BOOK BANK MANAGEMENT

Aim:

To identify a Book Bank Management System that needs to be developed.

Problem Statement:

The main objective of the system was to design an online book-bank management system to enable a central monitoring mechanism of the book-bank be more faster and less error prone. Apart from this book-bank also used for ,

- To help the users acquire the right books for the studying at the right time.
- To ensure availability of basic textbooks to students against limited funds and to develop students ability to handle property loaned to them.
- Eliminating the need for manual registration and filling of forms by the users.
- Tracking the due dates and fine amounts for the books borrowed by the users.
- Generating reports and statics on the Book-Bank activities and performance.
- Show the availability of the books to collect them.
- To avoid due dates they can get warn or tell us through E-mails.
- Easily renewal of the books before due date crossing.

Therefore these are the problem statements associated with the Book-Bank Management System.

SRS Document & Usecase Modelling:

1. Introduction:

1.1 Purpose:

The purpose of this document is to provide a detailed specification of the Book Bank Management System. This system aims to efficiently manage the borrowing and returning of books from a book bank, streamlining the entire process for both users and administrators.

1.2 Scope:

The Book Bank Management System will be a web-based application accessible to authorized users. It will include functionalities such as user registration, book catalog management, book borrowing, book returning, and administrative features for managing users and book inventory.

1.3 Audience:

The primary audience for this document includes developers, testers, project stakeholders, and other parties involved in the development and evaluation of the Book Bank Management System.

2. Overall Description

2.1 Product Perspective:

The Book Bank Management System will be a standalone web application. It will interact with a backend database to store and retrieve user data, book information, and transaction history. The system will be integrated with a secure login mechanism to ensure that only authorized users can access its features.

2.2 User Classes and Characteristics:

The system will have two main user classes:

Students: They will be able to register, browse the available books, borrow books, and return books. Students can only borrow a limited number of books for a specific duration.

Administrators: They will have elevated privileges to manage user accounts, add/update/delete books from the catalog, and view transaction history. Administrators can also generate reports related to book availability and overdue books.

2.3 User Documentation:

The system will be accompanied by user manuals and documentation explaining its functionalities, usage guidelines, and troubleshooting procedures.

2.4 Constraints:

- The students require a computer to submit their information.
- Although the security is given high importance.
- The students has to be careful while submitting their information.

2.5 Assumptions and Dependencies:

- It is assumed that users will have basic computer literacy and web browsing skills.
- The system development will depend on the availability of the required hardware and software infrastructure.

3. Specific Requirements

3.1 Functional Requirements:

3.1.1 User Registration:

- The system shall allow new users (students) to register by providing necessary details like name, student ID, email, and password.
- The system shall validate the uniqueness of student IDs and email addresses during registration.

3.1.2 User Login:

- The system shall provide a secure login mechanism for both students and administrators.
- Users shall be authenticated based on their credentials before accessing the system.

3.1.3 Book Catalog Management:

- Administrators shall be able to add new books to the catalog with details such as title, author, ISBN, publication year, and availability status.
- Administrators shall be able to update or delete book information from the catalog.

3.1.4 Book Borrowing:

- Students shall be able to browse the available books in the catalog.
- Students shall be able to borrow books by selecting from the available options and specifying the desired duration of the loan.
- The system shall track the number of books borrowed by each student and enforce the borrowing limit.

3.1.5 Book Returning:

- Students shall be able to return books they have borrowed.
- The system shall update the book availability status upon book return.

3.1.6 View Transaction History:

Students and administrators shall be able to view their respective transaction histories, showing details of borrowed and returned books along with dates.

3.1.7 Manage User Accounts:

Administrators shall be able to manage user accounts, including creating, updating, and deleting accounts as necessary.

3.2 Non-Functional Requirements:

3.2.1 Performance:

- The system should respond to user actions promptly, with minimal delay.
- The backend database should be optimized to handle a large number of concurrent users without performance degradation.

3.2.2 Security:

- User passwords shall be securely stored using encryption techniques.
- Access to sensitive administrative functions shall be protected by role-based access control.

3.2.3 Usability:

- The user interface should be intuitive and easy to navigate.
- Clear error messages and help sections should be provided to assist users in case of any issues.

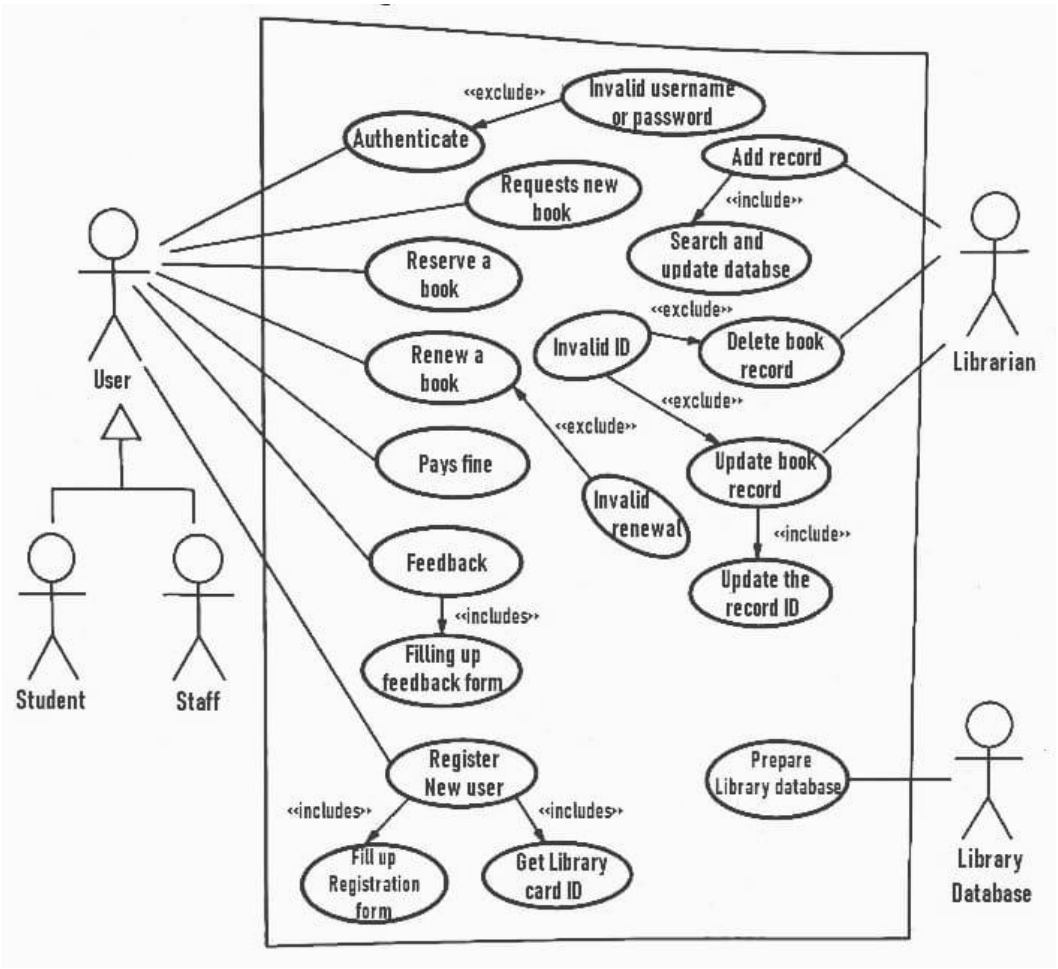
3.2.4 Reliability:

- The system shall maintain data integrity and consistency throughout its operation.
- It should be designed to recover gracefully from any unexpected failures.

Usecase Diagram:

A use case diagram provides a visual representation of the interactions between actors (users) and the system in terms of use cases (functionalities). Here is a use case diagram for the Book Bank Management System:

- **Book Bank Management System:** This is the main system, representing the Book Bank Management Software, which includes various use cases to interact with users and administrators.
- **Administrator:** This actor represents a system administrator who has elevated privileges and can manage user accounts, book catalog, and view transaction history.
- **User Registration:** Represents the use case where a student can register on the system by providing necessary details.

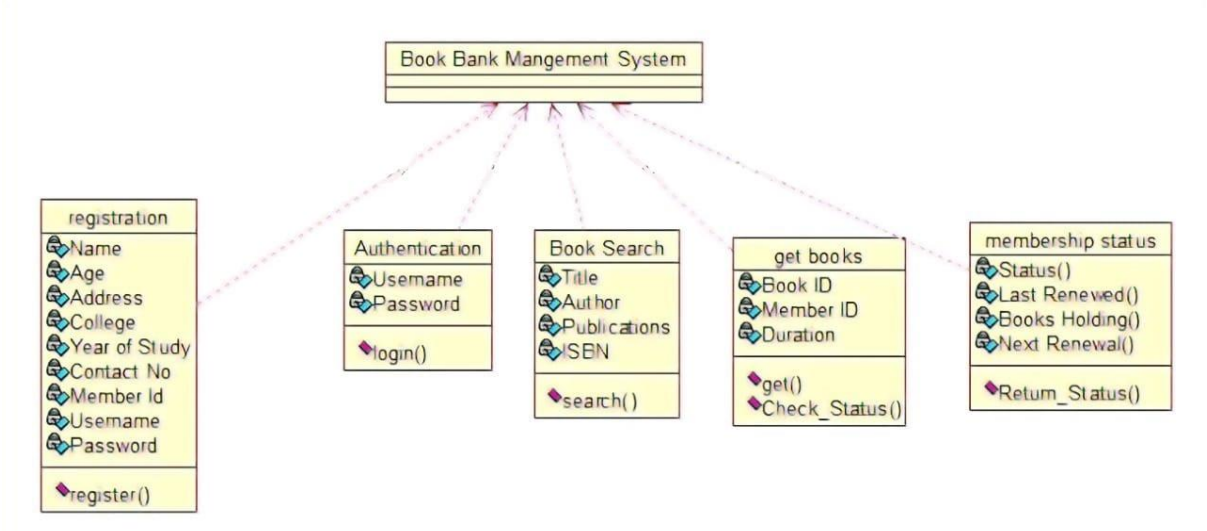


- **User Login:** This use case allows both students and administrators to log in to the system using their credentials.
- **Browse Book Catalog:** This use case enables students to view the available books in the book bank.
- **Borrow Book:** Students can borrow books from the book bank using this use case.
- **Return Book:** Students can return borrowed books using this use case.
- **View Transaction History:** Both students and administrators can view their respective transaction histories, showing details of borrowed and returned books along with dates.

Each use case represents a specific functionality of the system. The lines connecting the actors to the use cases indicate which actors can interact with those use cases. For example, the Administrator can interact with "Manage User Accounts," "Book Catalog Management," and "View Transaction History" use

cases, while the Student can access "Browse Book Catalog," "Borrow Book," and "Return Book" use cases.

