NIET
Greater Noida
Autonomous Institute
Get Future Ready

2024 - 25

LIBRARY MANAGEMENT SYSTEM

UNDER THE SUPERVISION OF
MR. CHANDRAPAL SINGH ARYA
MR. ARCHIT GOEL

G13

ADVANCED PYTHON WORKSHOP

2024 - 25

PYTHON PROJECT REPORT

A PYTHON PROJECT SUBMITTED BY: CSE-DS(D)

BY : AMAN SINGH BHADORIYA (2401331540045)
DEENDAYAL (2401331540097)
KASHIF HUSSAIN (2401331540136)

UNDER THE SUPERVISION OF

MR. CHANDRAPAL SINGH ARYA
MR. ARCHIT GOEL

DEPARTMENT NAME: CSE (DS)
COLLEGE NAME:
NOIDA INSTITUTE OF ENGINEERING AND
TECHNOLOGY, GREATER NOIDA
(AN AUTONOMOUS INSTITUTE)
AFFILIATED TO
DR. A. P. J. ABDUL KALAM TECHNICAL UNIVERSITY,
LUCKNOW

JUNE 2024-2025

NIET
Greater Noida
Autonomous Institute
Get Future Ready

ACKNOWLEDGEMENT

WE WOULD LIKE TO EXPRESS OUR SINCERE GRATITUDE TO OUR
GUIDE MR. CHANDRAPAL SINGH ARYA ,
MR. ARCHIT GOEL FOR THEIR GUIDANCE AND SUPPORT IN HELPING ME COMPLETE THIS PROJECT
SUCCESSFULLY. WE ALSO THANK OUR PEERS FOR THEIR
CONSTANT MOTIVATION THROUGHOUT THIS JOURNEY.

NIET
Greater Noida
Autonomous Institute
Get Future Ready

INDEX

NIET
Greater Noida
Autonomous Institute
Get Future Ready

LIBRARY MANAGEMENT SYSTEM

1. INTRODUCTION

WELCOME TO OUR LIBRARY MANAGEMENT SYSTEM! THIS PROJECT SIMULATES A REAL-WORLD LIBRARY WHERE LIBRARIANS CAN MANAGE BOOKS, USERS, AND BORROWING OPERATIONS. IMAGINE WALKING INTO YOUR SCHOOL OR LOCAL LIBRARY - THIS SYSTEM REPLICATES THOSE CORE FUNCTIONS DIGITALLY.

THINK OF IT AS A VIRTUAL LIBRARIAN THAT NEVER SLEEPS:

- KEEPS TRACK OF ALL BOOKS AND THEIR AVAILABILITY
- MANAGES USER REGISTRATIONS AND BORROWING HISTORY
- AUTOMATICALLY CALCULATES LATE FEES FOR OVERDUE BOOKS
- MAINTAINS RECORDS EVEN WHEN THE SYSTEM IS TURNED OFF

BUILT WITH PYTHON, THIS SYSTEM DEMONSTRATES KEY PROGRAMMING CONCEPTS IN A PRACTICAL, REAL-WORLD APPLICATION THAT COULD ACTUALLY BE USED IN SMALL LIBRARIES OR BOOK CLUBS.

NIET
Greater Noida
Autonomous Institute
Get Future Ready

PROJECT OBJECTIVES

OUR SYSTEM WAS DESIGNED TO ACHIEVE THESE SPECIFIC GOALS:

CORE FUNCTIONS:

- 📚 ORGANIZE BOOKS WITH TITLES, AUTHORS, AND UNIQUE ISBNS
- 👥 MANAGE DIFFERENT USER TYPES (STUDENTS, FACULTY, REGULAR USERS)
- 🔄 HANDLE BOOK BORROWING AND RETURNING OPERATIONS
- ⏰ AUTOMATICALLY TRACK DUE DATES AND CALCULATE LATE FEES

TECHNICAL GOALS:

- ✅ IMPLEMENT OBJECT-ORIENTED PROGRAMMING PRINCIPLES
- 💾 SAVE ALL DATA BETWEEN SESSIONS USING JSON FILES
- 📝 MAINTAIN DETAILED TRANSACTION LOGS IN CSV FORMAT
- 🛠️ CREATE AN INTUITIVE CONSOLE INTERFACE

LEARNING OUTCOMES:

