CERTIFICATE

This is to certify that **BIBHASINDHU PRADHAN** of class: **XII A** of **KENDRIYA VIDYALAYA BERHAMPUR** has done his project on **LIBRARY MANAGEMENT SYSTEM** under my supervision. He has taken interest and has shown at most sincerity in completion of this project.

I certify this project up to my expectation & as per guidelines issued by **CBSE, NEW DELHI.**

Internal Examiner *External Examiner*

Principal

ACKNOWLEDGMENT

It is with pleasure that I acknowledge my sincere gratitude to our teacher, ***MR. SAROJ KANTA MISRA*** who taught and undertook the responsibility of teaching the subject computer science. I have been greatly benefited from his classes.

I am especially indebted to our Principal ***MR. SHIVAPRIYA DASH*** who has always been a source of encouragement and support and without whose inspiration this project would not have been a successful I would like to place on record heartfelt thanks to him.

Finally, I would like to express my sincere appreciation for all the other students for my batch their friendship & the fine time that we all shared together.

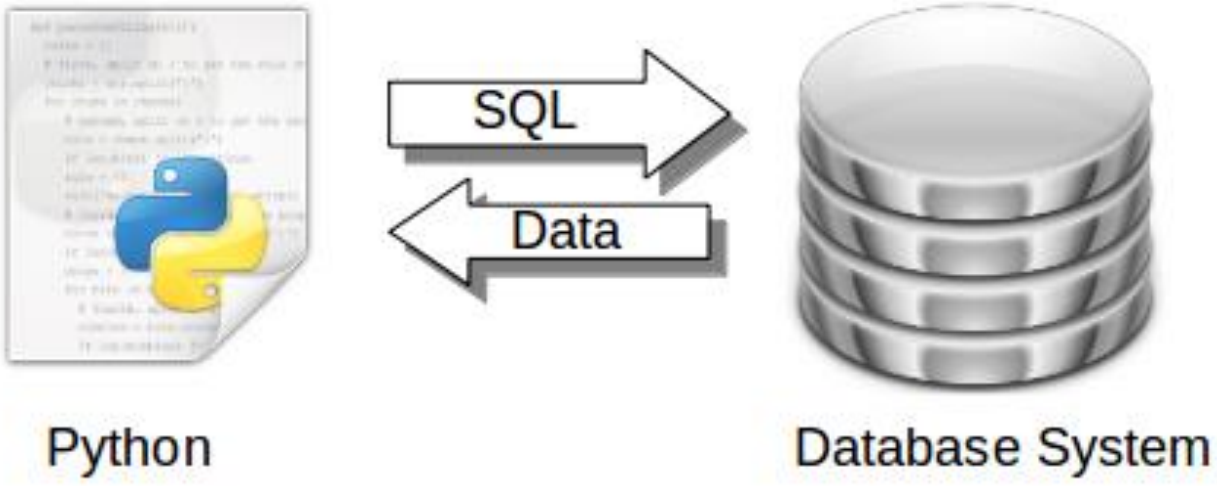
HARDWARES AND SOFTWARES REQUIRED

HARDWARES

1. Desktop Computer / Laptop
2. Mobile Phone

SOFTWARES

1. Python (latest version)
2. MySQL
3. Python Connector Module



CONTENTS

S.No.	Topic	Page No.
1	Certificate	1
2	Acknowledgement	2
3	Hardwares and Softwares Required	3
4	Introduction	5
5	Python Source Code	6
6	MySQL Database	41
7	Outputs	44
8	References	52

INTRODUCTION

The project **Library Management System** includes enrolment of users, adding of books into the library system. The software has the facility to search for news, wikipedia articles. It includes a authentication facility for admin and user to login into the admin panel and user panel resp. of the system. User can see the books available, details of books issued by the user in the digital library. The **Library Management System** can be login using a user ID and password. It is accessible either by an admin or user. Only the admin can add, delete and update the data of users and books into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast.

The purpose of the project entitled as “DIGITAL LIBRARY” is to computerize the Front Library Management to develop software which is user friendly, simple, fast, and cost-effective. It also has a notes facility where the user can add notes at any point of the program into the database.

