LIBRARY MANAGEMENT SYSTEM

Name: Grihika Jeloka

Class: XII Section: Science A

Session: 2023 - 24

INDEX

ACKNOWLEDGMENT

Apart from my personal efforts, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

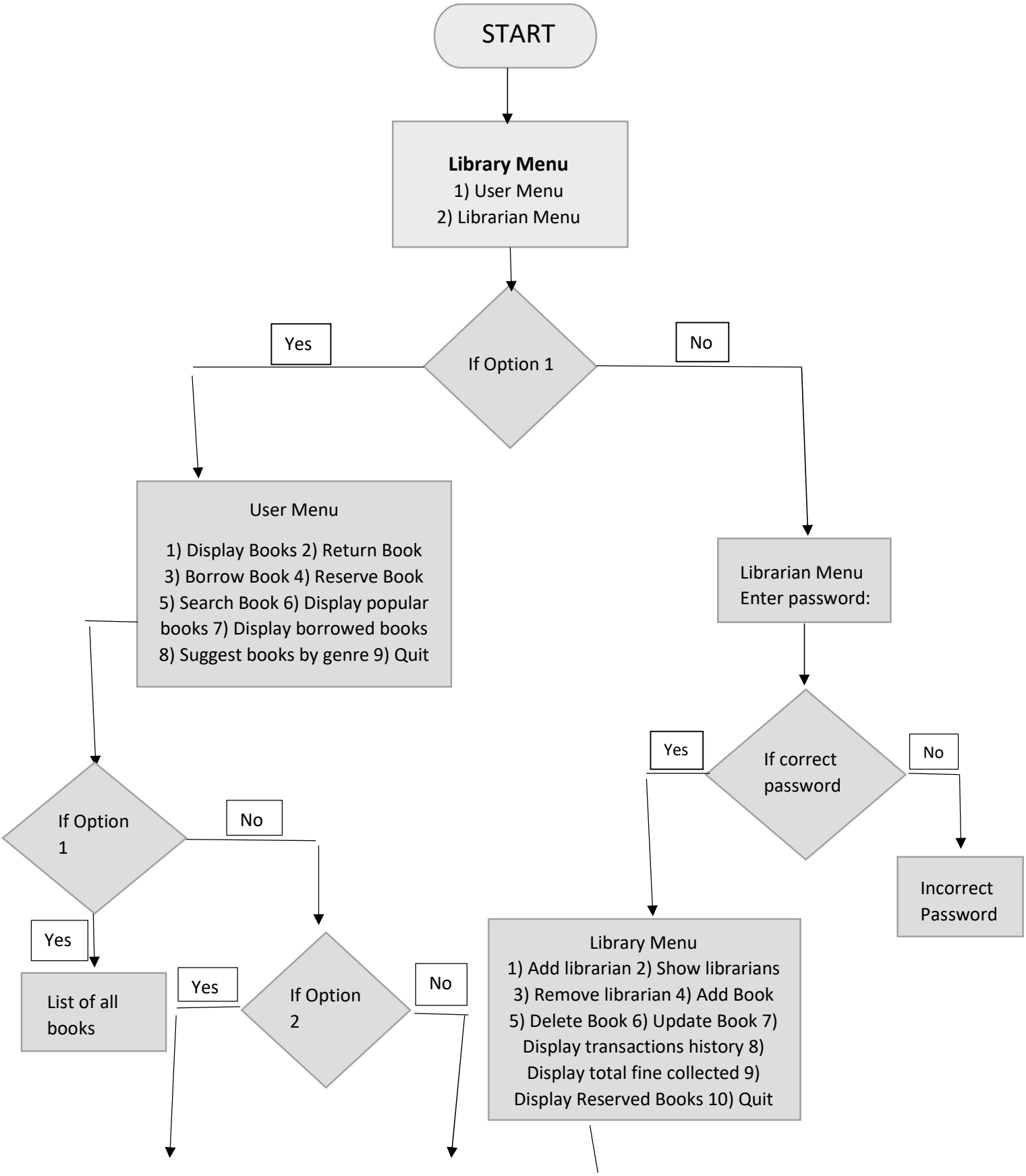
I would like to express my sincere gratitude to my Vice principal Mrs. Rina Maitra and my Senior School Coordinator, Mrs. Tanushree Bhattacharyya for providing me with facilities required to do this project.

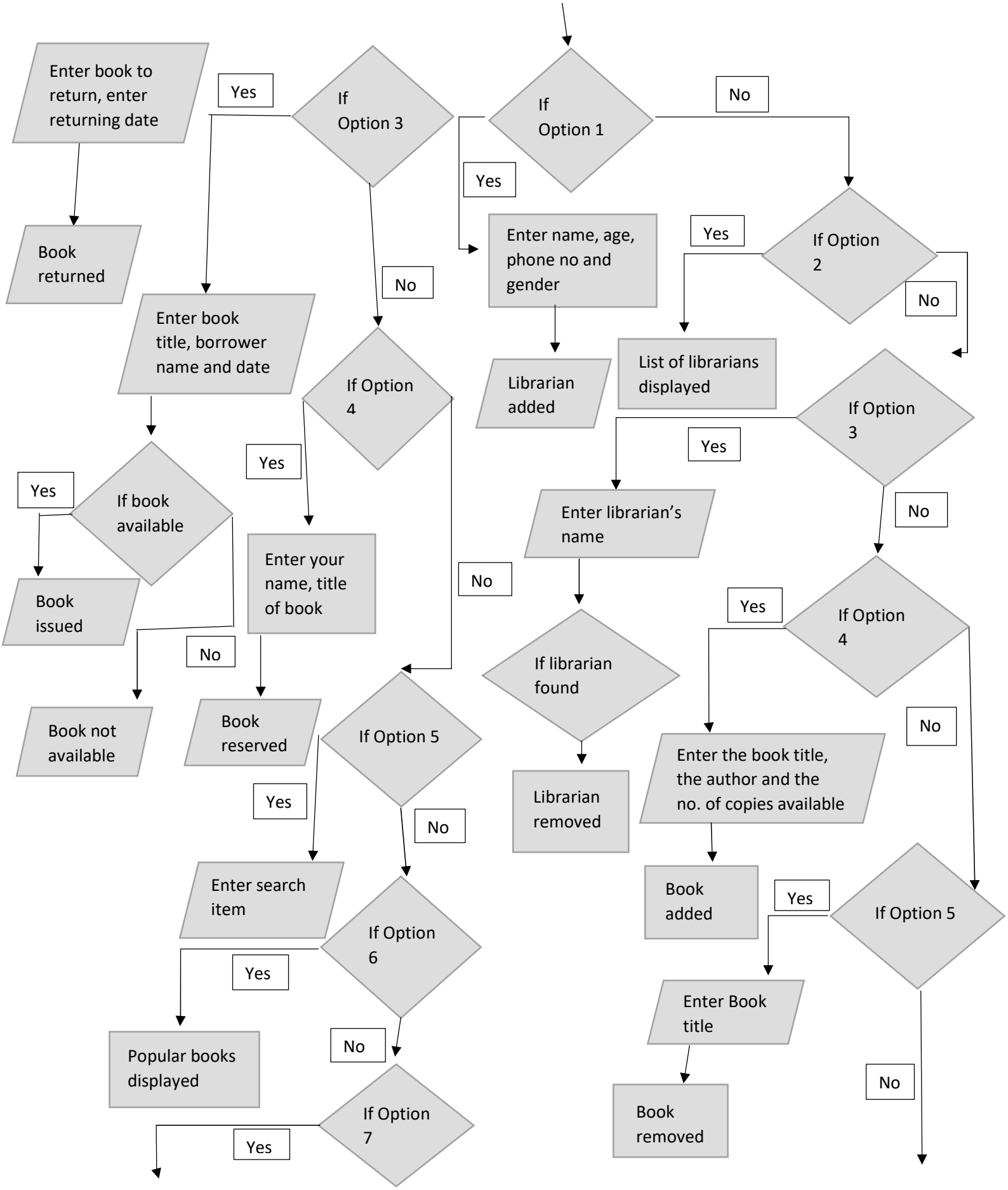
I am highly indebted to my Computer Science teacher Mr. Indranath Singha for his expertise and mentorship in the subject matter that greatly contributed to the success of this project.

