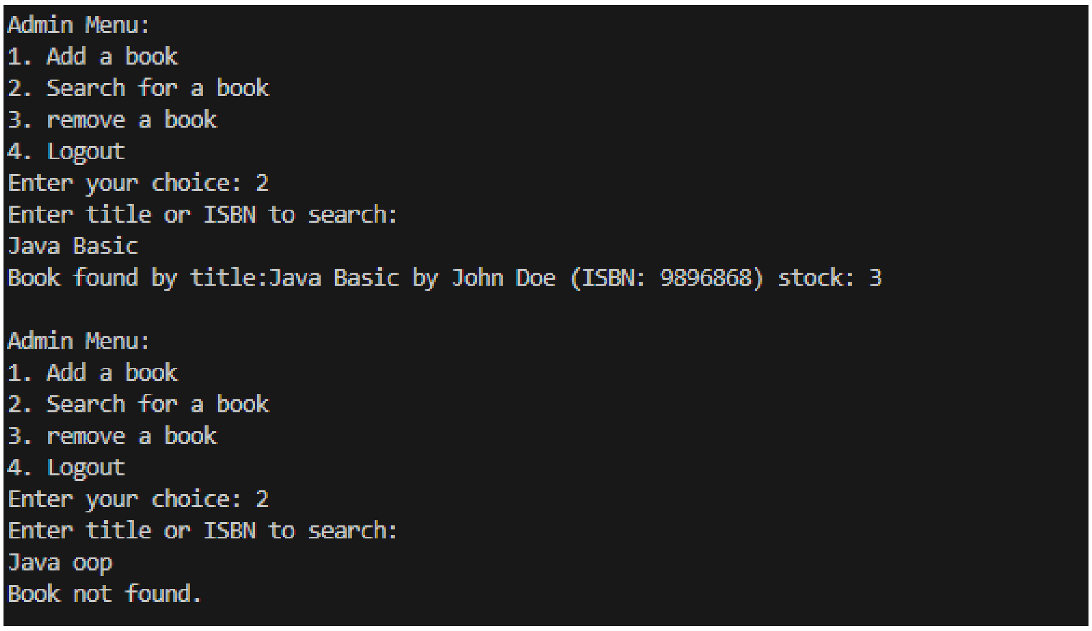
   The system will prompt the admin to choose whether they want to add physical or digital books, and ask for other details before adding the books into the database.



```
1. Login as Admin
2. Login as User
3. Exit
Enter your choice: 1
Enter admin password: iamanadmin

Admin Menu:
1. Add a book
2. Search for a book
3. remove a book
4. Logout
Enter your choice: 1
Enter Book type
1. Pyhsical
2. Digital
1
Enter book title: Java Basic
Enter book author: John Doe
Enter book ISBN: 9896868
Enter stock:
3
You have added "Java Basic by John Doe (ISBN: 9896868) stock: 3"
```

admin searching:

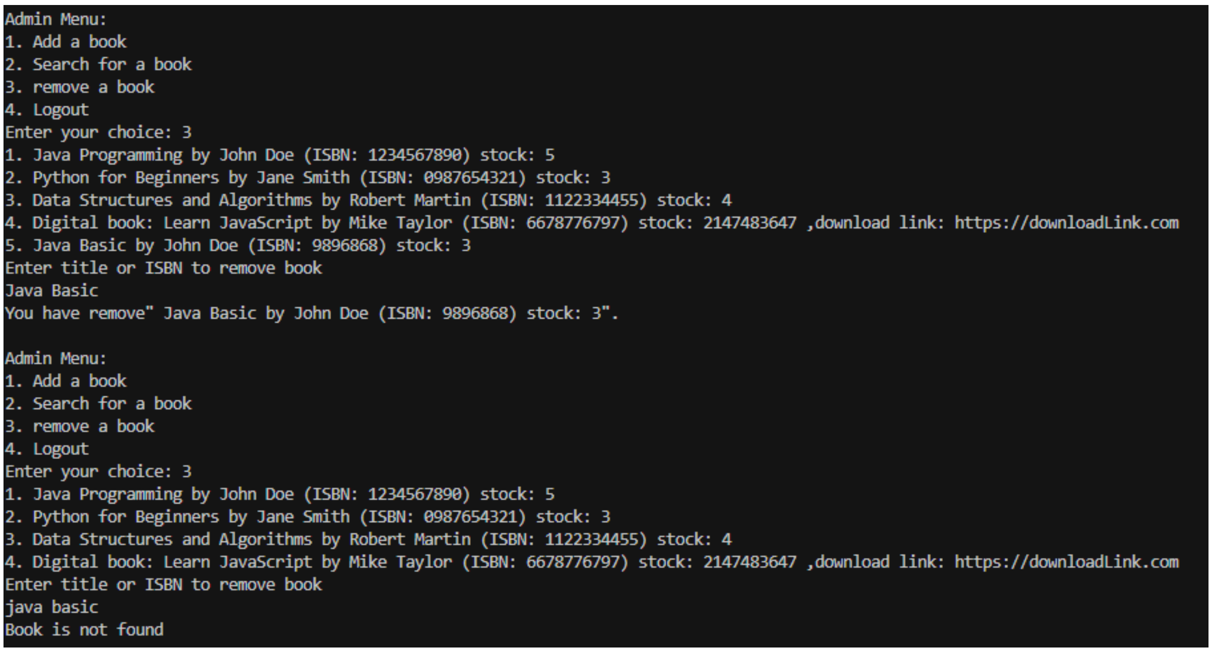
The system will prompt the admin to input the title or ISBN of the book to search for the book, if there is a matching , then the system will display the book's details, and if there is no matching, then the system will display an error message.



```
Admin Menu:
1. Add a book
2. Search for a book
3. remove a book
4. Logout
Enter your choice: 2
Enter title or ISBN to search:
Java Basic
Book found by title:Java Basic by John Doe (ISBN: 9896868) stock: 3

Admin Menu:
1. Add a book
2. Search for a book
3. remove a book
4. Logout
Enter your choice: 2
Enter title or ISBN to search:
Java oop
Book not found.
```

admin removing book:

For removing books, the system will display all the books in the library, and it will ask the admin to input the title or ISBN to remove the book. Once the admin input the book title or ISBN , it will be removed as well as its stock no matter how many it is, and if the title or ISBN is wrong, the system will display an error message.



