LIBRARY MANAGEMENT

This Library Management System is implemented in Python and provides functionalities for managing books, users, and borrowing/returning operations. It is designed for both admin and user roles, with specific functionalities for each.

Code:

from datetime import datetime, timedelta

class Book:

def \_\_init\_\_(self, book\_id, title, author):

self.book\_id = book\_id

self.title = title

self.author = author

self.is\_available = True

self.borrowed\_by = None

self.due\_date = None

def \_\_str\_\_(self):

return f"{self.title} by {self.author}"

class User:

def \_\_init\_\_(self, user\_id, name, password, max\_books=3):

self.user\_id = user\_id

self.name = name

self.password = password

self.borrowed\_books = {}

self.max\_books = max\_books

def borrow\_book(self, book, borrow\_date):

if len(self.borrowed\_books) >= self.max\_books:

print(f"Sorry, {self.name} cannot borrow more than {self.max\_books} books.")

return

if book.is\_available:

book.is\_available = False

book.borrowed\_by = self

book.due\_date = borrow\_date + timedelta(days=7)

self.borrowed\_books[book.book\_id] = book

print(f"{self.name} borrowed {book.title}. Due date: {book.due\_date.date()}")

else:

print(f"Sorry, {book.title} is currently not available.")

def return\_book(self, book, return\_date):

if book.book\_id in self.borrowed\_books:

overdue\_days = (return\_date - book.due\_date).days

fine = max(0, overdue\_days) \* 1 # $1 per day fine for late return

self.borrowed\_books.pop(book.book\_id)

book.is\_available = True

book.borrowed\_by = None

book.due\_date = None

print(f"{self.name} returned {book.title}. Fine: ${fine}")

else:

print(f"{self.name} did not borrow {book.title}.")

def display\_borrowed\_books(self):

if self.borrowed\_books:

print(f"{self.name} has borrowed the following books:")

for book in self.borrowed\_books.values():

print(f"- {book.title} (Due Date: {book.due\_date.date()})")

else:

print(f"{self.name} has not borrowed any books.")

class Library:

def \_\_init\_\_(self):

self.books = {}

self.users = {}

self.admins = {"admin": "admin123"} # Admin credentials

def add\_book(self, book):

self.books[book.book\_id] = book

print(f"Book '{book.title}' added to the library.")

def delete\_book(self, book\_id):

if book\_id in self.books:

deleted\_book = self.books.pop(book\_id)

print(f"Book '{deleted\_book.title}' has been deleted from the library.")

else:

print("Book not found in the library.")

def add\_user(self, user):

self.users[user.user\_id] = user

print(f"User '{user.name}' added to the library system.")

def display\_all\_books(self):

if self.books:

print("Library Books:")

for book in self.books.values():

print(f"- {book.title} by {book.author} (Available: {'Yes' if book.is\_available else 'No'})")

else:

print("No books in the library.")

def display\_borrowed\_books(self):

print("Borrowed Books:")

for book in self.books.values():

if not book.is\_available:

print(f"- {book.title} by {book.author} (Borrowed by: {book.borrowed\_by.name}, Due: {book.due\_date.date()})")

def get\_date\_input(prompt):

while True:

date\_str = input(prompt)

try:

return datetime.strptime(date\_str, "%Y-%m-%d")

except ValueError:

print("Invalid date format. Please enter the date in YYYY-MM-DD format.")

# Main program

if \_\_name\_\_ == "\_\_main\_\_":

library = Library()

# Predefined users

library.add\_user(User("user1", "ravali", "ravali123"))

library.add\_user(User("user2", "chandana", "chandana456"))

# Predefined books

library.add\_book(Book(1, "1984", "George Orwell"))

library.add\_book(Book(2, "To Kill a Mockingbird", "Harper Lee"))

while True:

print("\nLibrary System:")

print("1. Admin Login")

print("2. User Login")

print("3. Exit")

choice = input("Enter your choice: ")

if choice == "1":

username = input("Enter admin username: ").strip()

password = input("Enter admin password: ").strip()

if username in library.admins and library.admins[username] == password:

print("Welcome, Admin!")

while True:

print("\nAdmin Menu:")

print("1. Add Book")

print("2. Delete Book")

print("3. Display All Books")

print("4. Display Borrowed Books")

print("5. Logout")

admin\_choice = input("Enter your choice: ")

if admin\_choice == "1":

title = input("Enter book title: ")

author = input("Enter book author: ")

book\_id = len(library.books) + 1

library.add\_book(Book(book\_id, title, author))

elif admin\_choice == "2":

book\_id = int(input("Enter book ID to delete: "))

library.delete\_book(book\_id)

elif admin\_choice == "3":

library.display\_all\_books()

elif admin\_choice == "4":

library.display\_borrowed\_books()

elif admin\_choice == "5":

print("Logging out...")

break

else:

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

else:

print("Invalid admin credentials.")

elif choice == "2":

user\_id = input("Enter your user ID: ").strip()

password = input("Enter your password: ").strip()

if user\_id in library.users and library.users[user\_id].password == password:

user = library.users[user\_id]

print(f"Welcome, {user.name}!")

while True:

print("\nUser Menu:")

print("1. Borrow Book")

print("2. Return Book")

print("3. Display Borrowed Books")

print("4. Logout")

user\_choice = input("Enter your choice: ")

if user\_choice == "1":

book\_id = int(input("Enter book ID to borrow: "))

if book\_id in library.books:

borrow\_date = datetime.now()

user.borrow\_book(library.books[book\_id], borrow\_date)

else:

print("Book not found.")

elif user\_choice == "2":

book\_id = int(input("Enter book ID to return: "))

if book\_id in library.books:

return\_date = datetime.now()

user.return\_book(library.books[book\_id], return\_date)

else:

print("Book not found.")

elif user\_choice == "3":

user.display\_borrowed\_books()

elif user\_choice == "4":

print("Logging out...")

break

else:

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

else:

print("Invalid user ID or password.")

elif choice == "3":

print("Exiting the system...")

break

else:

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

The program provides a menu-driven interface for admins and users.

**Admin Menu**

1. Add Book: Add a new book to the library.
2. Delete Book: Remove a book from the library by its ID.
3. Display All Books: View all books in the library.
4. Display Borrowed Books: View details of all borrowed books.
5. Logout: Exit the admin session.

**User Menu**

1. Borrow Book: Borrow a book by providing its ID.
2. Return Book: Return a previously borrowed book.
3. Display Borrowed Books: View all books currently borrowed by the user.
4. Logout: Exit the user session.

**Features**

* **Book Management**: Admins can add or delete books, and view the library's inventory.
* **User Management**: Admins can register new users.
* **Borrow and Return**:
  + Users can borrow books if they have not exceeded the borrowing limit.
  + Late returns incur a fine of $1 per day.
* **Security**: Separate login credentials for admins and users.
* **Date Handling**: Uses datetime to manage borrowing and returning dates.