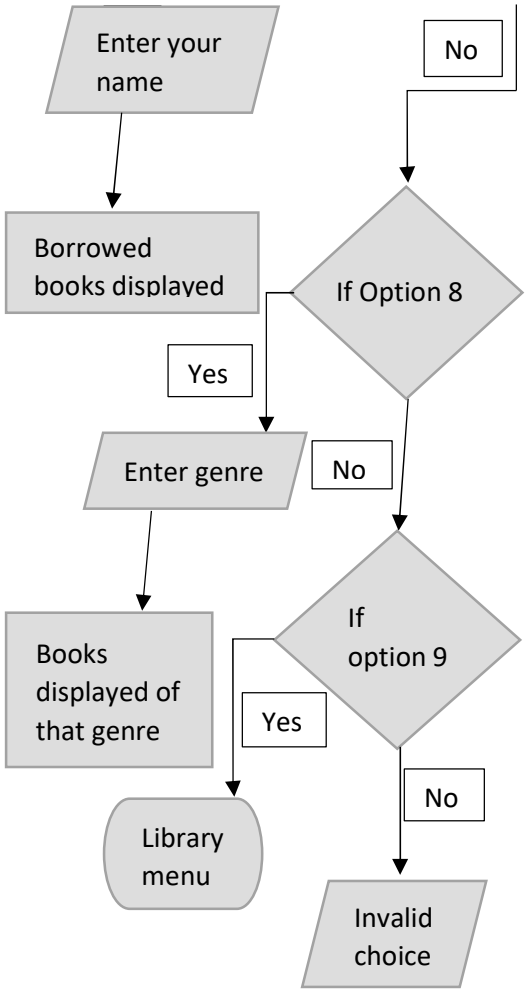
I would also like to thank my parents for their unwavering encouragement, support, and belief in my abilities. Their love and guidance have been my constant source of inspiration.

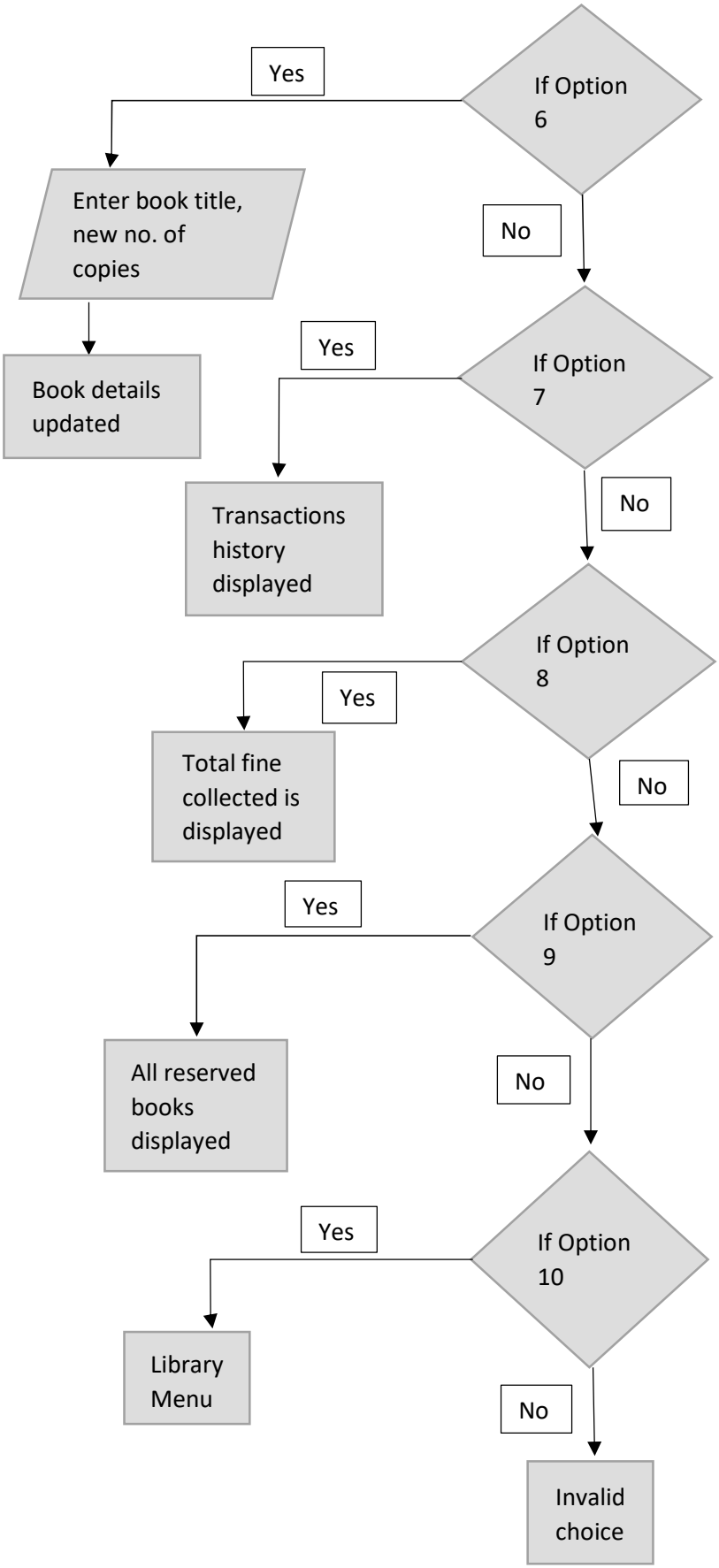
I would like to express my heartfelt thanks to Soumili Saha, my group member for her collaborative efforts and dedication throughout the completion of this project.