- UNDERSTAND CLASS RELATIONSHIPS AND INHERITANCE
- PRACTICE FILE HANDLING AND DATA PERSISTENCE
- IMPLEMENT REAL-WORLD ERROR HANDLING
- DESIGN COMPLETE SOFTWARE SYSTEMS FROM SCRATCH

NIET
Greater Noida
Autonomous Institute
Get Future Ready

FEATURES AND FUNCTIONALITY

OUR SYSTEM OFFERS THESE KEY FEATURES:

BOOK MANAGEMENT:

- ADD NEW BOOKS TO THE COLLECTION
- REMOVE BOOKS WHEN THEY'RE RETIRED
- SEARCH BOOKS BY TITLE, AUTHOR OR ISBN
- VIEW ALL BOOKS OR ONLY AVAILABLE ONES

USER MANAGEMENT:

- REGISTER NEW USERS WITH DIFFERENT TYPES
- REMOVE USERS WHO LEAVE THE SYSTEM
- DIFFERENT BORROWING RULES PER USER TYPE:
  - STUDENTS: 5 BOOKS FOR 21 DAYS
  - FACULTY: 10 BOOKS FOR 90 DAYS
  - REGULAR: 3 BOOKS FOR 14 DAYS

BORROWING SYSTEM:

- BORROW BOOKS WITH AUTOMATIC DUE DATE CALCULATION
- RETURN BOOKS WITH LATE FEE CALCULATION ($0.50/DAY)
- PREVENT OVER-BORROWING BEYOND LIMITS
- VIEW BOOKS BORROWED BY SPECIFIC USERS

REPORTING:

- CURRENTLY BORROWED BOOKS REPORT
- OVERDUE BOOKS WITH DAYS OVERDUE
- DETAILED TRANSACTION HISTORY
- AUTOMATIC DATA BACKUP TO FILES

NIET
Greater Noida
Autonomous Institute
Get Future Ready

TOOLS AND RESOURCES USED

LEARNING RESOURCES:

- PYTHON OFFICIAL DOCUMENTATION: LEARNING RESOURCES:
- REAL PYTHON OOP TUTORIAL: LEARNING RESOURCES:
- PYTHON JSON HANDLING GUIDE: LEARNING RESOURCES:

TECHNICAL INSPIRATION:

- LIBRARY OF CONGRESS SYSTEM OVERVIEW
- OPEN-SOURCE LIBRARY SYSTEMS LIKE KOHA
- UNIVERSITY LIBRARY MANAGEMENT PRACTICES

PROJECT TOOLS:

- PYTHON 3.X PROGRAMMING LANGUAGE
- VISUAL STUDIO CODE EDITOR
- GIT FOR VERSION CONTROL
- PYTHON DATETIME, JSON, OS, AND CSV MODULES