Class Diagram:



- The Book Bank Management System class represents the core system. It contains collections of User, Book, and Transaction objects. It also has methods for adding/removing users and books, handling book borrowing and returning, and generating reports.

- The User class represents users of the system, such as students. It has attributes like userId, name, and email, as well as a method getUserInfo().

- The Book class represents books in the book bank. It has attributes like bookId, title, author, ISBN, and publicationYear.

- The Transaction class represents book borrowing transactions. It has attributes like transactionId, userId, bookId, dateBorrowed, and dateReturned. It also has a method to calculate fines if a book is returned late.

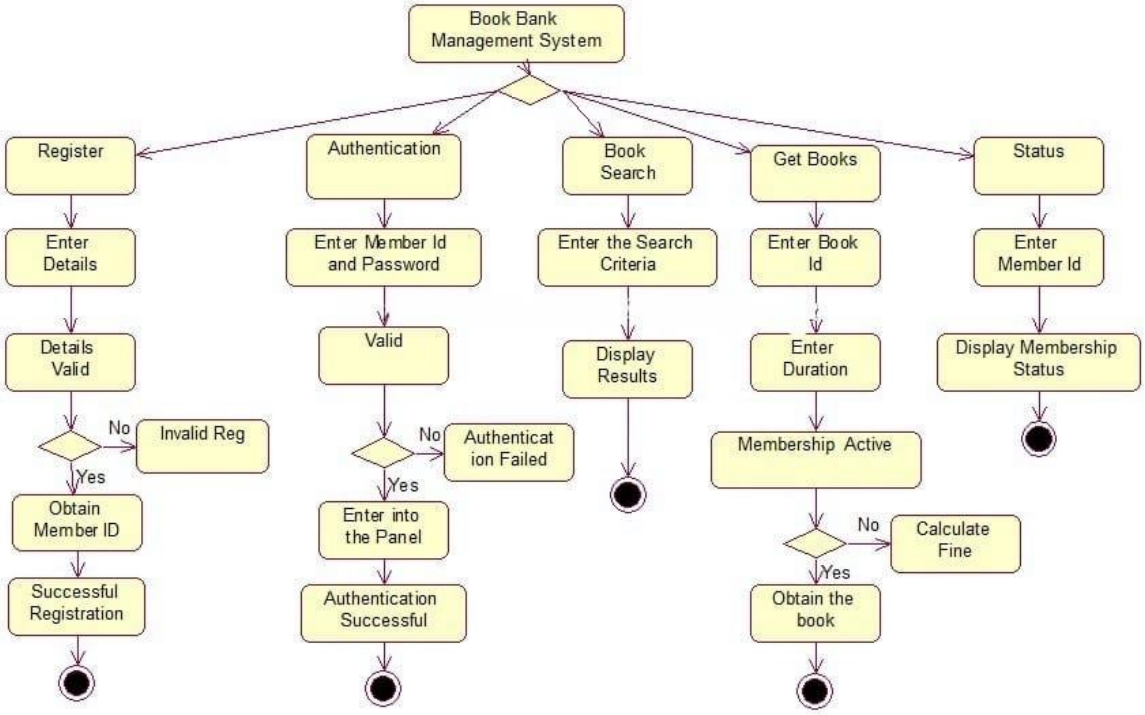
- The Report class represents system-generated reports. It has attributes like reportId and content, and a method to generate reports.

Sequence Diagram:

In this sequence diagram:

1. The User sends a request to borrow a book to the Book Bank System.
2. The User confirms their credentials (e.g., username and password) to the system.
3. The Book Bank System verifies the user's credentials.
4. The Book Bank System checks the availability of the requested book.
5. If the book is available, the Book Bank System reserves the book for the user and sends a confirmation.

Activity Diagram:



1. Start Activity: This is the beginning of the activity diagram.

2. User Activity - Request to Borrow Book: The user initiates the process by requesting to borrow a book.

3. System Activity - Verify User Credentials: The system verifies the user's credentials (e.g., username and password) to ensure the user is authorized to borrow a book.

4. User Activity - Select a Book: The user selects a book they want to borrow from the available options.

5. System Activity - Check Book Availability: The system checks if the selected book is available for borrowing.

6. Decision Point - Is the book available?: Depending on the result of the availability check, the process takes one of two paths:

    - If the book is available, the process continues to the next step.

- If the book is not available, the process may involve notifying the user that the book is unavailable.

7. User Activity - Choose Duration and Confirm: If the book is available, the user chooses the duration for which they want to borrow the book (e.g., the number of days) and confirms the borrowing.

8. System Activity - Reserve Book for User: The system reserves the selected book for the user for the specified duration.

9. End Activity - Borrowing Complete: This marks the end of the process, indicating that the user has successfully borrowed the book.

Package Diagram:

A package diagram is used to organize and show the structure of a system by grouping related elements into packages. In the context of a Book Bank Management System, here's a simplified package diagram that represents the high-level structure of the system:

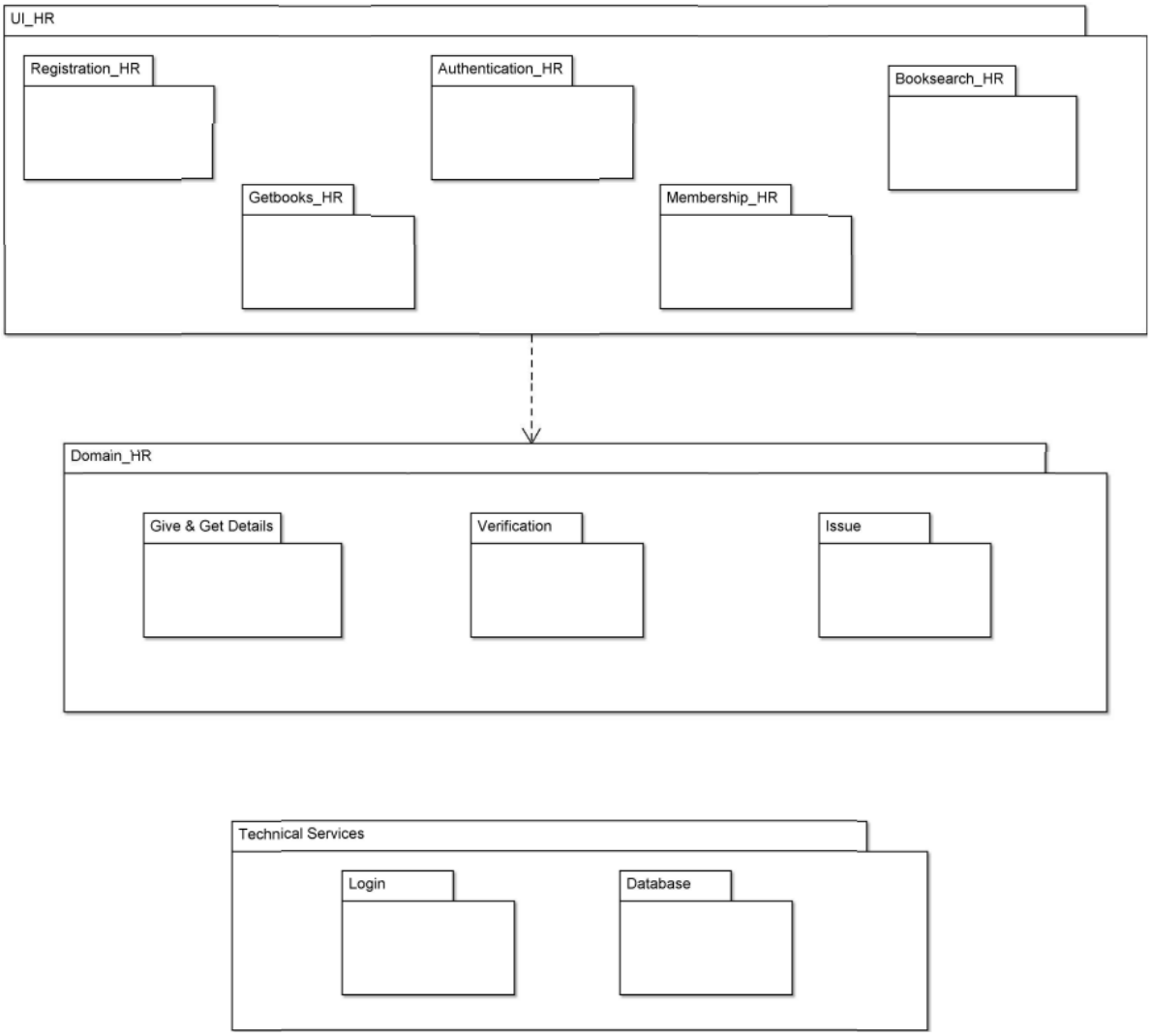
In this package diagram:

- Book Bank Management System: This is the top-level package representing the entire system.

- User Management: This package contains functionalities related to user management, including user registration, authentication, profile management, and role management.

- Book Management: This package includes functionalities related to book management, such as managing the book catalog, handling borrowing and returning of books, book reservation, and fine calculation.

- Transaction Management: This package manages the recording of transactions and maintains a history of all transactions.



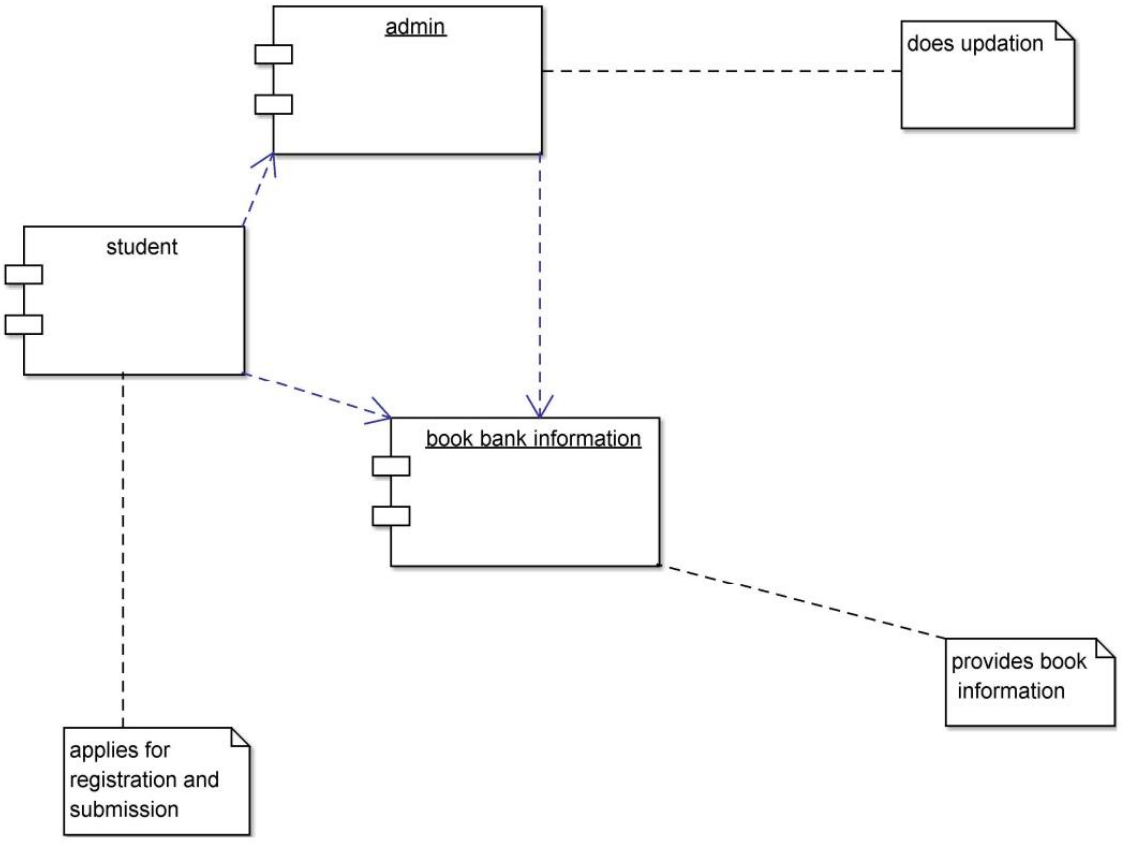
- Reporting: The reporting package handles the generation and export of reports, such as transaction reports, book availability reports, and user activity reports.