DATA FLOW DIAGRAM




If Option 1
Yes
No
Enter name, age, phone no and gender
Librarian added
If Option 2
Yes
No
List of librarians displayed
If Option 3
Yes
No
Enter librarian's name
If librarian found
Librarian removed
If Option 4
Yes
No
Enter the book title, the author and the no. of copies available
Book added
If Option 5
Yes
No
Enter Book title
Book removed
If Option 3
Yes
No
Enter book to return, enter returning date
Book returned
If Option 4
Yes
No
Enter book title, borrower name and date
If book available
Yes
No
Book issued
Book not available
If Option 5
Yes
No
Enter your name, title of book
Book reserved
If Option 6
Yes
No
Enter search item
Popular books displayed
If Option 7
Yes


Enter your name
Borrowed books displayed
If Option 8
Yes
No
Enter genre
No
Books displayed of that genre
If option 9
Yes
No
Library menu
Invalid choice


If Option 6
Yes
No
Enter book title, new no. of copies
Book details updated
If Option 7
Yes
No
Transactions history displayed
If Option 8
Yes
No
Total fine collected is displayed
If Option 9
Yes
No
All reserved books displayed
If Option 10
Yes
No
Library Menu
Invalid choice

SYNOPSIS

The Library Management System is a project created using the Python programming language. The main goal of this project is to create a system that makes managing a library easier. It aims to be user-friendly and efficient, helping with tasks like handling books, transactions, and providing tools for librarians to manage things smoothly.

The system caters to users by providing an integrated interface for seamless navigation through the library's extensive book collection. Users can effortlessly explore titles, search for specific authors, borrow or return books, and display their borrowing history. Additionally, the system offers personalized book suggestions, enhancing the overall user experience.

With a set of strong administrative tools, librarians can effectively handle the library's resources. This includes tasks such as adding or removing books, updating book details, overseeing transactions, and generating insightful reports on total fines collected. The system's reporting features empower librarians with data-driven insights for informed decision-making.

The **create_database** function establishes a connection to the MySQL server and creates the "library" database if it does not exist, ensuring database persistence. The **create_transactions_table** function, also establishing a connection, creates the "transactions" table, defining its structure for tracking book transactions. Similarly, the **create_librarians_table** function ensures the existence of the "librarians" table, defining its structure to store librarian information. These functions collectively set up the foundational elements of the library management system's MySQL database, facilitating the recording of book transactions and librarian details.

User Menu:

1. Display Books

This function provides a user-friendly interface for users to read the library catalogue and gain insights into the diverse range of books at their disposal.

2. Search Books

It caters to user preferences and facilitates quick access to desired books, enhancing the overall user experience within the library system.

3. Borrow Books

Users can input their name, the borrowing date, and the title of the book they wish to borrow. The system then processes the transaction, updating the database to reflect the change in the book's availability.

4. Return Books

Users provide their name, the return date, and the title of the book they are returning. The system calculates any applicable fines and updates the database to reflect the return of the book, ensuring accurate record-keeping.

5. Reserve Books

It caters to users who want to reserve a book for future borrowing. Users provide their name and the title of the book they wish to reserve. The system checks the availability of the book, updates the reservation list, and adjusts the book's availability accordingly.

6. Display User Borrowed Books

Users input their name, and the system retrieves and displays a list of borrowed books, providing transparency and assisting users in managing their borrowed items.

7. Suggest Books Based on Genre

It is designed to enhance user experience by suggesting books based on a specified genre.

8. Display Popular Books

It is designed to benefit library users by offering insights into the popularity of books. Users can access a ranked list of the most borrowed books, aiding them in discovering popular and in-demand titles within the library collection.

Librarian Menu:

1. Add Librarian

Librarians can input details such as the name, age, phone number, and gender of the new librarian, facilitating the growth and management of the librarian workforce.

2. Remove Librarian

Librarians can input the name of the librarian they wish to remove, ensuring an up-to-date and well-managed librarian roster.

3. Show Librarian

It provides librarians with a convenient way to view the details of all librarians within the system.

4. Add Book

It simplifies the process of expanding the library's collection by allowing librarians to add new books.

5. Remove Book

This enables librarians to manage the library's collection by removing specific books.

6. Update Book

This allows librarians to update the details of a specific book within the library catalogue.

7. Display Transactions History

It provides librarians with a comprehensive overview of the library's transaction history. This includes details about borrowed and returned books, aiding librarians in tracking library activities and addressing any issues promptly.

8. Display Total Fine Collected

This is a reporting feature that showcases the total amount of fines collected by the library. Librarians can access this information, facilitating financial tracking and ensuring that the library's financial records are accurate.

9. Display Reserved Books

It assists librarians in monitoring and managing book reservations.

SOURCE CODE

```
import csv
import mysql.connector

# Define the list of books
books = [

    ["To Kill a Mockingbird", "Harper Lee", 10, "Fiction"],
    ["1984", "George Orwell", 15, "Dystopian"],
    ["Pride and Prejudice", "Jane Austen", 12, "Romance"],
    ["The Great Gatsby", "F. Scott Fitzgerald", 8, "Classic"],
    ["The Catcher in the Rye", "J.D. Salinger", 7, "Coming of Age"],
    ["Moby-Dick", "Herman Melville", 5, "Adventure"],
    ["The Lord of the Rings", "J.R.R. Tolkien", 20, "Fantasy"],
    ["The Hobbit", "J.R.R. Tolkien", 18, "Fantasy"],
    ["Brave New World", "Aldous Huxley", 9, "Science Fiction"],
    ["The Chronicles of Narnia", "C.S. Lewis", 14, "Fantasy"],
    ["War and Peace", "Leo Tolstoy", 11, "Historical Fiction"],
    ["Crime and Punishment", "Fyodor Dostoevsky", 6, "Psychological Thriller"],
    ["The Odyssey", "Homer", 13, "Epic Poetry"],
    ["The Grapes of Wrath", "John Steinbeck", 7, "Classic"],
    ["Frankenstein", "Mary Shelley", 8, "Gothic Fiction"],
    ["The Hunger Games", "Suzanne Collins", 12, "Dystopian"],
    ["The Road", "Cormac McCarthy", 10, "Post-Apocalyptic"],
    ["The Shining", "Stephen King", 9, "Horror"],
    ["Harry Potter and the Sorcerer's Stone", "J.K. Rowling", 25, "Fantasy"],
    ["The Da Vinci Code", "Dan Brown", 14, "Mystery"],
```