PROJECT LINK -

https://github.com/amansingh-develops/Library_Management_System

CODE SNIPPET

File - C:\Users\Aman Singh\OneDrive - Noida Institute of Engineering and Technology\Desktop\python project\Library_manage

```
import json
import os
import csv
from datetime import datetime, timedelta
from abc import ABC, abstractmethod

# Constants
BOOKS_FILE = 'books.json'
USERS_FILE = 'users.json'
TRANSACTION_LOG = 'library_log.csv'
DATE_FORMAT = "%Y-%m-%d"
SESSION_END_DATE = "2023-12-31"  # Example session end date

# Book Class
class Book:
    def __init__(self, title: str, author: str, isbn: str):
        self._title = title
        self._author = author
        self._isbn = isbn
        self._is_borrowed = False
        self._due_date = None
        self._borrower_id = None
        self._borrower_name = None

    @property
    def title(self):
        return self._title

    @property
    def author(self):
        return self._author

    @property
    def isbn(self):
        return self._isbn

    @property
    def is_borrowed(self):
        return self._is_borrowed

    @is_borrowed.setter
    def is_borrowed(self, value):
```

```
43          if isinstance(value, bool):
44              self._is_borrowed = value
45
46      @property
47      def due_date(self):
48          return self._due_date
49
50      @due_date.setter
51      def due_date(self, value):
52          self._due_date = value
53
54      @property
55      def borrower_id(self):
56          return self._borrower_id
57
58      @borrower_id.setter
59      def borrower_id(self, value):
60          self._borrower_id = value
61
62      @property
63      def borrower_name(self):
64          return self._borrower_name
65
66      @borrower_name.setter
67      def borrower_name(self, value):
68          self._borrower_name = value
69
70      def borrow(self, days=14, borrower_id=None,
    borrower_name=None) -> bool:
71          if not self._is_borrowed:
72              self._is_borrowed = True
73              self._due_date = (datetime.now() + timedelta(
    days=days)).strftime(DATE_FORMAT)
74              self._borrower_id = borrower_id
75              self._borrower_name = borrower_name
76              return True
77          return False
78
79      def return_book(self) -> bool:
80          if self._is_borrowed:
81              self._is_borrowed = False
82              self._due_date = None
83              self._borrower_id = None
84              self._borrower_name = None
```

```
            return True
        return False

    def get_days_remaining(self):
        """Calculate days remaining until due date"""
        if not self._due_date or not self._is_borrowed:
            return None
        due = datetime.strptime(self._due_date, DATE_FORMAT)
        remaining = (due - datetime.now()).days
        return max(0, remaining)  # Return 0 if overdue

    def __str__(self):
        status = "Borrowed" if self._is_borrowed else "Available"
        if self._is_borrowed:
            days_remaining = self.get_days_remaining()
            days_text = f"{days_remaining} days remaining" if days_remaining is not None else "Due date not set"
            return (f"Title: {self._title}, Author: {self._author}, ISBN: {self._isbn}, "
                    f"Status: {status} by {self._borrower_name} (Due: {self._due_date}, {days_text})")
        return f"Title: {self._title}, Author: {self._author}, ISBN: {self._isbn}, Status: {status}"

    def to_dict(self):
        return {
            "title": self._title,
            "author": self._author,
            "isbn": self._isbn,
            "is_borrowed": self._is_borrowed,
            "due_date": self._due_date,
            "borrower_id": self._borrower_id,
            "borrower_name": self._borrower_name
        }

# Abstract User Class
class AbstractUser(ABC):
    @abstractmethod
    def get_borrowing_limit(self):
        pass

    @abstractmethod
```

```
123     def get_borrowing_period(self):
124         pass
125 
126 # User Class
127 class User(AbstractUser):
128     def __init__(self, name: str, user_id: str):
129         self._name = name
130         self._user_id = user_id
131         self._borrowed_books_isbns = []
132 
133     @property
134     def name(self):
135         return self._name
136 
137     @property
138     def user_id(self):
139         return self._user_id
140 
141     @property
142     def borrowed_books_isbns(self):
143         return self._borrowed_books_isbns.copy()
144 
145     def get_borrowing_limit(self):
146         return 3  # Default limit
147 
148     def get_borrowing_period(self):
149         return 14  # Default period
150 
151     def add_borrowed_book_isbn(self, isbn: str) -> None:
152         if isbn not in self._borrowed_books_isbns:
153             self._borrowed_books_isbns.append(isbn)
154 
155     def remove_borrowed_book_isbn(self, isbn: str) ->
    None:
156         if isbn in self._borrowed_books_isbns:
157             self._borrowed_books_isbns.remove(isbn)
158 
159     def __str__(self):
160         return f"User: {self._name} (ID: {self._user_id}
    ), Borrowed Books: {len(self._borrowed_books_isbns)}"
161 
162     def to_dict(self):
163         return {
164             "name": self._name,
```

```
            "user_id": self._user_id,
            "borrowed_books_isbns": self._borrowed_books_isbns,
            "type": self.__class__.__name__
        }

# Student Class
class Student(User):
    def get_borrowing_limit(self):
        return 5

    def get_borrowing_period(self):
        return 21

# Faculty Class
class Faculty(User):
    def get_borrowing_limit(self):
        return 10

    def get_borrowing_period(self):
        return 90

# Library Class
class Library:
    def __init__(self, book_file=BOOKS_FILE, user_file=USERS_FILE):
        self._books = {}
        self._users = {}
        self._data_file_books = book_file
        self._data_file_users = user_file
        self._load_data()

    def _load_data(self):