- Security: This package encapsulates security-related functionalities, including access control and data encryption.

Each of these packages can be further broken down into sub-packages or classes as needed, depending on the system's complexity and requirements. Additionally, the relationships between these packages, such as dependencies and associations, can be specified in the diagram to provide a more detailed view of the system's architecture.

Component Diagram:

A component diagram illustrates the high-level structure of a system, focusing on the components or modules and their dependencies. In the context of a Book Bank Management System, here's a simplified component diagram that represents the major components of the system and their interactions:



In this component diagram:

- Book Bank Management System: This is the top-level component representing the entire system.

- User Management: This component manages user-related functionalities, including registration, authentication, profile management, and role management.

- Book Management: This component handles book-related functionalities, such as managing the book catalog, book borrowing and returning, book reservation, and fine calculation.

- Transaction Management: This component is responsible for transaction-related functionalities, including recording transactions and maintaining transaction history.

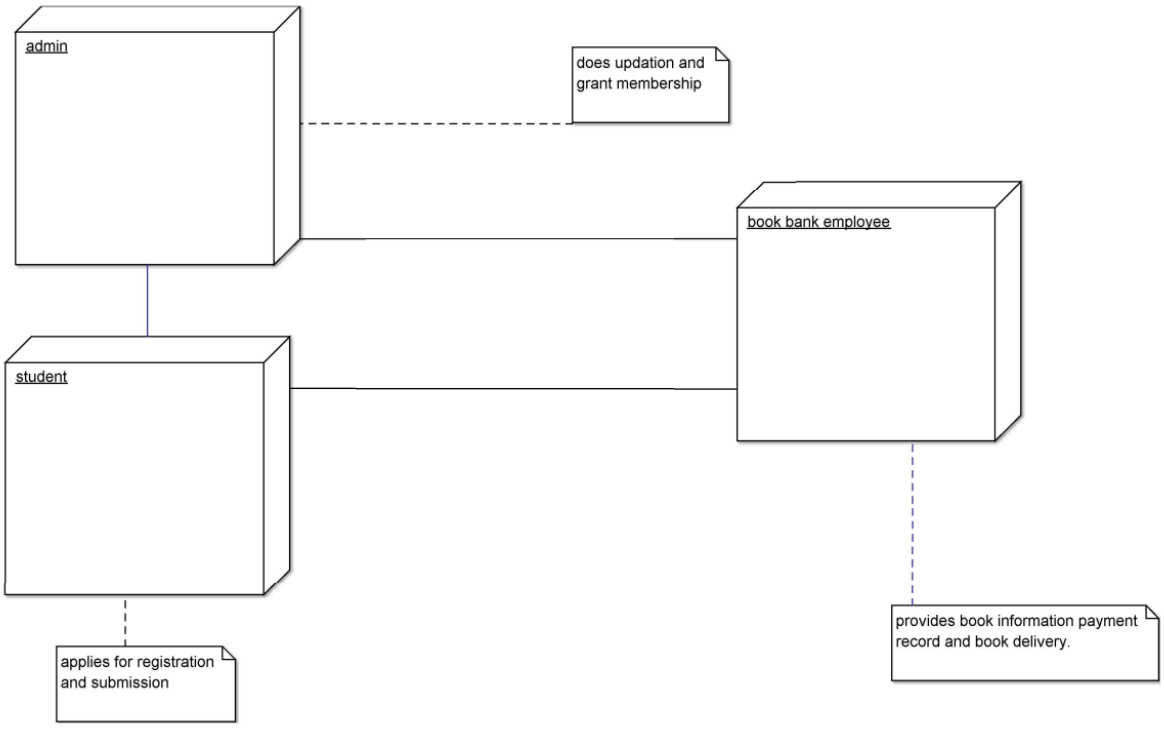
- Reporting: The reporting component takes care of report generation and export, providing various reports related to system activities.

- Security: This component focuses on security aspects, such as access control and data encryption, to ensure the system's integrity and protect sensitive data.

Each of these components may further consist of sub-components, classes, and interfaces depending on the system's design. The component diagram gives an overview of the system's architecture and how its major components interact.

Deployment Diagram:

A deployment diagram illustrates the physical architecture of a system, showing how software components are deployed onto hardware nodes. In the context of a Book Bank Management System, here's a simplified deployment diagram that represents the major hardware and software components and their interactions:



In this deployment diagram:

- User's Device: Represents the user's device (e.g., a laptop, tablet, or smartphone) where they access the Book Bank Management System through a web browser.

- Web Server: Hosts the web application of the Book Bank Management System. It handles HTTP requests from users' devices and serves web pages.

- Web Application: The software component that includes the user interface and application logic. It runs on the web server and communicates with the database server.

- Database Server: Hosts the database management system (DBMS) that stores and manages data related to users, books, transactions, and more.

- Database Management System (DBMS): The software component responsible for managing and querying the database. It interacts with the database to store and retrieve data.

- Database: Stores data related to the Book Bank Management System, including user data, book catalog, transaction records, and more.

- The arrows in the diagram represent the communication between components:

- The web browser on the user's device communicates with the web server over HTTP to request web pages and send user input.

- The web application on the web server communicates with the database server, typically using a database connection method like JDBC (Java Database Connectivity).

This deployment diagram provides an overview of how the software components of the Book Bank Management System are distributed across hardware nodes. In a real-world deployment, you may have additional components for load balancing, redundancy, and security.

Statechart Diagram:

A statechart diagram, also known as a state machine diagram, is a visual representation of the various states that an object or system can go through and the transitions between those states. In the context of a Book Bank Management System, we can create a statechart diagram to illustrate the states and transitions related to user registration, authentication, getting books, and searching for books. Let's break down these states and transitions:

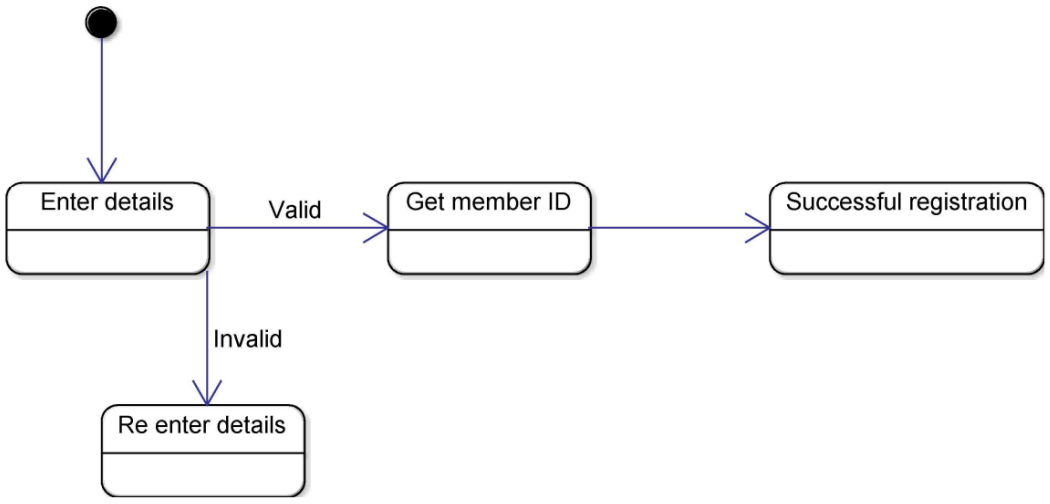
1. **User Registration:**

States:

- Initial: This is the starting point before registration begins.
- Entering Details: The user is in the process of entering registration details.
- Validation: The system is validating the entered details.
- Registered: The user has successfully registered.

Transitions:

- From Initial to Entering Details: Triggered when the user starts the registration process.
- From Entering Details to Validation: Triggered when the user submits their registration details.
- From Validation to Registered: Triggered when the system confirms that the registration details are valid.



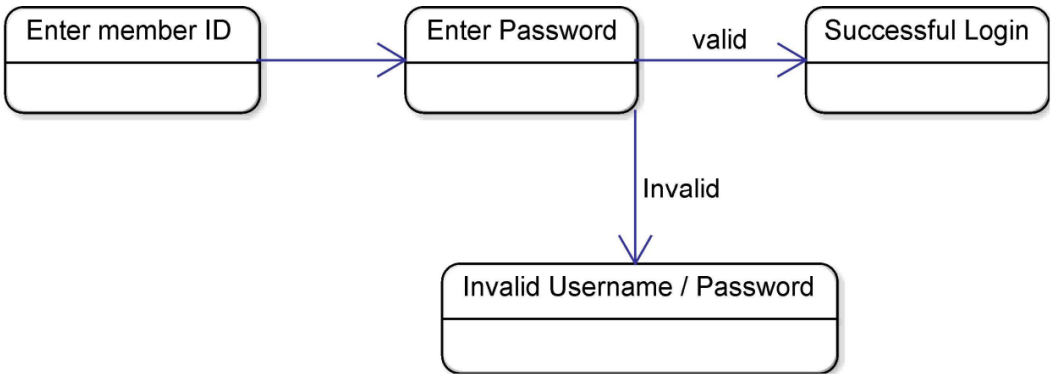
2. Authentication:

States:

- Initial: The starting point before authentication begins.
- Entering Credentials: The user is entering their login credentials (e.g., username and password).
- Validation: The system is validating the entered credentials.
- Authenticated: The user has successfully authenticated.

Transitions:

- From Initial to Entering Credentials: Triggered when the user initiates the login process.
- From Entering Credentials to Validation: Triggered when the user submits their login credentials.
- From Validation to Authenticated: Triggered when the system confirms that the credentials are valid.