["The Alchemist", "Paulo Coelho", 11, "Philosophical Fiction"],

["Gone with the Wind", "Margaret Mitchell", 7, "Historical Romance"],

["The Hitchhiker's Guide to the Galaxy", "Douglas Adams", 15, "Science Fiction"],

["The Sun Also Rises", "Ernest Hemingway", 6, "Modernist"],

["The Stranger", "Albert Camus", 8, "Philosophical Fiction"],

["The Fault in Our Stars", "John Green", 10, "Young Adult"],

["The Girl with the Dragon Tattoo", "Stieg Larsson", 9, "Mystery"],

["Catch-22", "Joseph Heller", 7, "Satire"],

["Wuthering Heights", "Emily Brontë", 6, "Gothic Fiction"],

["The Picture of Dorian Gray", "Oscar Wilde", 8, "Gothic Fiction"],

["The Road Not Taken", "Robert Frost", 5, "Poetry"],

["The Old Man and the Sea", "Ernest Hemingway", 6, "Adventure"],

["One Hundred Years of Solitude", "Gabriel García Márquez", 11, "Magical Realism"],

["The Divine Comedy", "Dante Alighieri", 10, "Epic Poetry"],

["The Art of War", "Sun Tzu", 12, "Military Strategy"],

["The Adventures of Sherlock Holmes", "Arthur Conan Doyle", 14, "Mystery"],

["Jane Eyre", "Charlotte Brontë", 9, "Gothic Fiction"],

["A Tale of Two Cities", "Charles Dickens", 10, "Historical Fiction"],

["The Scarlet Letter", "Nathaniel Hawthorne", 7, "Historical Fiction"],

["The Count of Monte Cristo", "Alexandre Dumas", 11, "Adventure"],

["Lord of the Flies", "William Golding", 8, "Allegorical"],

["The Kite Runner", "Khaled Hosseini", 9, "Historical Fiction"],

["The Secret Garden", "Frances Hodgson Burnett", 10, "Children's Literature"],

["Dracula", "Bram Stoker", 7, "Gothic Fiction"],

["A Game of Thrones", "George R.R. Martin", 12, "Fantasy"],

["The Name of the Wind", "Patrick Rothfuss", 9, "Fantasy"],

```
    ["The Road Less Traveled", "M. Scott Peck", 6, "Psychology"],
    ["The Girl on the Train", "Paula Hawkins", 8, "Thriller"],
    ["The Silent Patient", "Alex Michaelides", 7, "Psychological Thriller"],
    ["Where the Crawdads Sing", "Delia Owens", 10, "Mystery"]

]



# Open the CSV file in write mode and write the book details
with open("book_details.csv", mode='w', newline='') as file:

    writer = csv.writer(file)
    # Write the header row
    writer.writerow(["Book Name", "Author", "Number of Copies Available",
"Genre"])
    # Write the book data
    writer.writerows(books)



# Open the CSV file in write mode and create a reservations file
with open("book_reservations.csv", mode='w', newline='') as reservations_file:

        writer = csv.writer(reservations_file)
        writer.writerow(["Book Title", "Reserver Name"])



# Create a MySQL database 'library'
def create_database():
```

```
    mydb = mysql.connector.connect(
        host="Localhost",
        user="root",
        password=""
    )

    cursor = mydb.cursor()
    cursor.execute("CREATE DATABASE IF NOT EXISTS library")

    mydb.commit()
    cursor.close()


# Open a MySQL database connection for transactions
def create_transactions_table():

    try:
        mydb = mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        cursor = mydb.cursor()
        cursor.execute('''CREATE TABLE IF NOT EXISTS transactions (
            book_title VARCHAR(255),
            borrower_name VARCHAR(255),
            borrow_date DATE,
```

```
            return_date DATE,
            fine DECIMAL(10,2)
        )''')

        mydb.commit()
        cursor.close()

    except mysql.connector.Error as err:
        print("Error:", err)
        print("Failed to create the transactions table. Please try again.")


# Open a MySQL database connection for librarians
def create_librarians_table():

    try:
        mydb = mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        cursor = mydb.cursor()
        cursor.execute('''CREATE TABLE IF NOT EXISTS librarians (
            name varchar(20),
            Age int(2),
            Phone_no int(20),
            Sex char(1)
```

```
        )"")

        mydb.commit()
        cursor.close()

    except mysql.connector.Error as err:
        print("Error:", err)
        print("Failed to create the librarians table. Please try again.")


# Define the library password
library_password="Library123"


# Define the maximum number of incorrect password attempts
max_pw_attempts = 3


# Function to validate librarian password
def validate_librarian_password():
    password_attempts = 0

    while password_attempts < max_pw_attempts:
        password = input("Enter the librarian password: ")

        if password == library_password:
            return True  # Password is correct, allow access
        else:
            print("Incorrect password. Please try again.")
```

```
            password_attempts += 1

    print("Too many incorrect password attempts.")
    return False  # Deny access after reaching maximum attempts



# Function to display books in the library
def display_books():

    print("Books in the library:")
    with open("book_details.csv", "r", newline="\r\n") as file:
        books=csv.reader(file)
        next(books)  # Skip the header row
        for book in books:
            print(book[0] + " by " + book[1] + ". Genre: " + book[3] + ". Copies
available: " + book[2])



# Function to search for books
def search_books():

    query = input("Enter a search term (title or author): ")
    results = []

    with open("book_details.csv", "r", newline="\r\n") as file:
        books=csv.reader(file)
        next(books)  # Skip the header row

        for book in books:
```

```
            if query.lower() in book[0].lower() or query.lower() in book[1].lower():
                results.append(book)
        if results:
            for book in results:
                print(book[0] + " by " + book[1] + ". Genre: " + book[3] + ". Copies available: " + book[2])
        else:
            print("No results found.")



# Function to borrow books and store transactions in MySQL
def borrow_books():

    borrower = input("Enter your name: ")
    borrow_date = input("Enter the borrowing date (YYYY-MM-DD): ")

    while True:

        # Continue borrowing books until the user types 'done'
        title = input("Enter the title of the book to borrow (or type 'done' to finish): ")
        if title.lower() == 'done':
            break

        # Check if the user has a reservation for the specified book
        with open("book_reservations.csv", mode='r', newline='') as reservations_file:
            reservations = csv.reader(reservations_file)

            for reservation in reservations:
```

```
            if reservation[0].lower() == title.lower() and reservation[1].lower() ==
borrower.lower():
                print("You have a reservation for the book", title + ".")
                proceed = input("Do you want to borrow it? (yes/no): ")
                if proceed.lower() != 'yes':
                    break  # Exit the loop if the user chooses not to proceed with
the reservation

                # Remove the reservation since the user is borrowing the book
                with open("book_reservations.csv", mode='w', newline='') as
updated_reservations_file:
                    writer = csv.writer(updated_reservations_file)
                    for r in reservations:
                        if r != reservation:
                            writer.writerow(r)
                break  # Exit the loop after processing the specified book

        # Open the CSV file in read and write mode
        with open("book_details.csv", "r+", newline="\r\n") as file:
            books = list(csv.reader(file))
            found = False  # Flag to track whether the book is found in the library

            for book in books:

                if book[0].lower() == title.lower():
                    found = True

                    # Check if there are available copies of the book
                    if int(book[2]) > 0:
```

```
                # Decrease the available copies of the book
                book[2] = str(int(book[2]) - 1)

                # Connect to the MySQL database for storing transactions
                mycon = mysql.connector.connect(
                    host="Localhost",
                    user="root",
                    password="",
                    database="library"
                )