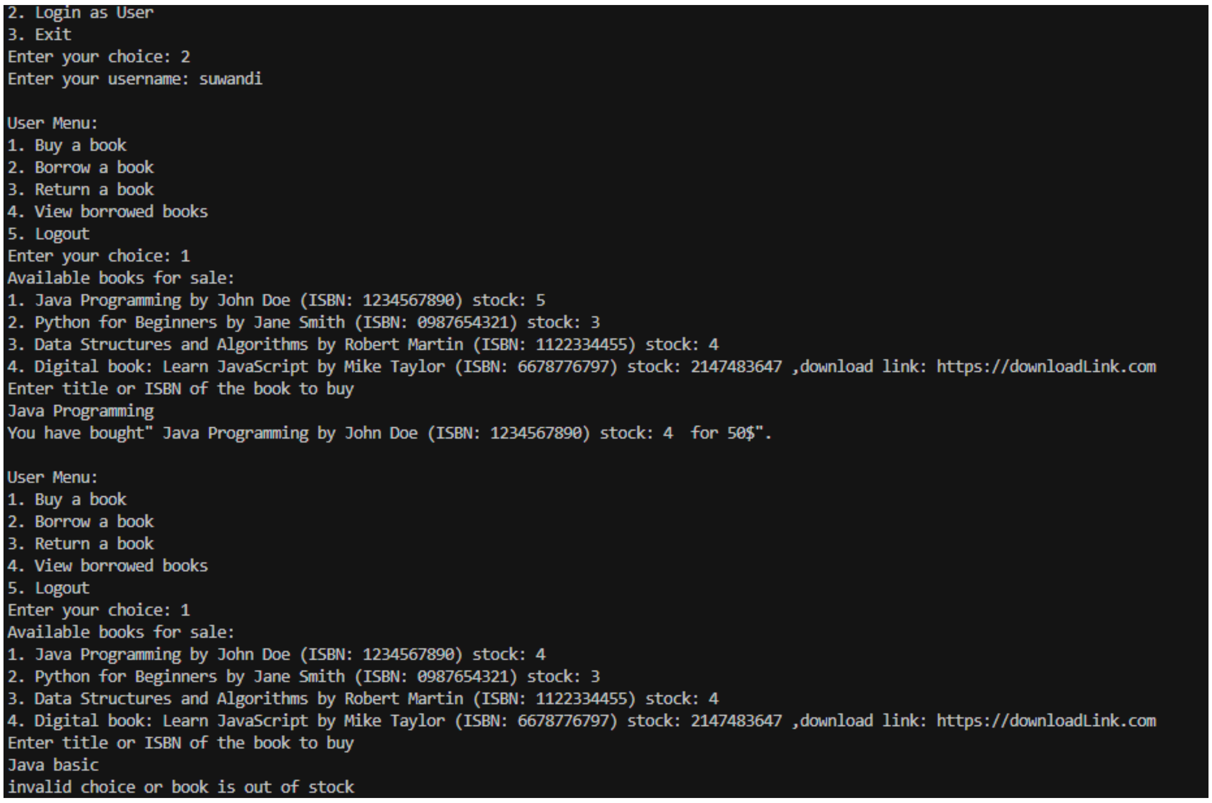
```
Admin Menu:
1. Add a book
2. Search for a book
3. remove a book
4. Logout
Enter your choice: 3
1. Java Programming by John Doe (ISBN: 1234567890) stock: 5
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
5. Java Basic by John Doe (ISBN: 9896868) stock: 3
Enter title or ISBN to remove book
Java Basic
You have remove" Java Basic by John Doe (ISBN: 9896868) stock: 3".

Admin Menu:
1. Add a book
2. Search for a book
3. remove a book
4. Logout
Enter your choice: 3
1. Java Programming by John Doe (ISBN: 1234567890) stock: 5
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
Enter title or ISBN to remove book
java basic
Book is not found
```

2. Users

The system will prompt the user to input their username that will be used in the system.

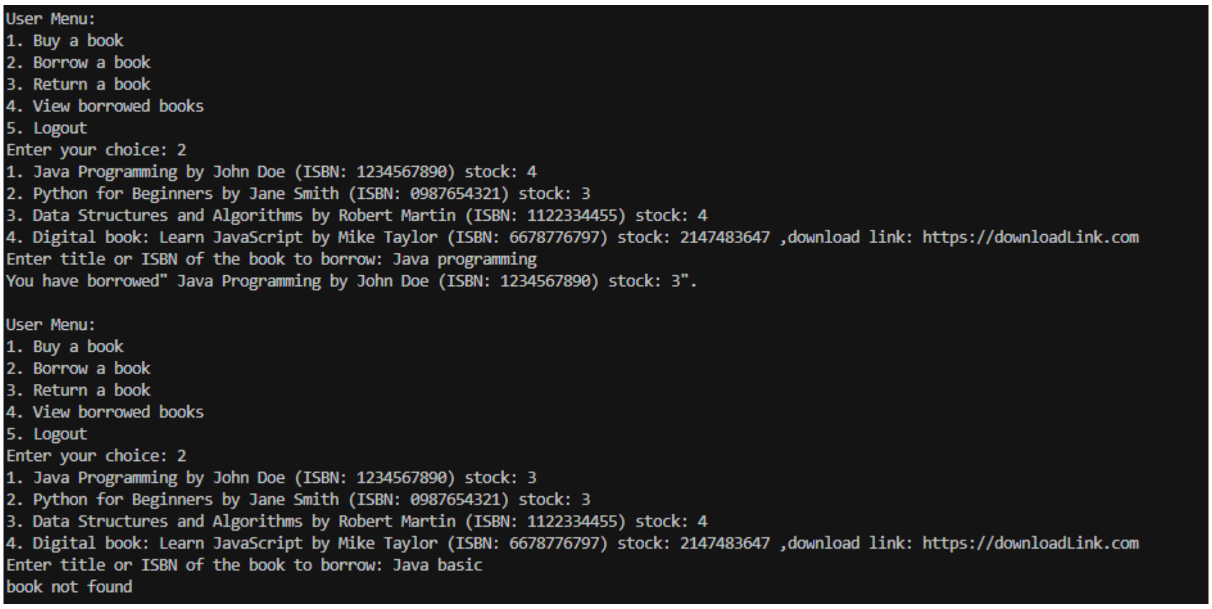
Users purchasing book :