        # Load books
        try:
            if os.path.exists(self._data_file_books):
                with open(self._data_file_books, 'r') as f:
                    books_data = json.load(f)
                    for book_dict in books_data:
                        book = Book(
                            book_dict['title'],
                            book_dict['author'],
                            book_dict['isbn']
```

```
                        )
                        book.is_borrowed = book_dict['is_borrowed']
                        book.due_date = book_dict.get('due_date')
                        book.borrower_id = book_dict.get('borrower_id')
                        book.borrower_name = book_dict.get('borrower_name')
                        self._books[book.isbn] = book
        except Exception as e:
            print(f"Error loading books: {e}")

        # Load users
        try:
            if os.path.exists(self._data_file_users):
                with open(self._data_file_users, 'r') as f:
                    users_data = json.load(f)
                    for user_dict in users_data:
                        user_type = user_dict.get('type', 'User')
                        if user_type == 'Student':
                            user = Student(user_dict['name'], user_dict['user_id'])
                        elif user_type == 'Faculty':
                            user = Faculty(user_dict['name'], user_dict['user_id'])
                        else:
                            user = User(user_dict['name'], user_dict['user_id'])
                        user._borrowed_books_isbns = user_dict['borrowed_books_isbns']
                        self._users[user.user_id] = user
        except Exception as e:
            print(f"Error loading users: {e}")

    def _save_data(self):
        # Save books
        try:
            books_data = [book.to_dict() for book in self._books.values()]
            with open(self._data_file_books, 'w') as f:
                json.dump(books_data, f)
```

```
        except Exception as e:
            print(f"Error saving books: {e}")

        # Save users
        try:
            users_data = [user.to_dict() for user in self._users.values()]
            with open(self._data_file_users, 'w') as f:
                json.dump(users_data, f)
        except Exception as e:
            print(f"Error saving users: {e}")

    def _log_transaction(self, action, user_id="", isbn="", details="", status="Success"):
        log_entry = {
            "timestamp": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
            "action": action,
            "user_id": user_id,
            "isbn": isbn,
            "details": details,
            "status": status
        }

        try:
            file_exists = os.path.isfile(TRANSACTION_LOG)
            with open(TRANSACTION_LOG, 'a', newline='') as f:
                writer = csv.DictWriter(f, fieldnames=log_entry.keys())
                if not file_exists:
                    writer.writeheader()
                writer.writerow(log_entry)
        except Exception as e:
            print(f"Error logging transaction: {e}")

    def add_book(self, title, author, isbn):
        if isbn in self._books:
            self._log_transaction("Add Book", isbn=isbn, details="ISBN already exists", status="Fail")
            return False

        self._books[isbn] = Book(title, author, isbn)
        self._save_data()
```

```
277         self._log_transaction("Add Book", isbn=isbn,
    details=f"Added: {title} by {author}")
278         return True
279 
280     def remove_book(self, isbn):
281         if isbn not in self._books:
282             self._log_transaction("Remove Book", isbn=
    isbn, details="Book not found", status="Fail")
283             return False
284 
285         if self._books[isbn].is_borrowed:
286             self._log_transaction("Remove Book", isbn=
    isbn, details="Book is borrowed", status="Fail")
287             return False
288 
289         del self._books[isbn]
290         self._save_data()
291         self._log_transaction("Remove Book", isbn=isbn,
    details="Book removed")
292         return True
293 
294     def register_user(self, name, user_id, user_type="
    regular"):
295         if user_id in self._users:
296             self._log_transaction("Register User",
    user_id=user_id, details="User ID already exists", status
    ="Fail")
297             return False
298 
299         if user_type.lower() == "student":
300             self._users[user_id] = Student(name, user_id)
301         elif user_type.lower() == "faculty":
302             self._users[user_id] = Faculty(name, user_id)
303         else:
304             self._users[user_id] = User(name, user_id)
305 
306         self._save_data()
307         self._log_transaction("Register User", user_id=
    user_id, details=f"Registered: {name} ({user_type})")
308         return True
309 
310     def remove_user(self, user_id):
311         if user_id not in self._users:
312             self._log_transaction("Remove User", user_id=
```

```
312 user_id, details="User not found", status="Fail")
313             return False
314 
315         if self._users[user_id].borrowed_books_isbns:
316             self._log_transaction("Remove User", user_id=
    user_id, details="User has borrowed books", status="Fail"
    )
317             return False
318 
319         del self._users[user_id]
320         self._save_data()
321         self._log_transaction("Remove User", user_id=
    user_id, details="User removed")
322         return True
323 
324     def borrow_book(self, isbn, user_id, period_choice):
325         """Borrow a book with flexible period options"""
326         # Validate book and user
327         if isbn not in self._books:
328             self._log_transaction("Borrow Book", user_id
    , isbn, "Book not found", "Fail")
329             return False
330 
331         if user_id not in self._users:
332             self._log_transaction("Borrow Book", user_id