                cursor = mycon.cursor()

                # Insert the transaction into the 'transactions' table
                cursor.execute("INSERT INTO transactions (book_title,
borrower_name, borrow_date) VALUES (%s, %s, %s)",
                               (title, borrower, borrow_date))

                mycon.commit()
                cursor.close()

                print("Book", title, "borrowed successfully.")
            else:
                print("Sorry, no available copies of", title, "left.")

            break  # Exit the loop after processing the specified book

    # If the book is not found in the library, print a message
    if not found:
```

```
            print("Sorry, the book", title, "is not available in the library.")

        # Update the CSV file with the new book data outside the for loop
        file.seek(0)
        writer = csv.writer(file)
        writer.writerows(books)  # Writing all books, including the updated one


# Function to display the user's borrowed books
def display_user_borrowed_books():

    borrower_name = input("Enter your name: ")

    mycon = mysql.connector.connect(
        host="Localhost",
        user="root",
        password="",
        database="library"
    )

    cursor = mycon.cursor()
    cursor.execute("SELECT book_title, borrow_date FROM transactions
WHERE borrower_name = %s AND return_date IS NULL", (borrower_name,))
    borrowed_books = cursor.fetchall()

    if cursor.rowcount > 0:
        print("\nYOUR BORROWED BOOKS:")
        print("Book Title | Borrow Date")
        for book in borrowed_books:
```

```
            print(book[0] + " | " + str(book[1]))
    else:
        print("You haven't borrowed any books.")

    cursor.close()
    mycon.close()


# Function to suggest books based on genre
def suggest_books_based_on_genre():
    target_genre = input("Enter a genre to get book suggestions: ")
    suggested_books = []

    with open("book_details.csv", "r", newline="\r\n") as file:
        books = csv.reader(file)
        next(books)  # Skip the header row

        for book in books:
            if book[3].lower() == target_genre.lower():
                suggested_books.append(book)

    if suggested_books:
        print("Suggested books in the", target_genre, "genre:")
        for book in suggested_books:
            print(book[0] + " by " + book[1] + ". Copies available: " + book[2])
    else:
        print("No books found in the", target_genre, "genre.")
```

```
#Function to generate a report on popular books
def popular_books():

    mycon = mysql.connector.connect(
        host="Localhost",
        user="root",
        password="",
        database="library"
    )

    cursor = mycon.cursor()
    cursor.execute("SELECT book_title, COUNT(*) as borrow_count FROM
transactions GROUP BY book_title ORDER BY borrow_count DESC")
    popular_books = cursor.fetchall()

    if cursor.rowcount > 0:
        print("\nPopular Books:")
        print("Rank | Book Title | Borrow Count")
        rank = 1
        for book in popular_books[:10]:
            print(str(rank) + ". " + book[0] + " | " + str(book[1]) + " times borrowed")
            rank += 1

    else:
        print("No transaction records found.")

    cursor.close()
    mycon.close()
```

```
# Function to return books with fine collection
def return_books():

    # Collect borrower's name and return date
    borrower = input("Enter the borrower's name: ")
    return_date = input("Enter the returning date (YYYY-MM-DD): ")

    while True:

        title = input("Enter the title of the book to return (or type 'done' to finish):
")
        if title.lower() == 'done':
            break

        mycon = mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        cursor = mycon.cursor()

        # Check if the book is borrowed by the specified borrower and return it
        cursor.execute("SELECT borrow_date FROM transactions WHERE
book_title=%s AND borrower_name=%s AND return_date IS NULL",
                       (title, borrower))
        borrow_date_row = cursor.fetchone()
```

```
    if borrow_date_row:

        # Extract the borrow date from the database result
        borrow_date = borrow_date_row[0]

        # Calculate the fine based on the specified conditions
        return_year, return_month, return_day = int(return_date[:4]), int(return_date[5:7]), int(return_date[8:])
        borrow_year, borrow_month, borrow_day = int(borrow_date.year), int(borrow_date.month), int(borrow_date.day)

        fine_days = (return_year - borrow_year) * 365 + (return_month - borrow_month) * 30 + (return_day - borrow_day)
        fine = max(fine_days - 30, 0) * 50  # Calculate the fine (50 rupees per day, after 30 days)

        # Update the transactions table with the return date and fine
        cursor.execute("UPDATE transactions SET return_date=%s, fine=%s WHERE book_title=%s AND borrower_name=%s AND return_date IS NULL",
                       (return_date, fine, title, borrower))
        mycon.commit()

        affected_rows = cursor.rowcount

        if affected_rows > 0:

            # Book returned successfully, print a message if there is a fine
            if fine > 0:
                print("Book", title, "returned successfully. You were late by", fine_days - 30, "day(s) hence you need to pay a fine of", fine, "rupees.")
```

```
            else:
                print("Book", title, "returned successfully.")

            # Update the book details in the CSV file
            with open("book_details.csv", "r+", newline="\r\n") as file:
                books = list(csv.reader(file))
                for book in books:
                    if book[0] == title:
                        book[2] = str(int(book[2]) + 1)  # Increase available copies

                file.seek(0)
                writer = csv.writer(file)
                writer.writerows(books)

        else:
            print("Book", title, "has already been returned.")

    else:
        # Book not found in the borrowing records
        print("Book", title, "not found in the borrowing records for the specified borrower.")

    cursor.close()
    mycon.close()

#Allows users to reserve a book for future borrowing
def reserve_book():

    reserver_name = input("Enter your name: ")
```

```
    book_title = input("Enter the title of the book to reserve: ")

    # Check if the book is available
    with open("book_details.csv", mode='r+', newline='') as details_file:

        details = csv.reader(details_file)
        next(details)  # Skip header

        for book in details:

            if book[0] == book_title and int(book[2]) > 0:

                # Reserve the book
                with open("book_reservations.csv", mode='a', newline='') as
reservations_file:
                    reservations_file.seek(0)
                    writer = csv.writer(reservations_file)
                    writer.writerow([book_title, reserver_name])

                # Update available copies in book_details.csv
                book[2] = str(int(book[2]) - 1)
                details_file.seek(0)
                writer = csv.writer(details_file)
                writer.writerows(details)

                print("The book", book_title, "has been reserved for", reserver_name
+ ".")
                return
```

```
        else:
            print("Sorry, the book", book_title, "is not available for reservation.")


#Displays the list of books reserved by users
def display_reserved_books():

    with open("book_reservations.csv", mode='r', newline='') as file:
        reservations = csv.reader(file)
        print("\nReserved Books:")
        for reservation in reservations:
            print(reservation[0], "reserved by", reservation[1])


#Function to update book details
def update_book_details():

    title = input("Enter the title of the book to update: ")

    with open("book_details.csv", "r+", newline="\r\n") as file:
        books = list(csv.reader(file))
        for book in books:

            if book[0] == title:
                new_copies = int(input("Enter the new number of copies available:
"))
                book[2] = str(new_copies)
                print("Book details updated successfully.")
                break
