# Advanced Library Management System

### **Project Description:**

Create an advanced library management system that allows users to perform various operations such as adding, removing, searching, borrowing, and returning books. The system will also include user authentication, logging, reservations, due dates, fines, and reporting.

### **Data Structures:**

**Library**: A list of dictionaries, where each dictionary represents a book with keys: 'title', 'author', 'year', 'status' (available/borrowed), 'borrower' (if borrowed), 'due\_date' (if borrowed), and 'reservations' (list of users).

**Users**: A dictionary where each key is a username and the value is another dictionary with keys: 'password', 'role' (librarian/borrower), and 'borrowed\_books' (list of borrowed books).

**Logs**: A deque (double-ended queue) to store the last 100 log entries for auditing purposes.

### **Functions to Implement:**

**User Authentication and Management**

1. authenticate\_user(users, username, password):

* Authenticate the user and return the user role.
* Validate the username and password inputs.
* Handle errors and provide meaningful messages for invalid credentials.

2. add\_user(users, username, password, role):

* Add a new user to the system.
* Validate user inputs (e.g., non-empty username and password, valid role).
* Restrict this operation to librarian users only.
* Handle errors and provide meaningful messages for invalid inputs or duplicate usernames.

3. remove\_user(users, username):

* Remove a user from the system.
* Validate the username input.
* Restrict this operation to librarian users only.
* Handle errors and provide meaningful messages for invalid usernames or non-existent users.

**Book Management**

4. add\_book(library, title, author, year):

* Add a new book to the library.
* Validate book inputs (e.g., non-empty title and author, valid year).
* Restrict this operation to librarian users only.
* Handle errors and provide meaningful messages for invalid inputs or duplicate books.

5. remove\_book(library, title):

* Remove a book from the library by title.
* Validate the title input.
* Restrict this operation to librarian users only.
* Handle errors and provide meaningful messages for invalid titles or non-existent books.

6. check\_duplicate\_book(library, title, author):

* Check if a book with the same title and author already exists in the library.
* Handle errors and provide meaningful messages for duplicate books.

**Search and Listing**

7. search\_books(library, search\_term, search\_type):

* Search for books by title, author, or year.
* Validate the search term and search type inputs.
* Handle errors and provide meaningful messages for invalid inputs or no search results.

8. list\_books(library):

* List all books in the library.
* Handle errors and provide meaningful messages if the library is empty.

9. list\_borrowed\_books(library):

* List all borrowed books and their borrowers.
* Handle errors and provide meaningful messages if no books are borrowed.

**Borrowing and Returning**

10. borrow\_book(library, title, borrower, due\_date):

* Allow a user to borrow a book if it is available.
* Validate the title, borrower, and due date inputs.
* Handle errors and provide meaningful messages for invalid inputs, non-existent books, or unavailable books.

11. return\_book(library, title):

* Allow a user to return a borrowed book.
* Validate the title input.
* Handle errors and provide meaningful messages for invalid titles or non-borrowed books.

12. calculate\_fine(due\_date, return\_date):

* Calculate the fine based on the number of overdue days.
* Validate the due date and return date inputs.
* Handle errors and provide meaningful messages for invalid dates.

**Reservations**

13. reserve\_book(library, title, user):

* Allow the user to reserve a book and add them to the reservation list.
* Validate the title and user inputs.
* Handle errors and provide meaningful messages for invalid inputs, non-existent books, or duplicate reservations.

14. notify\_reservation(library, title):

* Notify the next user in the reservation list when a book becomes available.
* Handle errors and provide meaningful messages if no reservations exist.

**Logging and Auditing**

15. log\_operation(logs, operation, user, details):

* Log the operation with details and timestamp.
* Include user information in the logs.
* Maintain a log of all operations performed for auditing purposes.

**Data Persistence**

16. save\_library\_to\_file(library, filename):

* Save the current library data to a file.
* Validate the filename input.
* Handle errors and provide meaningful messages for file-related issues.

17. load\_library\_from\_file(filename):

* Load library data from a file.
* Validate the filename input.
* Handle errors and provide meaningful messages for file-related issues or data corruption.

**Reporting and Analytics**

18. generate\_report(library):

* Generate a report on library usage and borrowing trends.
* Handle errors and provide meaningful messages if the library is empty.

**Main Program**

Menu-Driven Interface:

Provide a user interface with options to:

1. Add a new book
2. Remove a book
3. Search for a book
4. List all books
5. Borrow a book
6. Return a book
7. Reserve a book
8. List borrowed books
9. Save library data
10. Load library data
11. Generate report
12. Add user
13. Remove user
14. Exit

### **Menu-Driven Interface Framework**

#### **Main Menu**

1. **Add a New Book**
2. **Remove a Book**
3. **Search for a Book**
4. **List All Books**
5. **Borrow a Book**
6. **Return a Book**
7. **Reserve a Book**
8. **List Borrowed Books**
9. **Save Library Data**
10. **Load Library Data**
11. **Generate Report**
12. **Add User**
13. **Remove User**
14. **Exit**