    , isbn, "User not found", "Fail")
333             return False
334 
335         book = self._books[isbn]
336         user = self._users[user_id]
337 
338         # Check borrowing constraints
339         if book.is_borrowed:
340             self._log_transaction("Borrow Book", user_id
    , isbn, "Book already borrowed", "Fail")
341             return False
342 
343         if len(user.borrowed_books_isbns) >= user.
    get_borrowing_limit():
344             self._log_transaction("Borrow Book", user_id
    , isbn, "Borrowing limit reached", "Fail")
345             return False
346 
347         # Map period choice to days
```

```
348         period_options = {
349             '1': 10,    # 10 days
350             '2': 20,    # 20 days
351             '3': 30,    # 30 days
352             '4': (datetime.strptime(SESSION_END_DATE,
    DATE_FORMAT) - datetime.now()).days  # Whole session
353         }
354 
355         days = period_options.get(period_choice, user.
    get_borrowing_period())
356 
357         # Process borrowing
358         if book.borrow(days=days, borrower_id=user_id,
    borrower_name=user.name):
359             user.add_borrowed_book_isbn(isbn)
360             self._save_data()
361 
362             details = f"Period: {days} days, Due: {book.
    due_date}"
363             self._log_transaction("Borrow Book", user_id
    , isbn, details)
364             return True
365         return False
366 
367     def return_book(self, isbn, user_id):
368         # Validate book and user
369         if isbn not in self._books:
370             self._log_transaction("Return Book", user_id
    , isbn, "Book not found", "Fail")
371             return False, 0.0
372 
373         if user_id not in self._users:
374             self._log_transaction("Return Book", user_id
    , isbn, "User not found", "Fail")
375             return False, 0.0
376 
377         book = self._books[isbn]
378         user = self._users[user_id]
379 
380         # Check if book is borrowed by this user
381         if isbn not in user.borrowed_books_isbns:
382             self._log_transaction("Return Book", user_id
    , isbn, "User didn't borrow this book", "Fail")
383             return False, 0.0
```

```
384
385         # Calculate late fee if applicable
386         late_fee = 0.0
387         if book.due_date:
388             due_date = datetime.strptime(book.due_date,
    DATE_FORMAT)
389             if datetime.now() > due_date:
390                 days_late = (datetime.now() - due_date).
    days
391                 late_fee = days_late * 0.50  # $0.50 per
    day
392
393         # Process return
394         if book.return_book():
395             user.remove_borrowed_book_isbn(isbn)
396             self._save_data()
397
398             details = f"Late fee: ${late_fee:.2f}" if
    late_fee else "Returned on time"
399             self._log_transaction("Return Book", user_id
    , isbn, details)
400             return True, late_fee
401         return False, 0.0
402
403     def search_book(self, query):
404         results = []
405         query = query.lower()
406
407         for book in self._books.values():
408             if (query in book.title.lower() or
409                 query in book.author.lower() or
410                 query in book.isbn.lower()):
411                 results.append(book)
412
413         return results
414
415     def display_all_books(self, show_available_only=False
    ):
416         print("\n===== BOOK LIST =====")
417         for book in self._books.values():
418             if not show_available_only or not book.
    is_borrowed:
419                 print(book)
420         print("======================")
```

```
421
422     def display_all_users(self):
423         print("\n===== USER LIST =====")
424         for user in self._users.values():
425             print(user)
426         print("======================")
427
428     def display_user_borrowed_books(self, user_id):
429         if user_id not in self._users:
430             print("User not found!")
431             return
432
433         user = self._users[user_id]
434         print(f"\nBorrowed books by {user.name} (ID: {
    user_id}):")
435
436         if not user.borrowed_books_isbns:
437             print(" - No books borrowed")
438             return
439
440         for isbn in user.borrowed_books_isbns:
441             if isbn in self._books:
442                 book = self._books[isbn]
443                 days_remaining = book.get_days_remaining
    ()
444                 days_text = f"{days_remaining} days
    remaining" if days_remaining is not None else "Due date
    not set"
445                 print(f" - {book.title} (ISBN: {isbn}),
    Due: {book.due_date}, {days_text}")
446             else:
447                 print(f" - Book not found (ISBN: {isbn})"
    )
448
449     def generate_report(self, report_type):
450         if report_type == "borrowed":
451             print("\n===== CURRENTLY BORROWED BOOKS
     =====")
452             for book in self._books.values():
453                 if book.is_borrowed:
454                     days_remaining = book.
    get_days_remaining()
455                     days_text = f"{days_remaining} days
    remaining" if days_remaining is not None else "Due date
```

```
not set"
                    print(f"{book.title} by {book.author}
 (ISBN: {book.isbn})")
                    print(f"  Borrowed by: {book.
borrower_name} (ID: {book.borrower_id})")
                    print(f"  Due: {book.due_date}, {
days_text}\n")
            print("====================================")

        elif report_type == "overdue":
            print("\n===== OVERDUE BOOKS =====")
            today = datetime.now()
            for book in self._books.values():
                if book.is_borrowed and book.due_date:
                    due_date = datetime.strptime(book.
due_date, DATE_FORMAT)
                    if today > due_date:
                        days_overdue = (today - due_date