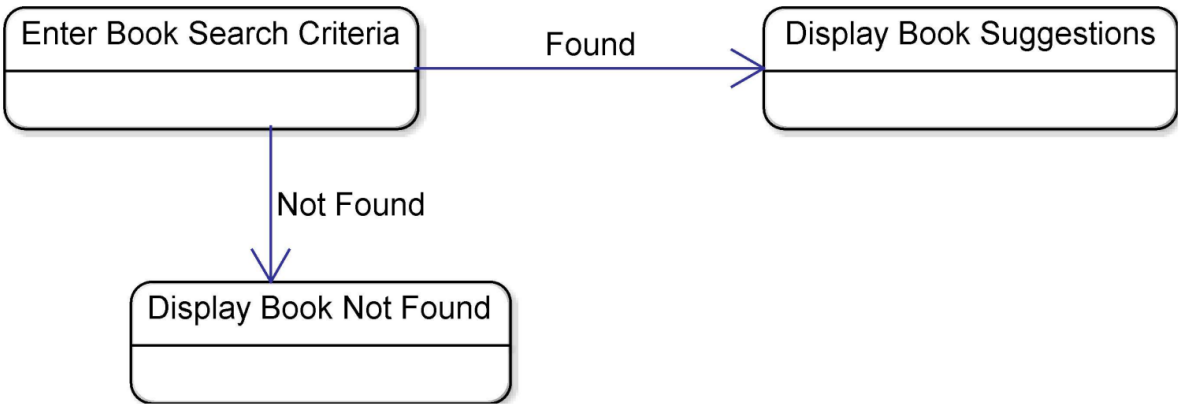
3. Searching for Books:

States:

- Initial: The starting point before the user initiates a search.
- Searching: The user is actively searching for books.
- Search Results: The system displays the search results.

Transitions:

- From Initial to Searching: Triggered when the user starts a search.
- From Searching to Search Results: Triggered when the system retrieves and displays the search results.



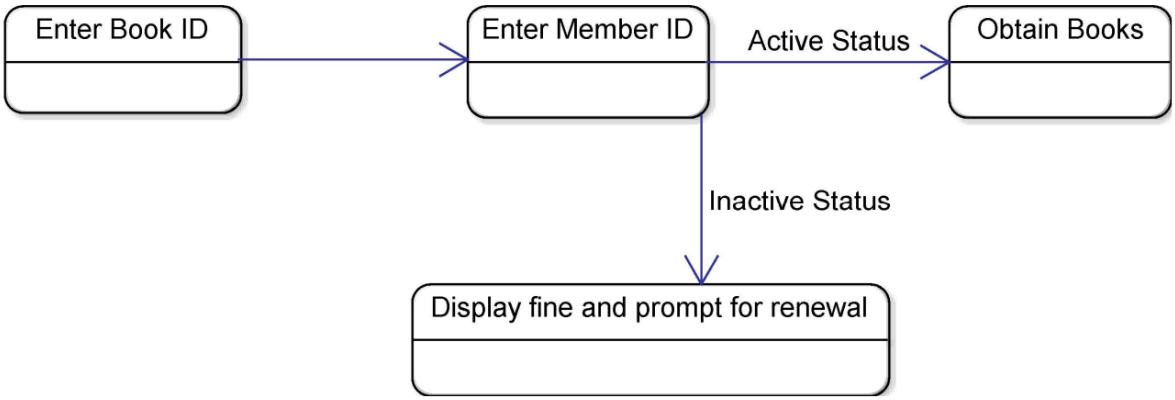
4. **Getting Books:**

States:

- Initial: The starting point before the user requests to get books.
- Requesting Books: The user is selecting books to borrow.
- Confirming Request: The user is confirming the selection.
- Transaction Processing: The system processes the request.
- Books Obtained: The user has successfully borrowed the selected books.

Transitions:

- From Initial to Requesting Books: Triggered when the user initiates the process of selecting books to borrow.
- From Requesting Books to Confirming Request: Triggered when the user confirms the selection.
- From Confirming Request to Transaction Processing: Triggered when the user submits the request.
- From Transaction Processing to Books Obtained: Triggered when the system successfully processes the request.



Coding:

```
import java.util.ArrayList;
import java.util.Scanner;

class Book {
    private String title;
    private String author;
    private boolean available;

    public Book(String title, String author) {
        this.title = title;
        this.author = author;
        this.available = true;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public boolean isAvailable() {
        return available;
    }

    public void borrow() {
```

```
        available = false;
    }

    public void returnBook() {
        available = true;
    }

    @Override
    public String toString() {
        return "Title: " + title + "\nAuthor: " + author + "\nAvailable: " + available;
    }
}

class User {
    private String name;

    public User(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    @Override
    public String toString() {
        return "User Name: " + name;
    }
}

public class LibrarySystem {
    private static ArrayList<Book> books = new ArrayList<>();
    private static ArrayList<User> users = new ArrayList<>();

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("\nLibrary Management System");
            System.out.println("1. Add Book");
            System.out.println("2. Add User");
            System.out.println("3. Borrow Book");
            System.out.println("4. Return Book");
            System.out.println("5. List Books");
```

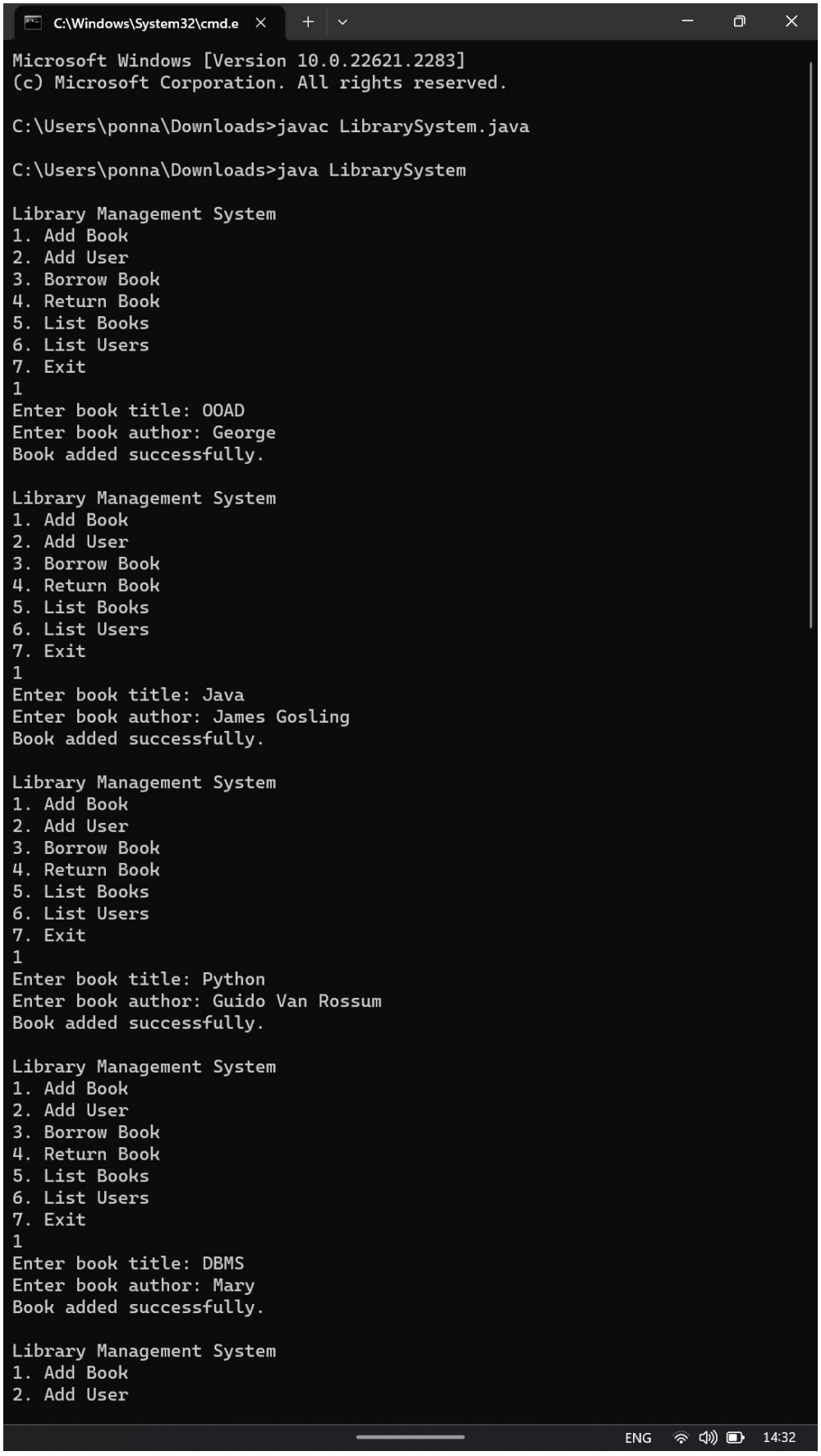
```
        System.out.println("6. List Users");
        System.out.println("7. Exit");

        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                addBook(scanner);
                break;
            case 2:
                addUser(scanner);
                break;
            case 3:
                borrowBook(scanner);
                break;
            case 4:
                returnBook(scanner);
                break;
            case 5:
                listBooks();
                break;
            case 6:
                listUsers();
                break;
            case 7:
                System.out.println("Exiting the program.");
                scanner.close();
                System.exit(0);
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}

private static void addBook(Scanner scanner) {
    System.out.print("Enter book title: ");
    String title = scanner.nextLine();
    System.out.print("Enter book author: ");
    String author = scanner.nextLine();

    Book book = new Book(title, author);
    books.add(book);
```