```

```
        else:
            print("Book not found in the library.")
            return

        # Update the CSV file with the new book data
        file.seek(0)
        writer = csv.writer(file)
        writer.writerows(books)  # Writing all books, including the updated one


# Function to display transactions history
def display_transactions_history():

    mycon = mysql.connector.connect(
        host="Localhost",
        user="root",
        password="",
        database="library"
    )

    cursor = mycon.cursor()
    cursor.execute("SELECT * FROM transactions")
    transactions = cursor.fetchall()

    if cursor.rowcount > 0:
        print("\nTRANSACTIONS HISTORY")
        print("Book Title | Borrower Name | Borrow Date | Return Date | Fine")
        for transaction in transactions:
```

```
            print(transaction[0], "|", transaction[1], "|", transaction[2], "|",
transaction[3], "|", transaction[4])
    else:
        print("No transactions found.")

    cursor.close()
    mycon.close()



# Function to calculate and display total fine collected
def display_total_fine_collected():

    mycon = mysql.connector.connect(
        host="Localhost",
        user="root",
        password="",
        database="library"
    )

    cursor = mycon.cursor()
    cursor.execute("SELECT SUM(fine) FROM transactions")
    total_fine = cursor.fetchone()[0]

    if total_fine:
        print("\nTotal Fine Collected:", total_fine, "rupees")
    else:
        print("No fine collected.")

    cursor.close()
```

```
    mycon.close()


# Function to add a librarian
def add_librarian():

    try:
        mycon=mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        name = input("Enter the librarian's name:")
        Age=int(input("Enter age:"))
        Phone_no=int(input("Enter phone no:"))
        Sex=input("Enter gender(M/F):")
        cursor=mycon.cursor()
        cursor.execute("INSERT INTO librarians (name, Age, Phone_no, Sex)
VALUES (%s, %s, %s, %s)",(name, Age, Phone_no, Sex,))

        mycon.commit()
        cursor.close()

        print("Librarian", name, "has been added.")

    except mysql.connector.Error as err:
        print("Error:", err)
```

```
        print("Failed to add librarian. Please try again.")

    finally:
        mycon.close()


#Function to delete a librarian
def remove_librarian():

    try:

        mycon = mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        cursor=mycon.cursor()

        name = input("Enter the librarian's name:")
        # Use the DELETE statement to delete the librarian
        cursor.execute("DELETE FROM librarians WHERE name = %s",(name,))
        mycon.commit()

        if cursor.rowcount > 0:
            print("Librarian, " ,name, "has been removed.")
        else:
            print("No librarian with the name" ,name, "found.")
```

```
        cursor.close()
        mycon.close()

    except mysql.connector.Error as err:
        print("Error:", err)
        print("Failed to remove librarian. Please try again.")

    finally:
        mycon.close()


# Function to show librarians
def show_librarians():

    try:
        mycon=mysql.connector.connect(
            host="Localhost",
            user="root",
            password="",
            database="library"
        )

        cursor=mycon.cursor()
        cursor.execute("SELECT * FROM librarians")
        data = cursor.fetchall()

        if cursor.rowcount > 0:
            print("\nLibrarians:")
```

```
            for librarian in data:
                print("Name: " + str(librarian[0]) + ", Age: " + str(librarian[1]) + ", Phone No: " + str(librarian[2]) + ", Gender: " + str(librarian[3]))
        else:
            print("No librarian found.")

        cursor.close()
        mycon.close()

    except mysql.connector.Error as err:
        print("Error:", err)
        print("Failed to show librarian. Please try again.")

    finally:
        mycon.close()



# Function to add a book
def add_book():

    title = input("Enter the title of the book: ")
    author = input("Enter the author of the book: ")
    copies = int(input("Enter the number of copies available: "))
    genre = input("Enter the genre of the book: ")

    with open("book_details.csv", "r+", newline="\r\n") as file:
        books = list(csv.reader(file))
        books.append([title, author, copies, genre])
        print("Book", title, "by", author, "has been added with", copies, "copies.")
```

```
        # Update the CSV file with the new book data
        file.seek(0)
        writer = csv.writer(file)
        writer.writerows(books)  # Writing all books, including the updated one


# Function to remove a book
def remove_book():

    title = input("Enter the title of the book to remove: ")
    with open("book_details.csv", "r+", newline="\r\n") as file:
        books = list(csv.reader(file))

        # Find and remove the book from the list
        for book in books:
            if book[0] == title:
                books.remove(book)
                print("Book", title, "has been removed.")
                break
        else:
            print("Book not found in the library.")
            return

        # Update the CSV file with the new book data
        file.seek(0)
        writer = csv.writer(file)
        writer.writerows(books)  # Writing all books, including the updated one
```

```
# Function to display the librarian menu
def librarian_menu():

    while True:

        print("\nLIBRARIAN MENU")
        print("1. Add Librarian")
        print("2. Show Librarians")
        print("3. Remove Librarian")
        print("4. Add Book")
        print("5. Delete Book")
        print("6. Update Book")
        print("7. Display Transactions History")
        print("8. Display Total Fine Collected")
        print("9. Display Reserved Books")
        print("10. Quit")

        choice = int(input("Enter your choice: "))
        if choice == 1:
            add_librarian()

        elif choice == 2:
            show_librarians()

        elif choice == 3:
            remove_librarian()

        elif choice == 4:
```

```
            add_book()

        elif choice == 5:
            remove_book()

        elif choice == 6:
            update_book_details()

        elif choice == 7:
            display_transactions_history()

        elif choice == 8:
            display_total_fine_collected()

        elif choice == 9:
            display_reserved_books()

        elif choice == 10:
            break

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


# Function to display the user menu
def user_menu():

    while True:
```

```
print("\nUSER MENU")
print("1. Display Books")
print("2. Borrow Book")
print("3. Reserve Book")
print("4. Return Book")
print("5. Search Book")
print("6. Display Popular Books")
print("7. Display Borrowed Books")
print("8. Suggest Books by Genre")
print("9. Quit")

choice = int(input("Enter your choice: "))
if choice == 1:
    display_books()

elif choice == 2:
    borrow_books()

elif choice == 3:
    reserve_book()

elif choice == 4:
    return_books()

elif choice == 5:
    search_books()

elif choice == 6:
    popular_books()
```

```
        elif choice == 7:
            display_user_borrowed_books()

        elif choice == 8:
            suggest_books_based_on_genre()

        elif choice == 9:
            break

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


# Start the user menu
def library_management_system():

    create_database()
    create_transactions_table()
    create_librarians_table()

    print("------WELCOME TO THE LIBRARY------")

    while True:
        print("1. User Menu")
        print("2. Librarian Menu")
        print("3. Exit")
        ch = int(input("Enter your choice: "))
```

```
        if ch == 1:
            user_menu()

        elif ch == 2:
            # Validate librarian password before entering the librarian menu
            if validate_librarian_password():
                librarian_menu()
            else:
                print("Access denied. Returning to the main menu.")

        elif ch == 3:
            break

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

library_management_system()
```