Python Source Code

```
# Importing necessary libraries
import mysql.connector
import pyfiglet
import requests
import wikipediaapi
from datetime import datetime


# Connect to the MySQL database
db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="admin",
    database="library",
)
c = db.cursor()


# Function to display the return policy information
def returnPolicy():
    print("Return Policy : ")
    print("The issued book should be returned within 14 days(2 weeks).")
    print(
        "If the user kept the issued book for more than 14 days, then the
user have to pay ₹5 as fine for each extra day the user kept the issued
book."
    )
    print("---------------------------")


# Function to calculate the length of a given integer after converting it
to a string
def length(i):
    s = str(i)
    length = len(s) + 2

    return length


# Function to display a message for an invalid option
def validOption():
    print("Please enter a valid option!")
    print("---------------------------")


# Function to handle program exit
def exiting():
    print("\033[3;34m---------------------------\033[0;0m")
    print("\033[3;33mExiting the program.")
    print("Thank You!\033[0;0m")
    print("\033[3;34m---------------------------\033[0;0m")
    exit()


# Function to display the user menu and handle user choices
def userMenu():
    # Displaying options for the user
    print("1. Add Note")
    print("2. Home")
    print("3. Back")
    print("4. Exit")
    # Taking user choice as input
    userChoice = int(input("Enter your Choice to Continue : "))
    print("---------------------------")

    # Handle user choices
```

```
    if userChoice == 1:
        addNote()
    elif userChoice == 2:
        home()
    elif userChoice == 3:
        user()
    elif userChoice == 4:
        exiting()
    else:
        validOption()


# Function to display information about the library
def aboutLibrary():
    # Retrieve the name of the librarian who is also an admin
    c.execute("SELECT userName FROM users WHERE adminStatus='admin'")
    userName = c.fetchall()

    # Retrieve the total number of books and users in the library
    c.execute("SELECT * FROM books")
    totalBooks = c.fetchall()

    c.execute("SELECT * FROM users")
    totalUsers = c.fetchall()
    db.commit()

    print("--------------------------")
    print("About Library")
    print("--------------------------")
    # Display library information
    print("Year of Library's Establishment : ", 2023)
    print("Name of the Librarian : ", userName[0][0])
    print("Total Number of Books Available in the Library : ",
len(totalBooks))
    print("Total Number of Users Enrolled in the Library : ",
len(totalUsers))
    print("--------------------------")
    userMenu()


# Function to display the list of books in the library
def displayBooks():
    print("--------------------------")
    print("Display Books")
    print("--------------------------")
    # Retrieve all books from the database
    c.execute("SELECT * FROM books ORDER BY bookId")
    result = c.fetchall()
    db.commit()

    # Display books if available, otherwise notify the user
    if result:
        print("Books available in the Digital Library are :")
        print("--------------------------")
        i = 0
        for row in result:
            i += 1
            r = length(i)
            print(f"{i}. Book ID : {row[0]}")
            print(" " * r + f"Book Name : {row[1]}")
            print(" " * r + f"Publication Year : {row[2]}")
            print(" " * r + f"Author Name : {row[7]}")
            print(" " * r + f"Issue Status : {row[8]}")
            print("--------------------------")
        userMenu()
    else:
```

```
            # Notify the user if no books are found
            print("No books found.")
            print("--------------------------")
            userMenu()


    # Search books menu options
    def searchBooksMenu():
        print("1. Add Note")
        print("2. Home")
        print("3. Back")
        print("4. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))

        # User choices handling
        if userChoice == 1:
            addNote()
        elif userChoice == 2:
            home()
        elif userChoice == 3:
            searchBooks()
        elif userChoice == 4:
            exiting()
        else:
            validOption()


    # Function to search books by Book ID
    def searchBooksbyId():
        print("--------------------------")
        print("Search Books by Book ID")
        print("--------------------------")
        # Get user input for Book ID
        bookId = int(input("Enter the Book ID to search the Book : "))
        print("--------------------------")

        # Execute SQL query to retrieve book information by Book ID
        c.execute("SELECT * FROM books WHERE bookId=%s", (bookId,))
        result = c.fetchall()
        db.commit()

        # Display search results if books are found, otherwise notify the user
        if result:
            print(f'Book available in the Digital Library with the Book ID
    "{bookId}" is :')
            print("--------------------------")
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Book ID : {row[0]}")
                print(" " * r + f"Book Name : {row[1]}")
                print(" " * r + f"Publication Year : {row[2]}")
                print(" " * r + f"Author Name : {row[7]}")
                print(" " * r + f"Issue Status : {row[8]}")