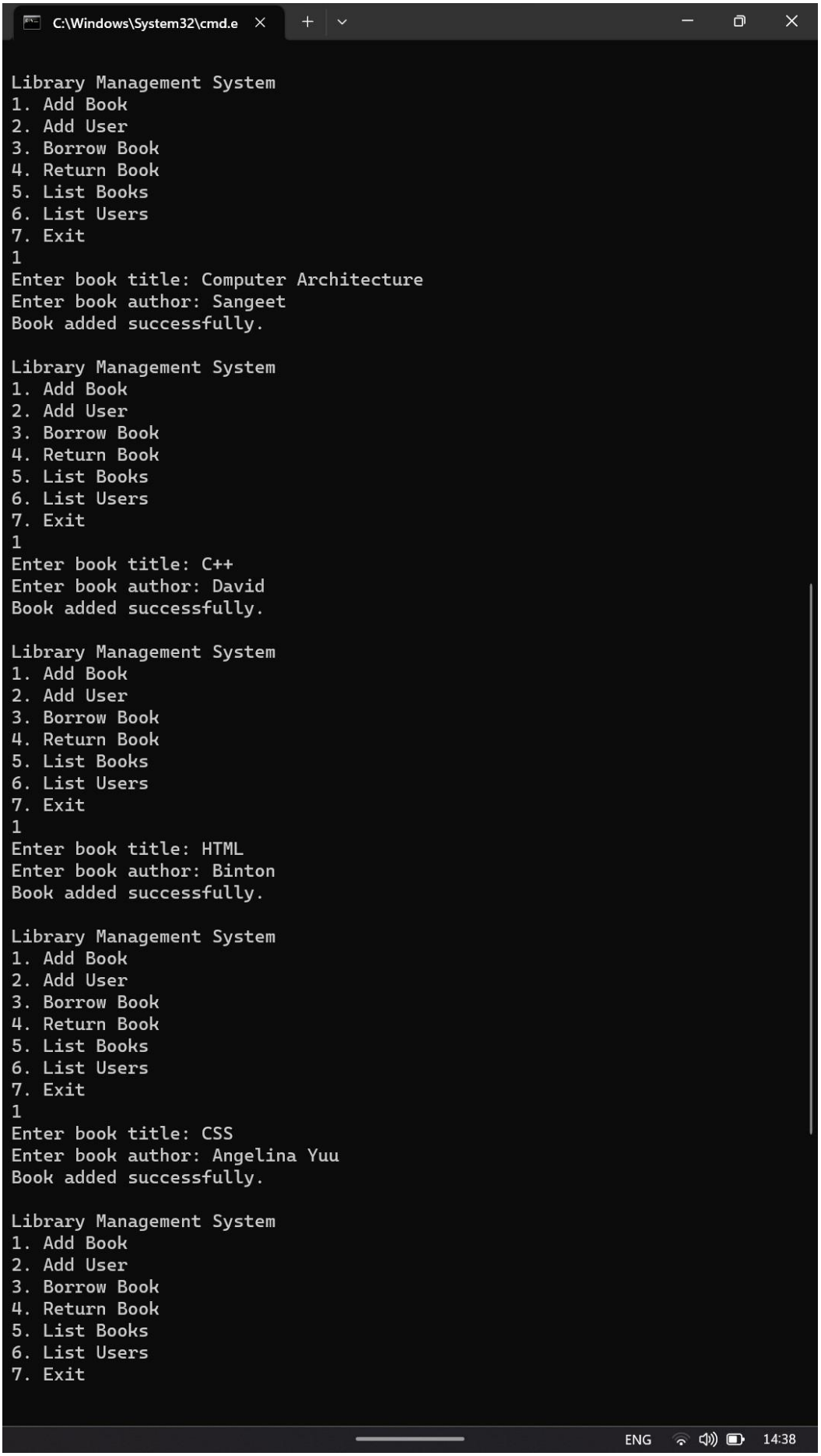
```
        System.out.println("Book added successfully.");
    }

    private static void addUser(Scanner scanner) {
        System.out.print("Enter user name: ");
        String name = scanner.nextLine();

        User user = new User(name);
        users.add(user);
        System.out.println("User added successfully.");
    }

    private static void borrowBook(Scanner scanner) {
        System.out.print("Enter user name: ");
        String userName = scanner.nextLine();

        User user = findUser(userName);
        if (user == null) {
            System.out.println("User not found.");
            return;
        }

        System.out.print("Enter book title to borrow: ");
        String bookTitle = scanner.nextLine();

        Book book = findBook(bookTitle);
        if (book == null) {
            System.out.println("Book not found.");
            return;
        }

        if (!book.isAvailable()) {
            System.out.println("Book is not available for borrowing.");
            return;
        }

        book.borrow();
        System.out.println("Book borrowed successfully.");
    }

    private static void returnBook(Scanner scanner) {
        System.out.print("Enter user name: ");
        String userName = scanner.nextLine();
```

```
        User user = findUser(userName);
        if (user == null) {
            System.out.println("User not found.");
            return;
        }

        System.out.print("Enter book title to return: ");
        String bookTitle = scanner.nextLine();

        Book book = findBook(bookTitle);
        if (book == null) {
            System.out.println("Book not found.");
            return;
        }

        if (book.isAvailable()) {
            System.out.println("This book is already in the library.");
            return;
        }

        book.returnBook();
        System.out.println("Book returned successfully.");
    }

    private static void listBooks() {
        System.out.println("\nList of Books:");
        for (Book book : books) {
            System.out.println("\n" + book);
        }
    }

    private static void listUsers() {
        System.out.println("\nList of Users:");
        for (User user : users) {
            System.out.println("\n" + user);
        }
    }

    private static User findUser(String userName) {
        for (User user : users) {
            if (user.getName().equalsIgnoreCase(userName)) {
                return user;
```

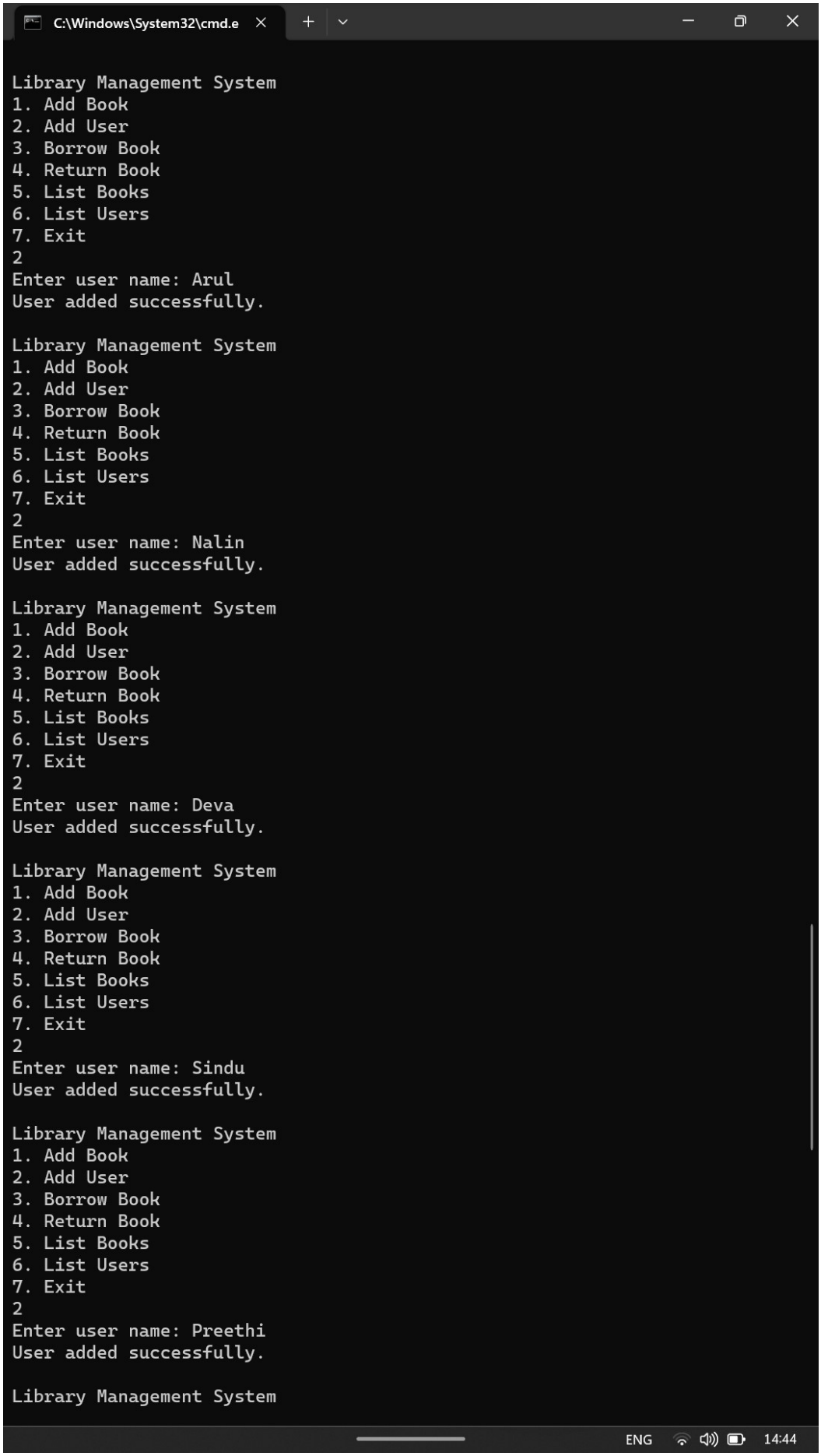
```
            }
        }
        return null;
    }

    private static Book findBook(String bookTitle) {
        for (Book book : books) {
            if (book.getTitle().equalsIgnoreCase(bookTitle)) {
                return book;
            }
        }
        return null;
    }
}
```

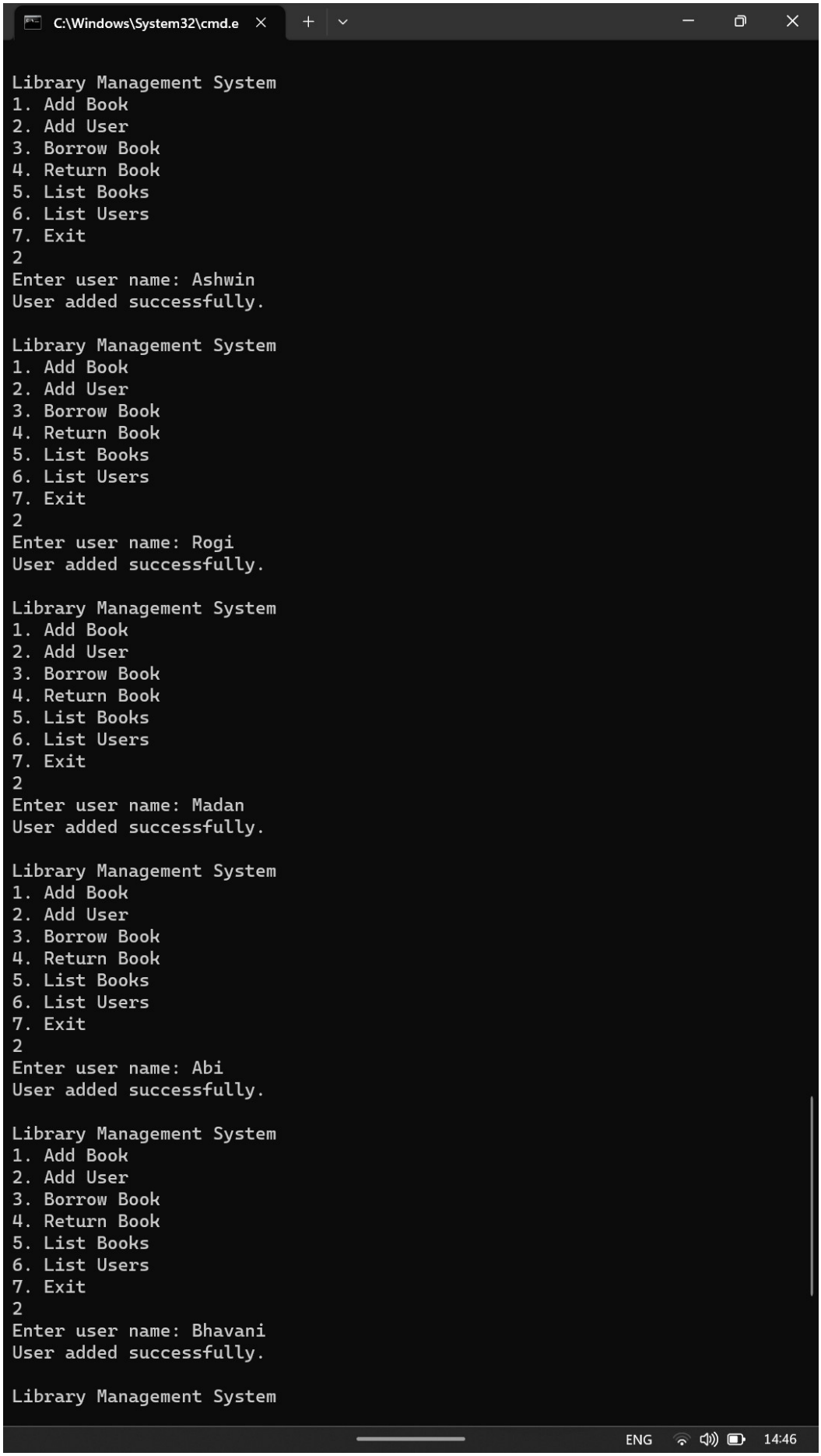
Output:

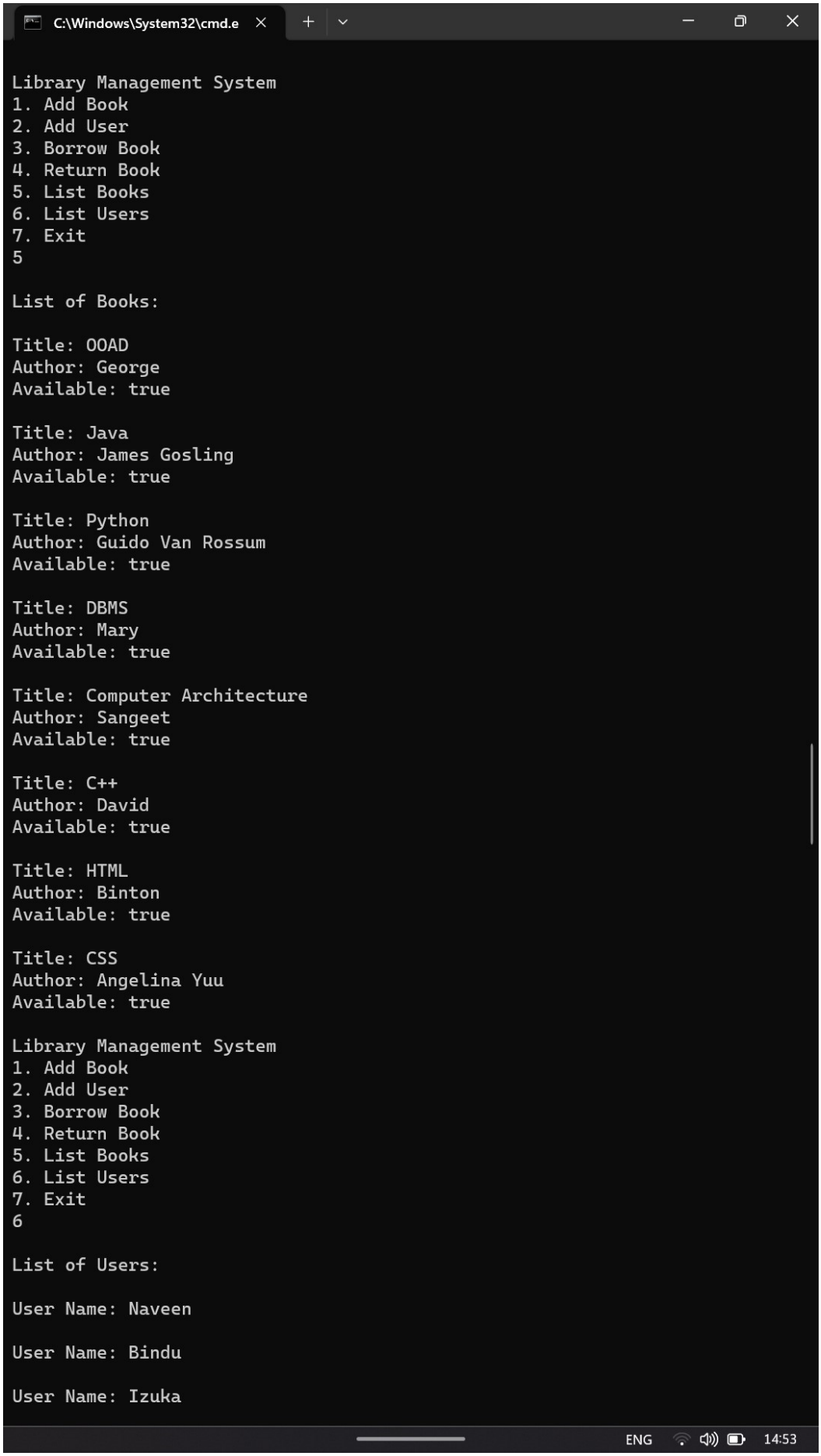

C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22621.2283]
(c) Microsoft Corporation. All rights reserved.
C:\Users\ponna\Downloads>javac LibrarySystem.java
C:\Users\ponna\Downloads>java LibrarySystem
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: OOAD
Enter book author: George
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: Java
Enter book author: James Gosling
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: Python
Enter book author: Guido Van Rossum
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: DBMS
Enter book author: Mary
Book added successfully.
Library Management System
1. Add Book
2. Add User
ENG
14:32

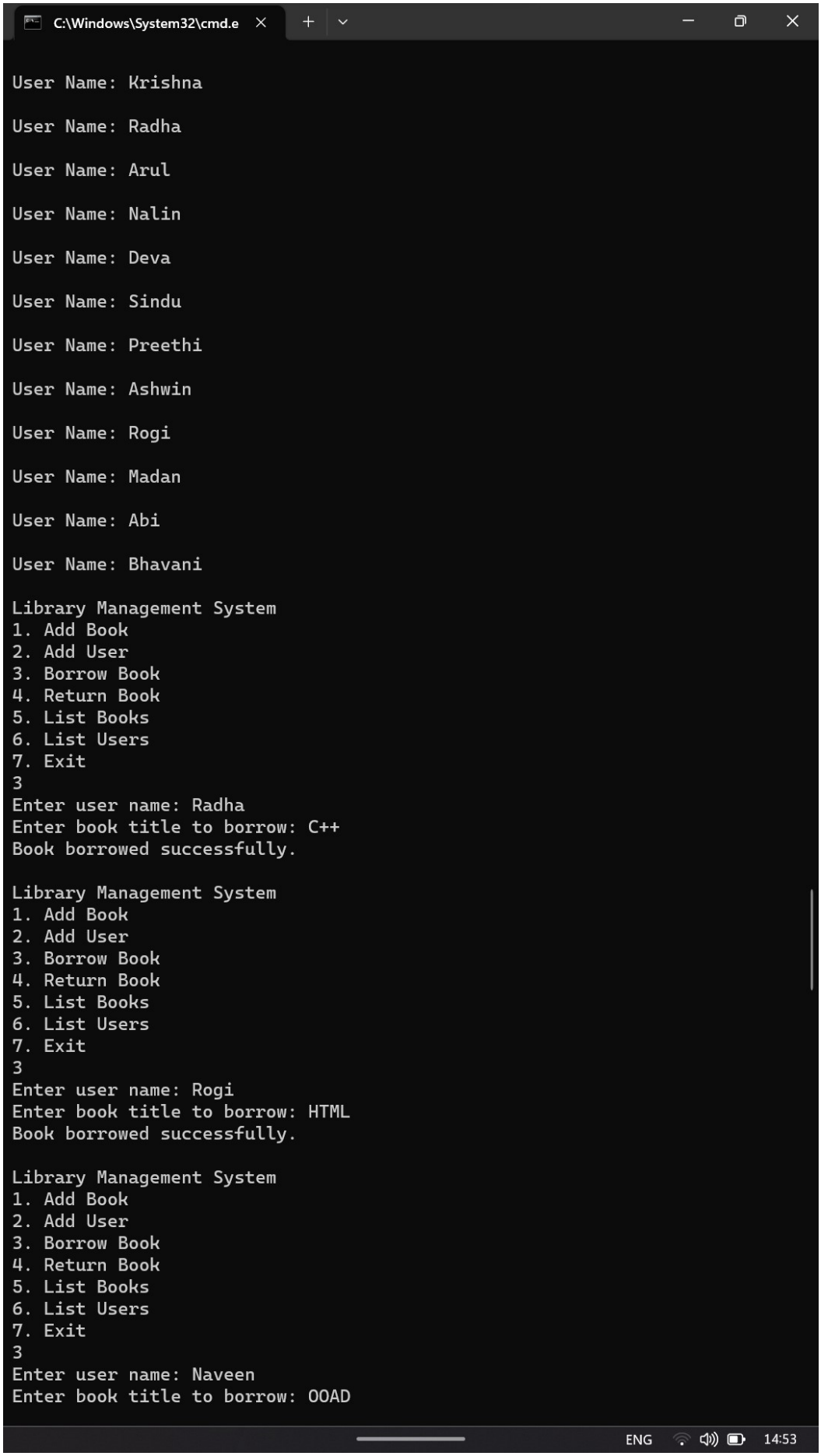

C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: Computer Architecture
Enter book author: Sangeet
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: C++
Enter book author: David
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: HTML
Enter book author: Binton
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
1
Enter book title: CSS
Enter book author: Angelina Yuu
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
ENG
14:38