).days
                        print(f"{book.title} by {book.
author} (ISBN: {book.isbn})")
                        print(f"  Borrowed by: {book.
borrower_name} (ID: {book.borrower_id})")
                        print(f"  Due: {book.due_date},
Overdue by {days_overdue} days\n")
            print("============================")

# Console Interface
def main():
    library = Library()

    while True:
        print("\n=====__________ LIBRARY MANAGEMENT
SYSTEM __________=====")
        print("1. Add Book                              2.
Remove Book ")
        print("3. Register User                         4.
Remove User")
        print("5. Borrow Book                           6.
Return Book")
        print("7. Search Books                          8.
Display All Books")
        print("9. Display Available Books               10
. Display All Users")
```

```
485         print("11. View User's Borrowed Books         12
    . Generate Reports")
486         print(
    "
        ")
487         print("                          13. Exit
                              ")
488
489         try:
490             choice = int(input("Enter your choice: "))
491         except ValueError:
492             print("Invalid input! Please enter a number."
    )
493             continue
494
495         if choice == 1:  # Add Book
496             title = input("Enter book title: ")
497             author = input("Enter author: ")
498             isbn = input("Enter ISBN: ")
499             if library.add_book(title, author, isbn):
500                 print("Book added successfully!")
501             else:
502                 print("Failed to add book. ISBN might
    already exist.")
503
504         elif choice == 2:  # Remove Book
505             isbn = input("Enter ISBN of book to remove: "
    )
506             if library.remove_book(isbn):
507                 print("Book removed successfully!")
508             else:
509                 print("Failed to remove book. It might
    not exist or is currently borrowed.")
510
511         elif choice == 3:  # Register User
512             name = input("Enter user name: ")
513             user_id = input("Enter user ID: ")
514             user_type = input("Enter user type (regular/
    student/faculty): ").lower()
515             if library.register_user(name, user_id,
    user_type):
516                 print("User registered successfully!")
517             else:
518                 print("Failed to register user. User ID
```

```
might already exist.")

        elif choice == 4:  # Remove User
            user_id = input("Enter user ID to remove: ")
            if library.remove_user(user_id):
                print("User removed successfully!")
            else:
                print("Failed to remove user. User might not exist or has borrowed books.")

        elif choice == 5:  # Borrow Book
            user_id = input("Enter user ID: ")
            isbn = input("Enter ISBN of book to borrow: ")

            # Show borrowing period options
            print("\nSelect borrowing period:")
            print("1. 10 days")
            print("2. 20 days")
            print("3. 30 days")
            print(f"4. Whole session (until {SESSION_END_DATE})")
            period_choice = input("Enter your choice (1-4): ")

            if library.borrow_book(isbn, user_id, period_choice):
                print("Book borrowed successfully!")
                # Show due date
                book = library._books.get(isbn)
                if book and book.due_date:
                    print(f"Due date: {book.due_date}")
            else:
                print("Failed to borrow book. Check if book/user exists, book is available, or user hasn't reached borrowing limit.")

        elif choice == 6:  # Return Book
            user_id = input("Enter user ID: ")
            isbn = input("Enter ISBN of book to return: ")
            success, late_fee = library.return_book(isbn, user_id)
            if success:
```

```
                if late_fee > 0:
                    print(f"Book returned successfully! Late fee: ${late_fee:.2f}")
                else:
                    print("Book returned successfully!")
            else:
                print("Failed to return book. Check if book/user exists or user didn't borrow this book.")

        elif choice == 7:  # Search Books
            query = input("Enter search query (title, author, or ISBN): ")
            results = library.search_book(query)
            if results:
                print("\nSearch Results:")
                for book in results:
                    print(book)
            else:
                print("No books found matching your query.")

        elif choice == 8:  # Display All Books
            library.display_all_books()

        elif choice == 9:  # Display Available Books
            library.display_all_books(show_available_only=True)

        elif choice == 10:  # Display All Users
            library.display_all_users()

        elif choice == 11:  # View User's Borrowed Books
            user_id = input("Enter user ID: ")
            library.display_user_borrowed_books(user_id)

        elif choice == 12:  # Generate Reports
            print("\nReport Types:")
            print("1. Currently Borrowed Books")
            print("2. Overdue Books")
            try:
                report_choice = int(input("Enter report choice: "))
                if report_choice == 1:
                    library.generate_report("borrowed")
```

```
                elif report_choice == 2:
                    library.generate_report("overdue")
                else:
                    print("Invalid choice!")
            except ValueError:
                print("Invalid input!")

        elif choice == 13:  # Exit
            print("Exiting the system. Goodbye!")
            break

        else:
            print("Invalid choice! Please try again.")

if __name__ == "__main__":
    main()
```