                print("--------------------------")
            searchBooksMenu()
        else:
            print(f'No book found with the book id "{bookId}".')
            print("--------------------------")
            searchBooksMenu()


    # Function to search books by keyword
    def searchBooksbyKeyword():
        print("--------------------------")
```

```
        print("Search Books by Keyword")
        print("--------------------------")
        # Get user input for keyword
        keyword = input("Enter a Keyword to search Books : ")
        print("--------------------------")

        # Execute SQL query to retrieve books by keyword
        c.execute(
            "SELECT * FROM books WHERE bookName LIKE '%{}%' ORDER BY
bookId".format(keyword)
        )
        result = c.fetchall()
        db.commit()

        # Display search results if books are found, otherwise notify the user
        if result:
            print(
                f'Books available in the Digital Library with the Keyword
"{keyword}" are :'
            )
            print("--------------------------")
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Book ID : {row[0]}")
                print(" " * r + f"Book Name : {row[1]}")
                print(" " * r + f"Publication Year : {row[2]}")
                print(" " * r + f"Author Name : {row[7]}")
                print(" " * r + f"Issue Status : {row[8]}")
                print("--------------------------")
            searchBooksMenu()
        else:
            print(f'No books found with the keyword "{keyword}".')
            print("--------------------------")
            searchBooksMenu()


# Function to display search options for books
def searchBooks():
    print("--------------------------")
    print("Search Books")
    print("--------------------------")
    print("1. Search by Book ID")
    print("2. Search by Keyword")
    print("3. Home")
    print("4. Back")
    print("5. Exit")
    userChoice = int(input("Enter your Choice to Continue : "))
    print("--------------------------")

    # User choices handling
    if userChoice == 1:
        searchBooksbyId()
    elif userChoice == 2:
        searchBooksbyKeyword()
    elif userChoice == 3:
        home()
    elif userChoice == 4:
        user()
    elif userChoice == 5:
        exiting()
    else:
        validOption()
```

```
265 # Function to display the add book menu and handle user choices
266 def addBookMenu():
267     # Add book menu options
268     print("1. Home")
269     print("2. Back")
270     print("3. Exit")
271     userChoice = int(input("Enter your Choice to Continue : "))
272     print("--------------------------")
273 
274     # User choices handling
275     if userChoice == 1:
276         home()
277     elif userChoice == 2:
278         modifyBook()
279     elif userChoice == 3:
280         exiting()
281     else:
282         validOption()
283 
284 
285 # Function to add a new book to the library
286 def addBook():
287     print("--------------------------")
288     print("Add Book")
289     print("--------------------------")
290     # Get user input for book details
291     bookId = int(input("Enter the Book ID : "))
292     bookName = input("Enter the Book Name : ")
293     publicationYear = int(input("Enter the Book Publication Year : "))
294     author = input("Enter the Book Author Name : ")
295     print("--------------------------")
296 
297     c.execute("SELECT bookId FROM books")
298     result = c.fetchall()
299     db.commit()
300 
301     if (bookId,) in result:
302         print(
303             f'The book of book id "{bookId}" is already available in the
304 digital library.'
305         )
306         print("--------------------------")
307         addBookMenu()
308     else:
309         # Execute SQL query to insert the new book into the database
310         c.execute(
311             "INSERT INTO books (bookId, bookName, publicationYear, author)
312 VALUES (%s, %s, %s, %s)",
313             (bookId, bookName, publicationYear, author),
314         )
315         db.commit()
316 
317         # Notify the user that the book has been added successfully
318         print("Book added Successfully!")
319         print("--------------------------")
320         addBookMenu()
321 
322 
323 # Function to display the delete book menu and handle user choices
324 def deleteBookMenu():
325     # Delete book menu options
326     print("1. Home")
327     print("2. Back")
328     print("3. Exit")
329     userChoice = int(input("Enter your Choice to Continue : "))
330     print("--------------------------")
```

```
331
332         # User choices handling
333         if userChoice == 1:
334             home()
335         elif userChoice == 2:
336             admin()
337         elif userChoice == 3:
338             exiting()
339         else:
340             validOption()
341
342
343     # Function to delete a book from the library
344     def deleteBook():
345         print("--------------------------")
346         print("Delete Book")
347         print("--------------------------")
348         # Get user input for the book ID to be deleted
349         bookId = int(input("Enter the Book ID : "))
350         choice = input("Are you sure to delete the Book? (Yes/No) : ")
351         print("--------------------------")
352
353         c.execute("SELECT bookId FROM books")
354         result = c.fetchall()
355         db.commit()