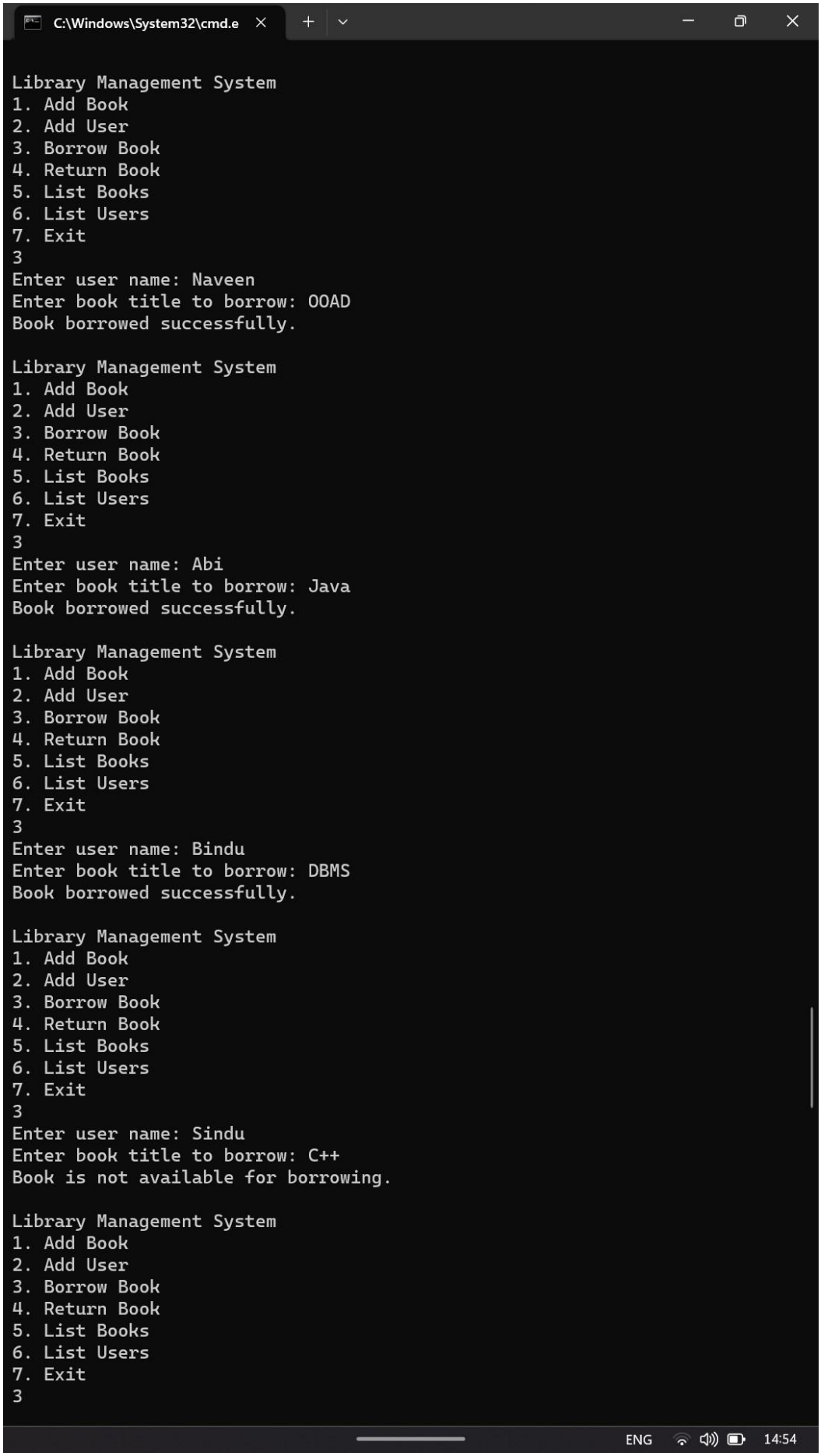

C:\Windows\System32\cmd.e
Book added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Naveen
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Bindu
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Izuka
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Krishna
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Radha
User added successfully.
ENG
14:42

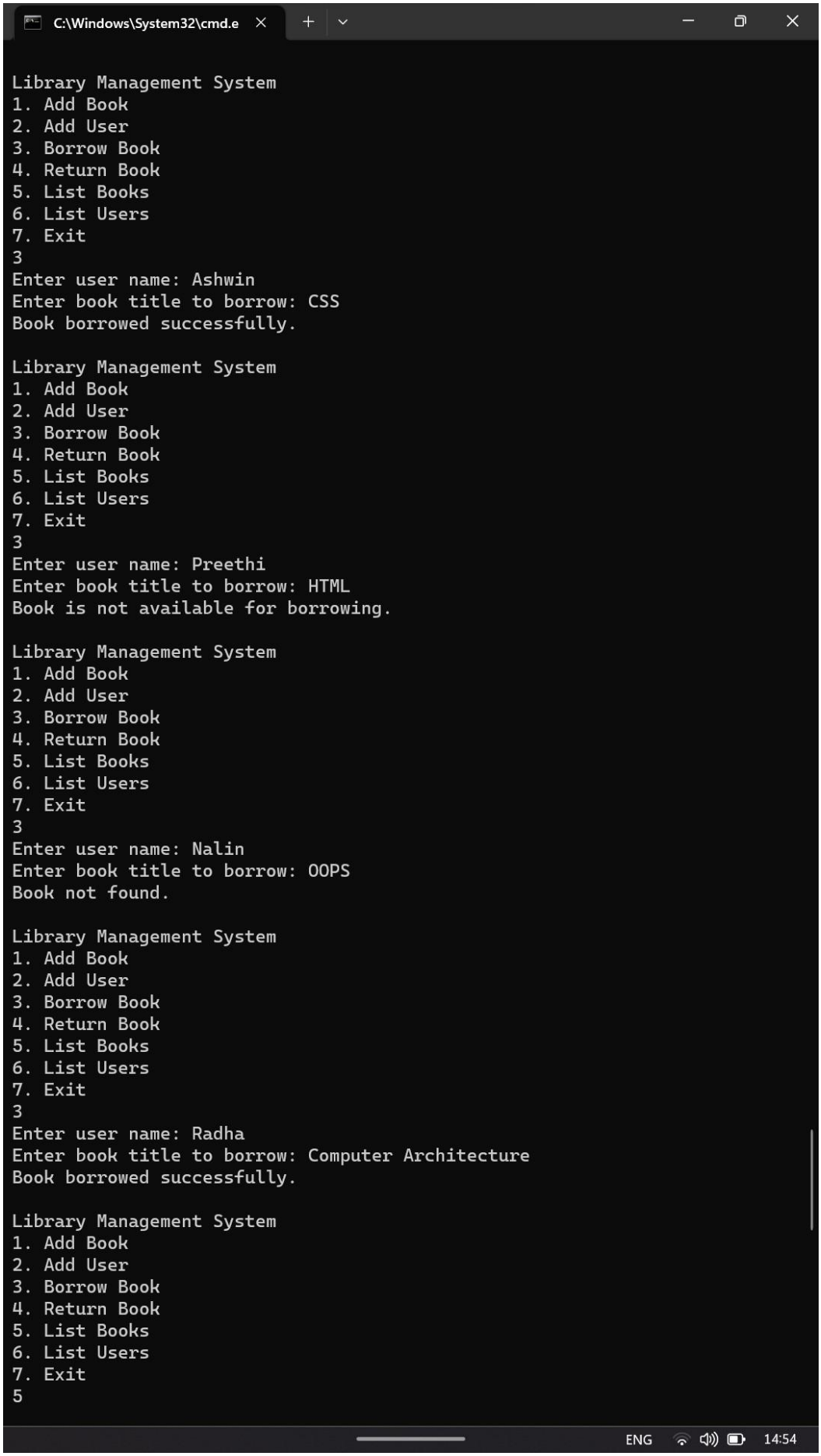

C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Arul
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Nalin
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Deva
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Sindu
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Preethi
User added successfully.
Library Management System
ENG
14:44

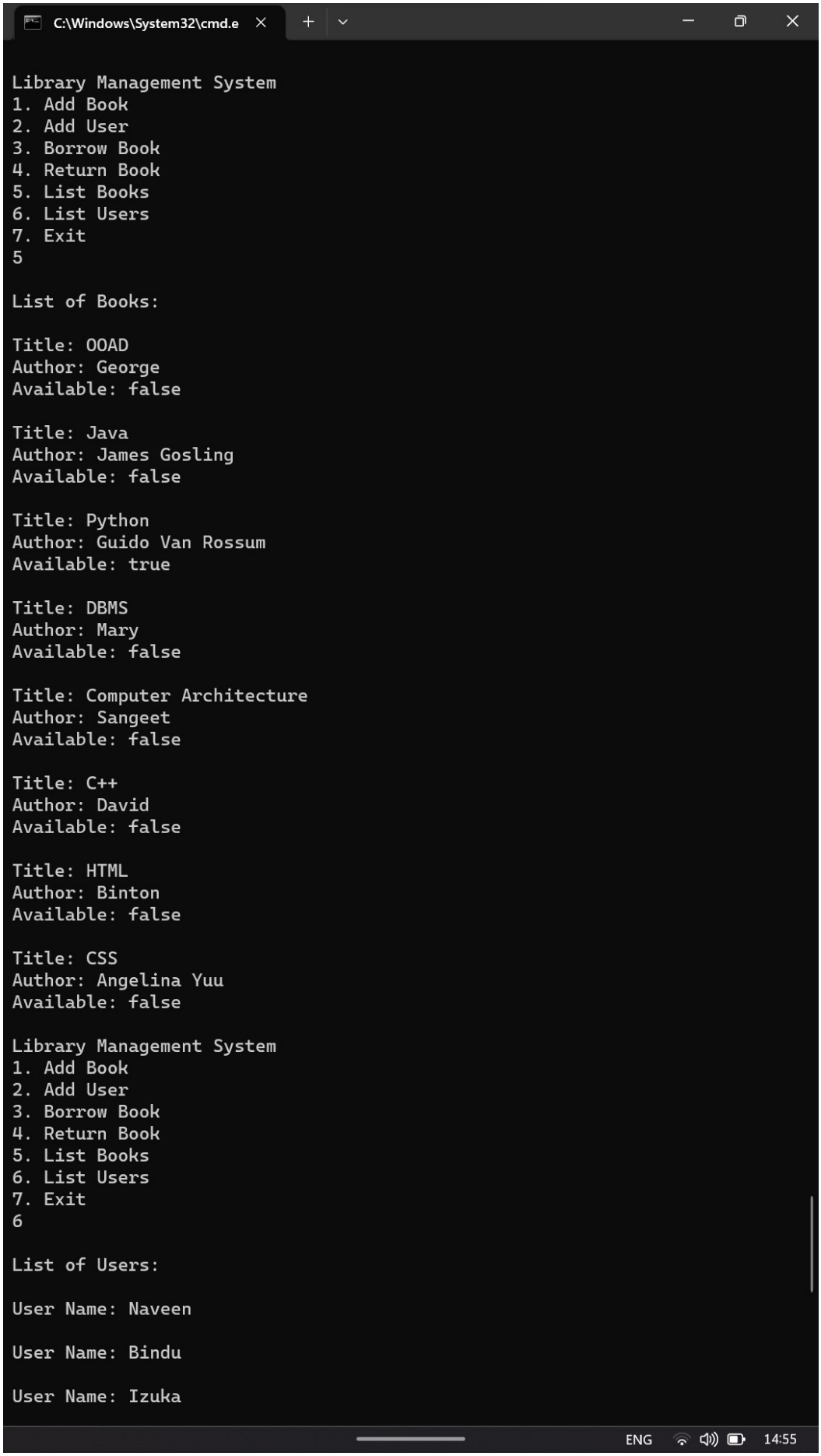

C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Ashwin
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Rogi
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Madan
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Abi
User added successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
2
Enter user name: Bhavani
User added successfully.
Library Management System
ENG
14:46