OUTPUT

LIBRARY MENU:

1. USER MENU

```
------WELCOME TO THE LIBRARY------
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice: 1

USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice:
```

If Choice is 1 (Display Books)

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 1
Books in the library:
To Kill a Mockingbird by Harper Lee. Genre: Fiction. Copies available: 10
1984 by George Orwell. Genre: Dystopian. Copies available: 15
Pride and Prejudice by Jane Austen. Genre: Romance. Copies available: 12
The Great Gatsby by F. Scott Fitzgerald. Genre: Classic. Copies available: 8
The Catcher in the Rye by J.D. Salinger. Genre: Coming of Age. Copies available: 7
Moby-Dick by Herman Melville. Genre: Adventure. Copies available: 5
The Lord of the Rings by J.R.R. Tolkien. Genre: Fantasy. Copies available: 20
The Hobbit by J.R.R. Tolkien. Genre: Fantasy. Copies available: 18
Brave New World by Aldous Huxley. Genre: Science Fiction. Copies available: 9
The Chronicles of Narnia by C.S. Lewis. Genre: Fantasy. Copies available: 14
War and Peace by Leo Tolstoy. Genre: Historical Fiction. Copies available: 11
Crime and Punishment by Fyodor Dostoevsky. Genre: Psychological Thriller. Copies available: 6
The Odyssey by Homer. Genre: Epic Poetry. Copies available: 13
The Grapes of Wrath by John Steinbeck. Genre: Classic. Copies available: 7
Frankenstein by Mary Shelley. Genre: Gothic Fiction. Copies available: 8
The Hunger Games by Suzanne Collins. Genre: Dystopian. Copies available: 12
The Road by Cormac McCarthy. Genre: Post-Apocalyptic. Copies available: 10
The Shining by Stephen King. Genre: Horror. Copies available: 9
Harry Potter and the Sorcerer's Stone by J.K. Rowling. Genre: Fantasy. Copies available: 25
The Da Vinci Code by Dan Brown. Genre: Mystery. Copies available: 14
The Alchemist by Paulo Coelho. Genre: Philosophical Fiction. Copies available: 11
Gone with the Wind by Margaret Mitchell. Genre: Historical Romance. Copies available: 7
The Hitchhiker's Guide to the Galaxy by Douglas Adams. Genre: Science Fiction. Copies available: 15
The Sun Also Rises by Ernest Hemingway. Genre: Modernist. Copies available: 6
```

If Choice is 2 (Borrow Books)

- Borrowing single book

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 2
Enter your name: Grihika
Enter the borrowing date (YYYY-MM-DD): 2023-11-25
Enter the title of the book to borrow (or type 'done' to finish): The Hobbit
Book The Hobbit borrowed successfully.
Enter the title of the book to borrow (or type 'done' to finish): done
```

- Borrowing multiple books

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 2
Enter your name: Soumili
Enter the borrowing date (YYYY-MM-DD): 2023-11-20
Enter the title of the book to borrow (or type 'done' to finish): 1984
Book 1984 borrowed successfully.
Enter the title of the book to borrow (or type 'done' to finish): Pride and Prejudice
Book Pride and Prejudice borrowed successfully.
Enter the title of the book to borrow (or type 'done' to finish): done
```

- Book is not available

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 2
Enter your name: abc
Enter the borrowing date (YYYY-MM-DD): 2023-10-11
Enter the title of the book to borrow (or type 'done' to finish): xyz
Sorry, the book xyz is not available in the library.
Enter the title of the book to borrow (or type 'done' to finish): done
```

If Choice is 3 (Reserve Book)

- Book is available

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 3
Enter your name: Grihika
Enter the title of the book to reserve: The Shining
The book The Shining has been reserved for Grihika.
```

- Book is not available

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 3
Enter your name: Soumili
Enter the title of the book to reserve: a
Sorry, the book a is not available for reservation.
```