356
357         if choice.lower() in ["yes", "y"]:
358             if (bookId,) in result:
359                 # Execute SQL query to delete the book from the database
360                 c.execute("DELETE FROM books WHERE bookId=%s", (bookId,))
361                 db.commit()
362
363                 # Notify the user that the book has been deleted successfully
364                 print("Book deleted Successfully!")
365                 print("--------------------------")
366                 deleteBookMenu()
367             else:
368                 print(
369                     f'The book of book id "{bookId}" does not available in the
370     digital library.'
371                 )
372                 print("--------------------------")
373                 deleteBookMenu()
374         elif choice.lower() in ["no", "n"]:
375             print("--------------------------")
376             print("Book Not Deleted!")
377             print("--------------------------")
378             deleteBookMenu()
379         else:
380             validOption()
381
382
383     # Update book menu options
384     def updateBookMenu():
385         print("1. Home")
386         print("2. Back")
387         print("3. Exit")
388         userChoice = int(input("Enter your Choice to Continue : "))
389         print("--------------------------")
390
391         # User choices handling
392         if userChoice == 1:
393             home()
394         elif userChoice == 2:
395             updateUser()
396         elif userChoice == 3:
```

```
            exiting()
        else:
            validOption()


    def notBook(bookId):
        print(f'The book of book id "{bookId}" does not available in the
    digital library.')
        print("--------------------------")
        updateBookMenu()


    # Function to update book details
    def updateBook():
        print("--------------------------")
        print("Update Book Details")
        print("--------------------------")
        print("1. Update the Book ID")
        print("2. Update the Book Name")
        print("3. Update the Book Publication Year")
        print("4. Update the Book Author Name")
        print("5. Home")
        print("6. Back")
        print("7. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        c.execute("SELECT bookId FROM books")
        result = c.fetchall()
        db.commit()

        # User choices handling
        if userChoice == 1:
            currentBookId = int(input("Enter the Current Book ID : "))
            newBookId = int(input("Enter the New Book ID : "))

            if (currentBookId,) in result:
                # Execute SQL query to update the Book ID
                c.execute(
                    "UPDATE books SET bookId=%s WHERE bookId=%s", (newBookId,
    currentBookId)
                )
                db.commit()

                print("Book ID changed Successfully!")
                print("--------------------------")
                updateBookMenu()
            else:
                notBook(currentBookId)

        elif userChoice == 2:
            bookId = int(input("Enter the Book ID : "))
            newBookName = input("Enter the New Book Name : ")

            if (bookId,) in result:
                # Execute SQL query to update the Book Name
                c.execute(
                    "UPDATE books SET bookName=%s WHERE bookId=%s",
    (newBookName, bookId)
                )
                db.commit()

                print("Book Name changed Successfully!")
                print("--------------------------")
                updateBookMenu()
            else:
```

```
463                 notBook(bookId)
464
465         elif userChoice == 3:
466             bookId = int(input("Enter the Current Book ID : "))
467             newPublicationYear = input("Enter the New Publication Year : ")
468
469             if (bookId,) in result:
470                 # Execute SQL query to update the Publication Year
471                 c.execute(
472                     "UPDATE books SET publicationYear=%s WHERE bookId=%s",
473                     (newPublicationYear, bookId),
474                 )
475                 db.commit()
476
477                 print("Book Publication Year changed Successfully!")
478                 print("--------------------------")
479                 updateBookMenu()
480
481         elif userChoice == 4:
482             bookId = int(input("Enter the Current Book ID : "))
483             newAuthor = input("Enter the New Author Name : ")
484
485             if (bookId,) in result:
486                 # Execute SQL query to update the Author Name
487                 c.execute(
488                     "UPDATE books SET author=%s WHERE bookId=%s",
489                     (newAuthor, bookId),
490                 )
491                 db.commit()
492
493                 print("Book Author Name changed Successfully!")
494                 print("--------------------------")
495                 updateBookMenu()
496             else:
497                 notBook(bookId)
498
499         elif userChoice == 5:
500             home()
501         elif userChoice == 6:
502             modifyBook()
503         elif userChoice == 7:
504             exiting()
505         else:
506             validOption()
507
508
509     # Function to display the issue book menu and handle user choices
510     def issueBookMenu():
511         print("1. Home")