#### **Menu Options and Descriptions**

1. **Add a New Book**
   * **Prompt**: Enter the book title, author, and year of publication.
   * **Validation**: Ensure that the title and author are non-empty strings and the year is a valid integer.
   * **Permissions**: Restricted to librarian users.
   * **Logging**: Log the operation with details and timestamp.
2. **Remove a Book**
   * **Prompt**: Enter the book title to remove.
   * **Validation**: Ensure that the title is a non-empty string.
   * **Permissions**: Restricted to librarian users.
   * **Logging**: Log the operation with details and timestamp.
3. **Search for a Book**
   * **Prompt**: Enter the search term and select the search type (title, author, year).
   * **Validation**: Ensure that the search term is a non-empty string and the search type is valid.
   * **Logging**: Log the operation with details and timestamp.
4. **List All Books**
   * **Display**: List all books in the library with their details.
   * **Logging**: Log the operation with details and timestamp.
5. **Borrow a Book**
   * **Prompt**: Enter the book title and borrower username.
   * **Validation**: Ensure that the title and username are non-empty strings.
   * **Permissions**: Available to all users.
   * **Logging**: Log the operation with details and timestamp.
6. **Return a Book**
   * **Prompt**: Enter the book title to return.
   * **Validation**: Ensure that the title is a non-empty string.
   * **Permissions**: Available to all users.
   * **Logging**: Log the operation with details and timestamp.
7. **Reserve a Book**
   * **Prompt**: Enter the book title and username.
   * **Validation**: Ensure that the title and username are non-empty strings.
   * **Permissions**: Available to all users.
   * **Logging**: Log the operation with details and timestamp.
8. **List Borrowed Books**
   * **Display**: List all borrowed books with their borrowers and due dates.
   * **Logging**: Log the operation with details and timestamp.
9. **Save Library Data**
   * **Prompt**: Enter the filename to save the library data.
   * **Validation**: Ensure that the filename is a valid string.
   * **Permissions**: Restricted to librarian users.
   * **Logging**: Log the operation with details and timestamp.
10. **Load Library Data**
    * **Prompt**: Enter the filename to load the library data.
    * **Validation**: Ensure that the filename is a valid string.
    * **Permissions**: Restricted to librarian users.
    * **Logging**: Log the operation with details and timestamp.
11. **Generate Report**
    * **Display**: Generate and display a report on library usage and borrowing trends.
    * **Logging**: Log the operation with details and timestamp.
12. **Add User**
    * **Prompt**: Enter the username, password, and role (librarian/borrower).
    * **Validation**: Ensure that the username and password are non-empty strings and the role is valid.
    * **Permissions**: Restricted to librarian users.
    * **Logging**: Log the operation with details and timestamp.
13. **Remove User**
    * **Prompt**: Enter the username to remove.
    * **Validation**: Ensure that the username is a non-empty string.
    * **Permissions**: Restricted to librarian users.
    * **Logging**: Log the operation with details and timestamp.
14. **Exit**
    * **Action**: Exit the Library Management System.
    * **Logging**: Log the exit operation with timestamp.

### **Example of Menu-Driven Interface Flow**

1. **Display Main Menu**:
   * Show the list of options to the user.
   * Prompt the user to enter their choice.
2. **Handle User Choice**:
   * Based on the user's choice, prompt for additional inputs if necessary.
   * Validate the inputs and handle errors gracefully.
   * Perform the selected operation.
   * Log the operation with details and timestamp.
3. **Repeat**:
   * After completing an operation, return to the main menu.
   * Continue until the user chooses to exit.

### **Pseudocode for Main Program**

Plain text

function main():

initialize library, users, logs

while True:

display\_main\_menu()

choice = get\_user\_choice()

if choice == '1':

handle\_add\_book()

elif choice == '2':

handle\_remove\_book()

elif choice == '3':

handle\_search\_book()

elif choice == '4':

handle\_list\_books()

elif choice == '5':

handle\_borrow\_book()

elif choice == '6':

handle\_return\_book()

elif choice == '7':

handle\_reserve\_book()

elif choice == '8':

handle\_list\_borrowed\_books()

elif choice == '9':

handle\_save\_library\_data()

elif choice == '10':

handle\_load\_library\_data()

elif choice == '11':

handle\_generate\_report()

elif choice == '12':

handle\_add\_user()

elif choice == '13':

handle\_remove\_user()

elif choice == '14':

log\_exit\_operation()

break

else:

display\_invalid\_choice\_message()

function display\_main\_menu():

print("Library Management System")

print("1. Add a new book")

print("2. Remove a book")

print("3. Search for a book")

print("4. List all books")

print("5. Borrow a book")

print("6. Return a book")

print("7. Reserve a book")

print("8. List borrowed books")

print("9. Save library data")

print("10. Load library data")

print("11. Generate report")

print("12. Add user")

print("13. Remove user")

print("14. Exit")

function get\_user\_choice():

return input("Enter your choice: ")

function handle\_add\_book():

# Prompt for book details, validate inputs, add book, log operation

function handle\_remove\_book():

# Prompt for book title, validate input, remove book, log operation

function handle\_search\_book():

# Prompt for search term and type, validate inputs, search books, log operation

function handle\_list\_books():

# List all books, log operation

function handle\_borrow\_book():

# Prompt for book title and borrower, validate inputs, borrow book, log operation

function handle\_return\_book():

# Prompt for book title, validate input, return book, log operation

function handle\_reserve\_book():

# Prompt for book title and user, validate inputs, reserve book, log operation

function handle\_list\_borrowed\_books():

# List all borrowed books, log operation

function handle\_save\_library\_data():

# Prompt for filename, validate input, save data, log operation

function handle\_load\_library\_data():

# Prompt for filename, validate input, load data, log operation

function handle\_generate\_report():

# Generate and display report, log operation

function handle\_add\_user():

# Prompt for username, password, role, validate inputs, add user, log operation

function handle\_remove\_user():

# Prompt for username, validate input, remove user, log operation

function log\_exit\_operation():

# Log the exit operation with timestamp

function display\_invalid\_choice\_message():

print("Invalid choice. Please try again.")

This framework outlines the structure of the menu-driven interface, ensuring that all functionalities are accessible and user-friendly. It includes prompts for user inputs, validation, permissions, logging, and error handling to create a robust and comprehensive Library Management System.