The system will display all the available books in the library, and ask the users to input the title or ISBN of the book that they would like to buy. If there is a matching , the system will display the book details and the stock of the bought book will be decremented by 1. If there is no matching, the system will display an error message.


```
2. Login as User
3. Exit
Enter your choice: 2
Enter your username: suwandi

User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 1
Available books for sale:
1. Java Programming by John Doe (ISBN: 1234567890) stock: 5
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
Enter title or ISBN of the book to buy
Java Programming
You have bought" Java Programming by John Doe (ISBN: 1234567890) stock: 4  for 50$".

User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 1
Available books for sale:
1. Java Programming by John Doe (ISBN: 1234567890) stock: 4
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
Enter title or ISBN of the book to buy
Java basic
invalid choice or book is out of stock
```

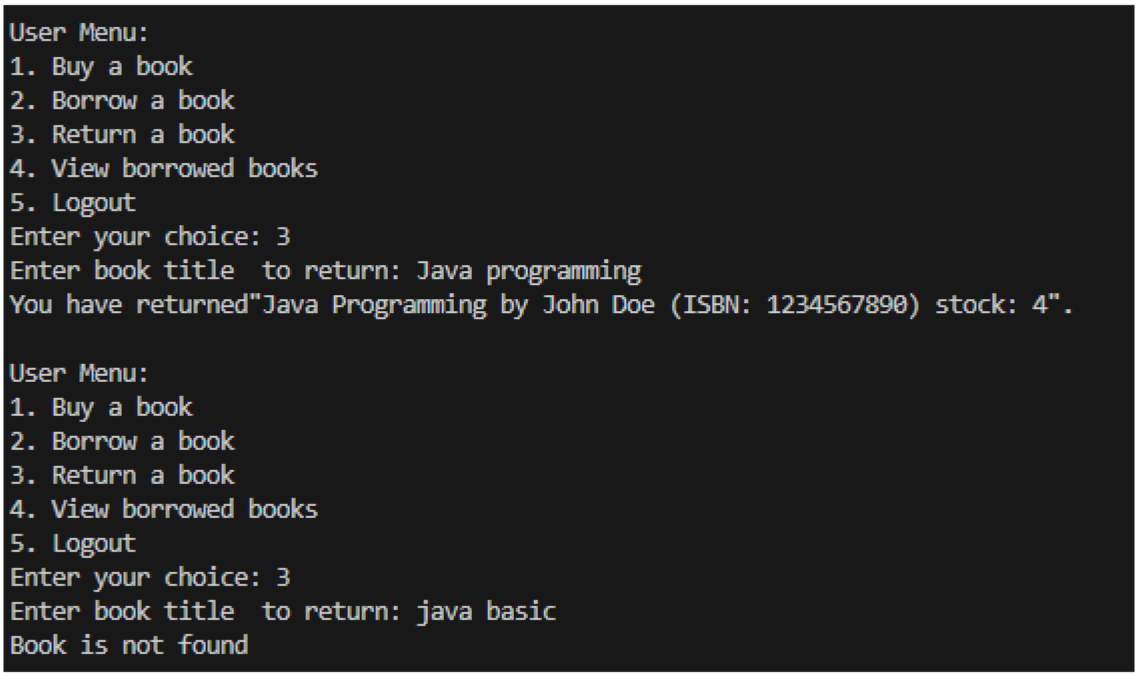
Users Borrowing book:

Same as purchasing, the system will also display all the available books in the library, and if there is matching, the stock of the book will be decremented by 1 and the book will be put in the user's borrowed books list. If there is no matching, the system will display an error message.


```
User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 2
1. Java Programming by John Doe (ISBN: 1234567890) stock: 4
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
Enter title or ISBN of the book to borrow: Java programming
You have borrowed" Java Programming by John Doe (ISBN: 1234567890) stock: 3".

User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 2
1. Java Programming by John Doe (ISBN: 1234567890) stock: 3
2. Python for Beginners by Jane Smith (ISBN: 0987654321) stock: 3
3. Data Structures and Algorithms by Robert Martin (ISBN: 1122334455) stock: 4
4. Digital book: Learn JavaScript by Mike Taylor (ISBN: 6678776797) stock: 2147483647 ,download link: https://downloadLink.com
Enter title or ISBN of the book to borrow: Java basic
book not found
```

Users borrowing book:

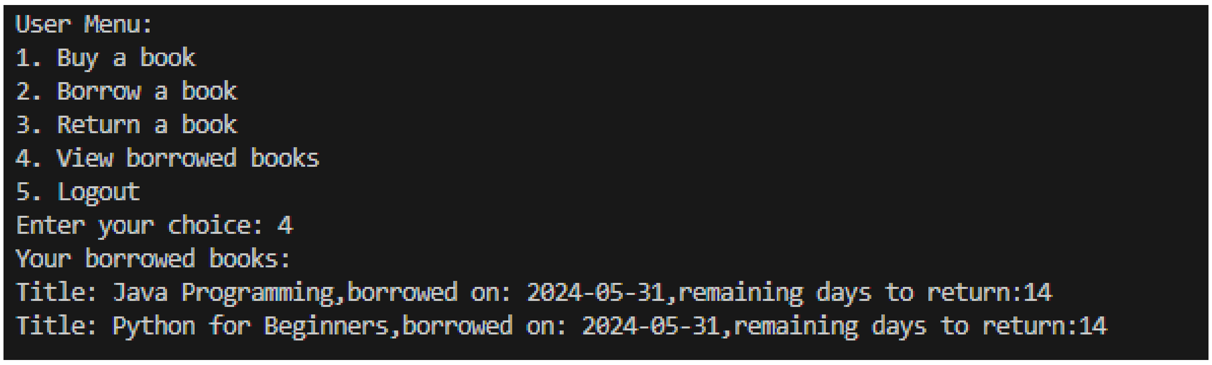
the system will ask the user to input the title or ISBN of the book that they would like to return. If there is matching, the stock of the book will be incremented by 1 which means the book is added back to the database. if there is no matching, the system will display an error message.



```
User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 3
Enter book title  to return: Java programming
You have returned"Java Programming by John Doe (ISBN: 1234567890) stock: 4".

User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 3
Enter book title  to return: java basic
Book is not found
```

Users Viewing borrowed book:

If the user wants to streamline their borrowed books, they can choose option 4 and the system will directly display what are the books they have borrowed and the remaining days to return it.



```
User Menu:
1. Buy a book
2. Borrow a book
3. Return a book
4. View borrowed books
5. Logout
Enter your choice: 4
Your borrowed books:
Title: Java Programming,borrowed on: 2024-05-31,remaining days to return:14
Title: Python for Beginners,borrowed on: 2024-05-31,remaining days to return:14
```

Chapter 4 - Resources

GeeksforGeeks. (2023, December 13). *HashMap in Java*. GeeksforGeeks. https://www.geeksforgeeks.org/java-util-hashmap-in-java-with-examples/

GeeksforGeeks. (2024, April 5). *ArrayList in Java*. GeeksforGeeks. https://www.geeksforgeeks.org/arraylist-in-java/

GeeksforGeeks. (2024b, May 6). *Linear Search Algorithm Data Structure and Algorithms Tutorials*. GeeksforGeeks. https://www.geeksforgeeks.org/linear-search/

Chapter 5 - Source code and Poster

Link to my GitHub file: https://github.com/Suwandithe/OOP_Final_Project

Link to my Poster file :

https://drive.google.com/file/d/1VUdUsoZVP5FVy0-9SgyaHINPknTCDz6y/view?usp=drivesdk