OUTPUT

File - Library_management_system

```
"C:¥Users¥Aman Singh¥PycharmProjects¥Advanced-python-workshop¥.venv¥Scripts¥python.exe" "C:¥Users¥Aman Singh¥OneDrive - Noida Institute of Engineering and Technology¥Desktop¥python project¥Library_management_system.py"
Error loading users: Extra data: line 1 column 90 (char 89)

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                  13. Exit
Enter your choice: 1
Enter book title: MY SQL
Enter author: AR HADDY
Enter ISBN: 004A
Book added successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                  13. Exit
Enter your choice: 2
Enter ISBN of book to remove: 004A
Book removed successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                  13. Exit
Enter your choice: 3
```

```
Enter user name: DEENDAYAL
Enter user ID: 0241CSDS255
Enter user type (regular/student/faculty): STUDENT
User registered successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                    13. Exit
Enter your choice: 4
Enter user ID to remove: 0241CSDS255
User removed successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                    13. Exit
Enter your choice: 5
Enter user ID: 0241CSDS218
Enter ISBN of book to borrow: 001A

Select borrowing period:
1. 10 days
2. 20 days
3. 30 days
4. Whole session (until 2023-12-31)
Enter your choice (1-4): 1
Failed to borrow book. Check if book/user exists, book is available, or user
hasn't reached borrowing limit.

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
```

```
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

                13. Exit
Enter your choice: 3
Enter user name: AMAN SINGH BHADORIYA
Enter user ID: 0241CSDS218
Enter user type (regular/student/faculty): STUDENT
User registered successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                     2. Remove Book
3. Register User                4. Remove User
5. Borrow Book                  6. Return Book
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

                13. Exit
Enter your choice: 5
Enter user ID: 0241CSDS218
Enter ISBN of book to borrow: 001A

Select borrowing period:
1. 10 days
2. 20 days
3. 30 days
4. Whole session (until 2023-12-31)
Enter your choice (1-4): 1
Book borrowed successfully!
Due date: 2025-06-29

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                     2. Remove Book
3. Register User                4. Remove User
5. Borrow Book                  6. Return Book
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

                13. Exit
Enter your choice: 6
Enter user ID: 0241CSDS218
```

```
Enter ISBN of book to return: 001A
Book returned successfully!

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books     10. Display All Users
11. View User's Borrowed Books 12. Generate Reports

                13. Exit
Enter your choice: 7
Enter search query (title, author, or ISBN): physics

Search Results:
Title: engineering physics, Author: naval gupta, ISBN: 002A, Status: Available

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books     10. Display All Users
11. View User's Borrowed Books 12. Generate Reports

                13. Exit
Enter your choice: 5
Enter user ID: 0241CSDS218
Enter ISBN of book to borrow: 002A

Select borrowing period:
1. 10 days
2. 20 days
3. 30 days
4. Whole session (until 2023-12-31)
Enter your choice (1-4): 2
Book borrowed successfully!
Due date: 2025-07-09

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
```

```
5. Borrow Book                  6. Return Book
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

                13. Exit
Enter your choice: 8

===== BOOK LIST =====
Title: engineering mathematics, Author: RS AGRAWALL, ISBN: 001A, Status: Available
Title: engineering physics, Author: naval gupta, ISBN: 002A, Status: Borrowed by AMAN SINGH BHADORIYA (Due: 2025-07-09, 19 days remaining)
Title: engineering chemistry, Author: H.A Reddy, ISBN: 003A, Status: Available
======================

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                     2. Remove Book
3. Register User                4. Remove User
5. Borrow Book                  6. Return Book
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

                13. Exit
Enter your choice: 9

===== BOOK LIST =====
Title: engineering mathematics, Author: RS AGRAWALL, ISBN: 001A, Status: Available
Title: engineering chemistry, Author: H.A Reddy, ISBN: 003A, Status: Available
======================

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                     2. Remove Book
3. Register User                4. Remove User
5. Borrow Book                  6. Return Book
7. Search Books                 8. Display All Books
9. Display Available Books      10. Display All Users
11. View User's Borrowed Books  12. Generate Reports