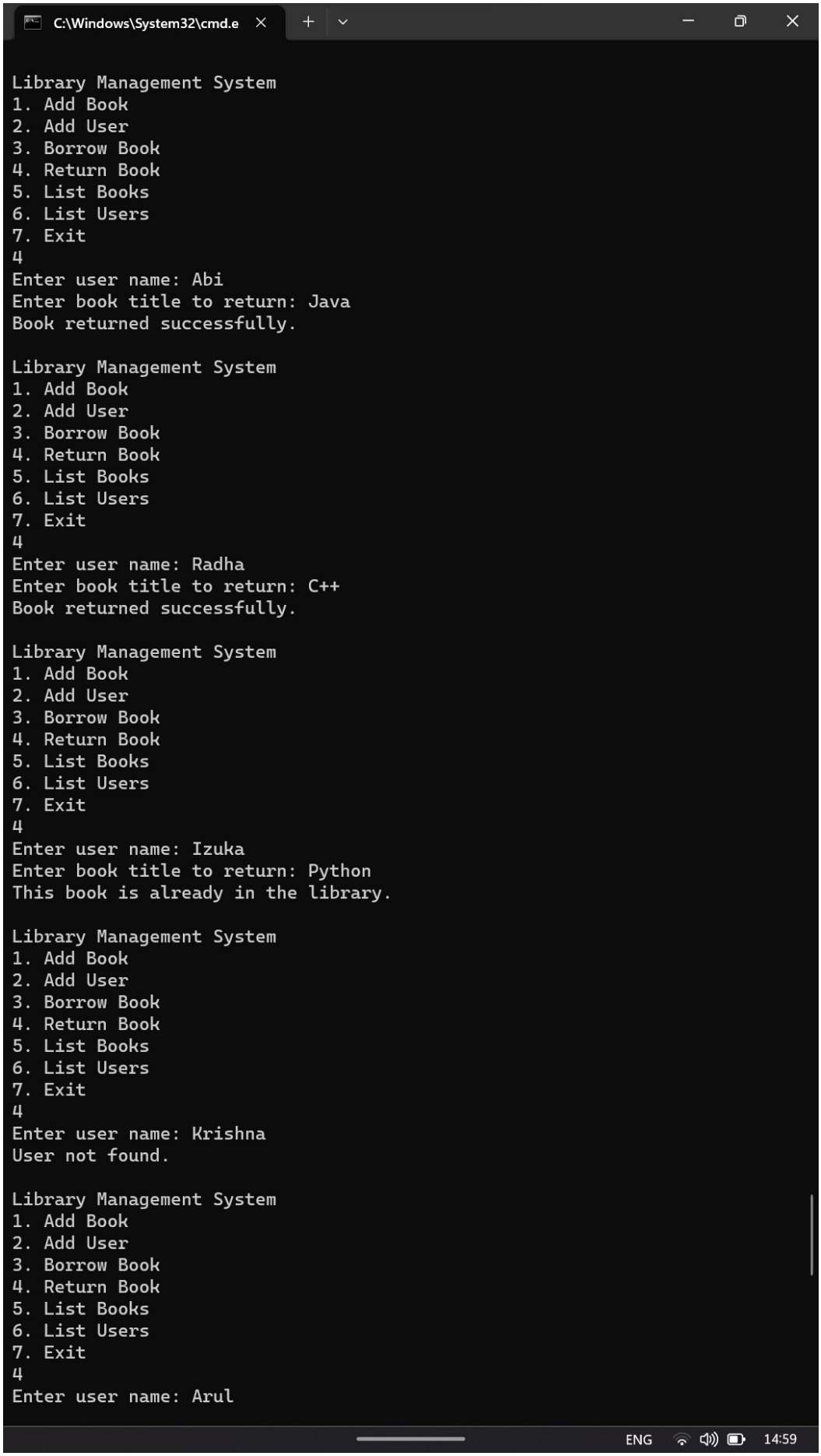

C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
5
List of Books:
Title: OOAD
Author: George
Available: true
Title: Java
Author: James Gosling
Available: true
Title: Python
Author: Guido Van Rossum
Available: true
Title: DBMS
Author: Mary
Available: true
Title: Computer Architecture
Author: Sangeet
Available: true
Title: C++
Author: David
Available: true
Title: HTML
Author: Binton
Available: true
Title: CSS
Author: Angelina Yuu
Available: true
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
6
List of Users:
User Name: Naveen
User Name: Bindu
User Name: Izuka
ENG
14:53

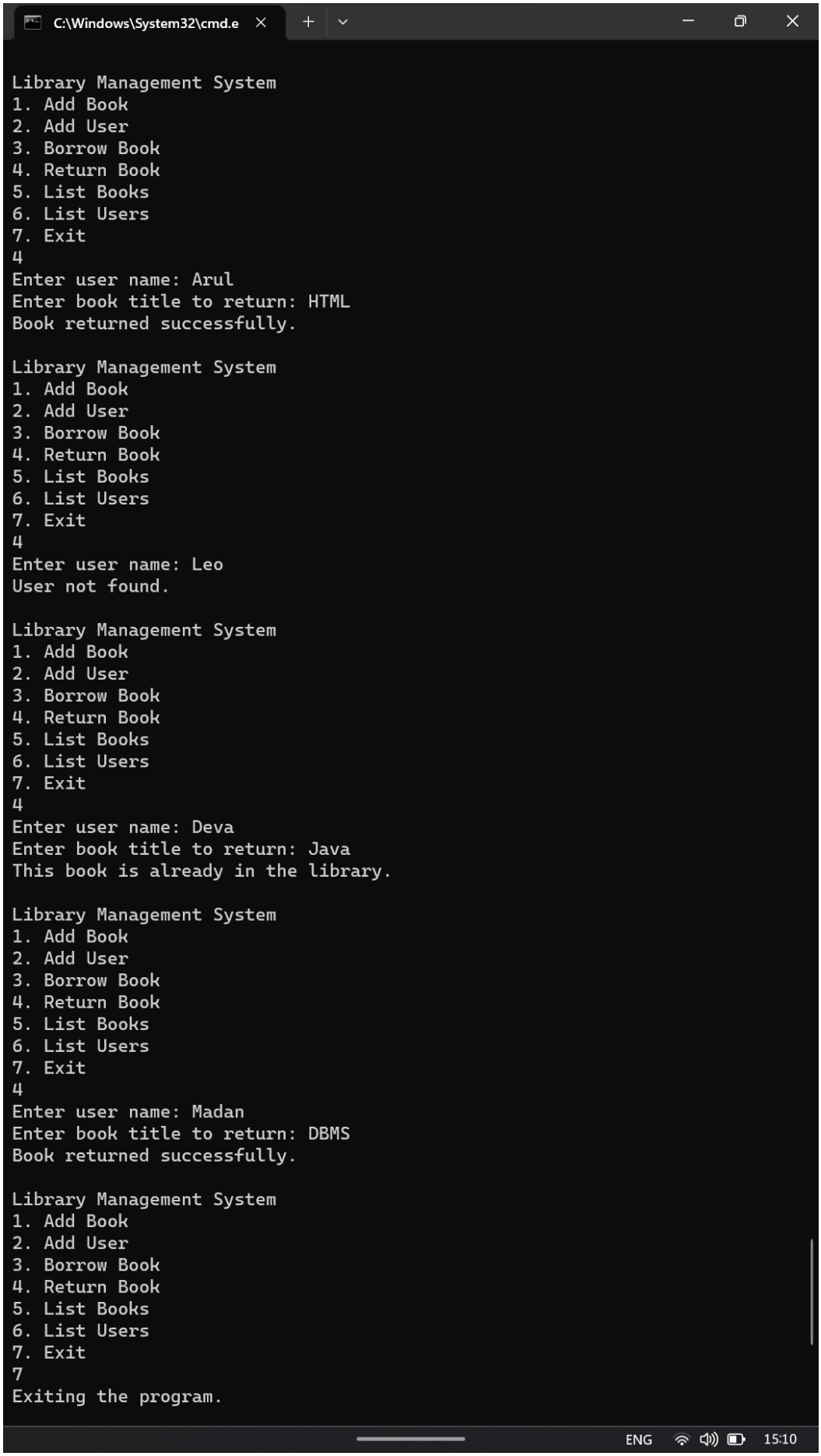

C:\Windows\System32\cmd.e
User Name: Krishna
User Name: Radha
User Name: Arul
User Name: Nalin
User Name: Deva
User Name: Sindu
User Name: Preethi
User Name: Ashwin
User Name: Rogi
User Name: Madan
User Name: Abi
User Name: Bhavani
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Radha
Enter book title to borrow: C++
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Rogi
Enter book title to borrow: HTML
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Naveen
Enter book title to borrow: OOAD
ENG
14:53


C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Naveen
Enter book title to borrow: OOAD
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Abi
Enter book title to borrow: Java
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Bindu
Enter book title to borrow: DBMS
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Sindu
Enter book title to borrow: C++
Book is not available for borrowing.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
ENG
14:54


C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Ashwin
Enter book title to borrow: CSS
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Preethi
Enter book title to borrow: HTML
Book is not available for borrowing.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Nalin
Enter book title to borrow: OOPS
Book not found.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
3
Enter user name: Radha
Enter book title to borrow: Computer Architecture
Book borrowed successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
5
ENG
14:54


C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
5
List of Books:
Title: OOAD
Author: George
Available: false
Title: Java
Author: James Gosling
Available: false
Title: Python
Author: Guido Van Rossum
Available: true
Title: DBMS
Author: Mary
Available: false
Title: Computer Architecture
Author: Sangeet
Available: false
Title: C++
Author: David
Available: false
Title: HTML
Author: Binton
Available: false
Title: CSS
Author: Angelina Yuu
Available: false
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
6
List of Users:
User Name: Naveen
User Name: Bindu
User Name: Izuka
ENG
14:55


C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Abi
Enter book title to return: Java
Book returned successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Radha
Enter book title to return: C++
Book returned successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Izuka
Enter book title to return: Python
This book is already in the library.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Krishna
User not found.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Arul
ENG
14:59


C:\Windows\System32\cmd.e
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Arul
Enter book title to return: HTML
Book returned successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Leo
User not found.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Deva
Enter book title to return: Java
This book is already in the library.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
4
Enter user name: Madan
Enter book title to return: DBMS
Book returned successfully.
Library Management System
1. Add Book
2. Add User
3. Borrow Book
4. Return Book
5. List Books
6. List Users
7. Exit
7
Exiting the program.
ENG
15:10