If Choice is 4 (Return Book)

- Returned on time

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 4
Enter the borrower's name: Soumili
Enter the returning date (YYYY-MM-DD): 2023-12-01
Enter the title of the book to return (or type 'done' to finish): Pride and Prejudice
Book Pride and Prejudice returned successfully.
Enter the title of the book to return (or type 'done' to finish): done
```

- Returned late

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 4
Enter the borrower's name: Soumili
Enter the returning date (YYYY-MM-DD): 2023-12-30
Enter the title of the book to return (or type 'done' to finish): 1984
Book 1984 returned successfully. You were late by 10 day(s) hence you need to pay a fine of 500 rupees.
Enter the title of the book to return (or type 'done' to finish): done
```

If Choice is 5 (Search Book)

- Book is in the library

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 5
Enter a search term (title or author): The Hunger Games
The Hunger Games by Suzanne Collins. Genre: Dystopian. Copies available: 12
```

- Book is not in the library

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 5
Enter a search term (title or author): Diary of Anne Frank
No results found.
```

If Choice is 6 (Display Popular Books)

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 6

Popular Books:
Rank | Book Title | Borrow Count
1. 1984 | 2 times borrowed
2. The Secret Garden | 1 times borrowed
3. The Road Not Taken | 1 times borrowed
4. The Art of War | 1 times borrowed
5. The Hobbit | 1 times borrowed
6. Pride and Prejudice | 1 times borrowed
```

If Choice is 7 (Display Borrowed Books)

- User has borrowed books

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 7
Enter your name: Grihika

YOUR BORROWED BOOKS:
Book Title | Borrow Date
The Art of War | 2023-11-18
The Hobbit | 2023-11-25
```

- User has not borrowed books

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 7
Enter your name: Soumili
You haven't borrowed any books.
```

If Choice is 8 (Suggest Books by Genre)

- Genre exists in the library

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 8
Enter a genre to get book suggestions: Thriller
Suggested books in the Thriller genre:
The Girl on the Train by Paula Hawkins. Copies available: 8
```

- Genre does not exist in the library

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 8
Enter a genre to get book suggestions: Travel
No books found in the Travel genre.
```

If Choice is 9 (Quit)

```
USER MENU
1. Display Books
2. Borrow Book
3. Reserve Book
4. Return Book
5. Search Book
6. Display Popular Books
7. Display Borrowed Books
8. Suggest Books by Genre
9. Quit
Enter your choice: 9
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice:
```

2. LIBRARIAN MENU:

- Entering the menu

```
------WELCOME TO THE LIBRARY------
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice: 2
Enter library password: Library123

LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice:
```

- Unauthorized Access Attempt

```
------WELCOME TO THE LIBRARY------
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice: 2
Enter the librarian password: 1
Incorrect password. Please try again.
Enter the librarian password: 3
Incorrect password. Please try again.
Enter the librarian password: 2
Incorrect password. Please try again.
Too many incorrect password attempts.
Access denied. Returning to the main menu.
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice:
```

If Choice is 1 (Add Librarian)

```
------WELCOME TO THE LIBRARY------
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice: 2
Enter library password: Library123

LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 1
Enter the librarian's name:Grihika Jeloka
Enter age:20
Enter phone no:875347291
Enter gender(M/F):F
Librarian Grihika Jeloka has been added.
```

If Choice is 2 (Show Librarians)

- Librarians exists in database

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 2

Librarians:
Name: Grihika Jeloka, Age: 20, Phone No: 875347291, Gender: F
Name: Soumili Saha, Age: 21, Phone No: 953926491, Gender: F
```

- Librarians not in database

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 2
No librarian found.
```

If Choice is 3 (Delete Librarian)

- Librarian exists in database

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 3
Enter the librarian's name:Soumili Saha
Librarian,  Soumili Saha has been removed.

LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 2

Librarians:
Name: Grihika Jeloka, Age: 20, Phone No: 875347291, Gender: F
```

- Librarian not in the database

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 3
Enter the librarian's name:ABC
No librarian with the name ABC found.
```

If Choice is 4 (Add Book)

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 4
Enter the title of the book: Harry Potter and the Chamber of Secrets
Enter the author of the book: J.K. Rowling
Enter the number of copies available: 25
Enter the genre of the book: Fantasy
Book Harry Potter and the Chamber of Secrets by J.K. Rowling has been added with 25 copies.
```

If Choice is 5 (Delete Book)

- Book is in the library

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 5
Enter the title of the book to remove: Catch-22
Book Catch-22 has been removed.
```

- Book is not in the library

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 5
Enter the title of the book to remove: xyz
Book not found in the library.
```

If Choice is 6 (Update Book)

- Book is in the library

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 6
Enter the title of the book to update: Wuthering Heights
Enter the new number of copies available: 20
Book details updated successfully.
```

- Book is not in the library

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 6
Enter the title of the book to update: xyz
Book not found in the library.
```

If Choice is 7 (Display Transactions History)

- Transactions found

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 7

TRANSACTIONS HISTORY
Book Title | Borrower Name | Borrow Date | Return Date | Fine
The Hobbit | Grihika | 2023-11-25 | None | None
1984 | Soumili | 2023-11-20 | 2023-12-30 | 500.00
Pride and Prejudice | Soumili | 2023-11-20 | 2023-12-01 | 0.00
```

- Transactions not found

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 7
No transactions found.
```

If Choice is 8 (Display Total Fine Collected)

- Fine collected

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 8

Total Fine Collected: 500.00 rupees
```

- Fine not collected

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 8
No fine collected.
```

If Choice is 9 (Display Reserved Books)

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 9

Reserved Books:
Book Title reserved by Reserver Name
A Game of Thrones reserved by Soumili
The Silent Patient reserved by Grihika
```

If Choice is 10 (Quit)

```
LIBRARIAN MENU
1. Add Librarian
2. Show Librarians
3. Remove Librarian
4. Add Book
5. Delete Book
6. Update Book
7. Display Transactions History
8. Display Total Fine Collected
9. Display Reserved Books
10. Quit
Enter your choice: 10
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice:
```

3. EXIT

```
------WELCOME TO THE LIBRARY------
1. User Menu
2. Librarian Menu
3. Exit
Enter your choice: 3
>>>
```

BIBLIOGRAPHY

- Computer Science with Python Textbook for Class XII
  by Sumita Arora

- www.w3schools.com

- www.geeksforgeeks.org

- www.tutorialspoint.com