```

```
                    13. Exit
Enter your choice: 10

===== USER LIST =====
User: AMAN SINGH BHADORIYA (ID: 0241CSDS218), Borrowed Books: 1
======================

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                    13. Exit
Enter your choice: 11
Enter user ID: 0241CSDS218

Borrowed books by AMAN SINGH BHADORIYA (ID: 0241CSDS218):
 - engineering physics (ISBN: 002A), Due: 2025-07-09, 19 days remaining

=====_________ LIBRARY MANAGEMENT SYSTEM _________=====
1. Add Book                    2. Remove Book
3. Register User               4. Remove User
5. Borrow Book                 6. Return Book
7. Search Books                8. Display All Books
9. Display Available Books          10. Display All Users
11. View User's Borrowed Books      12. Generate Reports

                    13. Exit
Enter your choice: 12

Report Types:
1. Currently Borrowed Books
2. Overdue Books
Enter report choice: 1

===== CURRENTLY BORROWED BOOKS =====
engineering physics by naval gupta (ISBN: 002A)
  Borrowed by: AMAN SINGH BHADORIYA (ID: 0241CSDS218)
  Due: 2025-07-09, 19 days remaining

====================================
```

```
254
255  =====_________ LIBRARY MANAGEMENT SYSTEM _________=====
256  1. Add Book                    2. Remove Book
257  3. Register User               4. Remove User
258  5. Borrow Book                 6. Return Book
259  7. Search Books                8. Display All Books
260  9. Display Available Books     10. Display All Users
261  11. View User's Borrowed Books 12. Generate Reports
262
263                    13. Exit
264  Enter your choice: 13
265  Exiting the system. Goodbye!
266
267  Process finished with exit code 0
268
```

NIET
Greater Noida
Autonomous Institute
Get Future Ready

CONCLUSION

THIS LIBRARY MANAGEMENT SYSTEM DEMONSTRATES HOW PROGRAMMING CONCEPTS TRANSLATE INTO REAL-WORLD SOLUTIONS. BY COMPLETING THIS PROJECT, WE'VE GAINED:

1. PRACTICAL OOP SKILLS: LEARNED TO DESIGN CLASS HIERARCHIES
2. DATA MANAGEMENT EXPERTISE: IMPLEMENTED FILE-BASED PERSISTENCE
3. PROBLEM-SOLVING ABILITY: TACKLED COMPLEX LOGIC CHALLENGES
4. FULL-PROJECT EXPERIENCE: WENT FROM REQUIREMENTS TO COMPLETE SYSTEM

THE SYSTEM SUCCESSFULLY MEETS ALL CORE REQUIREMENTS WHILE PROVIDING A FOUNDATION FOR FUTURE EXPANSION. IT SHOWCASES HOW THOUGHTFUL SOFTWARE DESIGN CAN AUTOMATE AND IMPROVE REAL-WORLD PROCESSES.