512         print("2. Back")
513         print("3. Exit")
514         userChoice = int(input("Enter your Choice to Continue : "))
515         print("--------------------------")
516
517         # User choices handling
518         if userChoice == 1:
519             home()
520         elif userChoice == 2:
521             admin()
522         elif userChoice == 3:
523             exiting()
524         else:
525             validOption()
526
527
528     # Function to issue a book
```

```
def issueBook():
    print("--------------------------")
    print("Issue Book")
    print("--------------------------")
    bookId = int(input("Enter the Book ID to be Issued: "))
    userId = int(input("Enter the User ID to whom Book will be Issued: "))

    # Execute SQL query to check the issue status of the book
    c.execute("SELECT userId FROM users")
    result1 = c.fetchall()
    c.execute("SELECT bookId FROM books")
    result2 = c.fetchall()
    c.execute("SELECT issueStatus FROM books WHERE bookId=%s", (bookId,))
    result3 = c.fetchall()
    db.commit()

    if (userId,) in result1:
        if (bookId,) in result2:
            # Check if the book is not already issued
            if result3[0][0] == "not issued":
                # Execute SQL queries to update book details and mark it as
issued
                c.execute(
                    "UPDATE books SET issueDate = CURRENT_DATE WHERE bookId
= %s",
                    (bookId,),
                )
                c.execute(
                    "UPDATE books SET issueTime = CURRENT_TIME WHERE bookId
= %s",
                    (bookId,),
                )
                c.execute(
                    "UPDATE books SET issueStatus = 'issued' WHERE bookId =
%s",
                    (bookId,),
                )
                c.execute(
                    "UPDATE books SET returnDate = NULL WHERE bookId = %s",
(bookId,)
                )
                c.execute(
                    "UPDATE books SET returnTime = NULL WHERE bookId = %s",
(bookId,)
                )
                c.execute(
                    "UPDATE books SET issuedUserId = %s WHERE bookId = %s",
                    (userId, bookId),
                )
                db.commit()
                c.execute(
                    "select issuedUserId,bookName,issueDate,issueTime from
books where bookId=%s",
                    (bookId,),
                )
                result = c.fetchall()
                c.execute(
                    "INSERT INTO issuedBooksDetails (userId,
bookId,bookName,issueDate,issueTime) VALUES (%s, %s, %s, %s, %s)",
                    (result[0][0], bookId, result[0][1], result[0][2],
result[0][3]),
                )
                db.commit()

                print("--------------------------")
                print(
```

```
                    f'Book of Book Id "{bookId}" is issued successfully to
the User of User Id "{userId}".'
                )
                print("--------------------------")
                returnPolicy()
                issueBookMenu()
            else:
                # Notify the user that the book is already issued
                print(
                    f'The book of book id "{bookId}" is already issued by
another user.'
                )
                print("--------------------------")
                issueBookMenu()
        else:
            print(
                f"Book with book id {bookId} does not available in the
digital library."
            )
            print("--------------------------")
            issueBookMenu()
    else:
        print(f"User with user id {userId} does not exists in the digital
library.")
        print("--------------------------")
        issueBookMenu()


# Function to display the return book menu and handle user choices
def returnBookMenu():
    print("1. Home")
    print("2. Back")
    print("3. Exit")
    userChoice = int(input("Enter your Choice to Continue : "))
    print("--------------------------")

    # User choices handling
    if userChoice == 1:
        home()
    elif userChoice == 2:
        admin()
    elif userChoice == 3:
        exiting()
    else:
        validOption()


# Function to return a book
def returnBook():
    print("--------------------------")
    print("Return Book")
    print("--------------------------")
    bookId = int(input("Enter the Book ID to be Returned: "))

    # Execute SQL query to check the issue status of the book
    c.execute("SELECT bookId FROM books")
    result1 = c.fetchall()
    c.execute("SELECT issueStatus FROM books WHERE bookId=%s", (bookId,))
    result2 = c.fetchall()

    db.commit()

    if (bookId,) in result1:
        # Check if the book is issued
        if result2[0][0] == "issued":
            # Execute SQL queries to update book details and mark it as
```

```
returned
                c.execute(
                    "UPDATE books SET returnDate = CURRENT_DATE WHERE bookId =
%s",
                    (bookId,),
                )
                c.execute(
                    "UPDATE books SET returnTime = CURRENT_TIME WHERE bookId =
%s",
                    (bookId,),
                )
                c.execute(
                    "UPDATE books SET issueStatus = 'not issued' WHERE bookId =
%s",
                    (bookId,),
                )
                db.commit()
                c.execute(
                    "select issuedUserId,returnDate,returnTime from books where
bookId=%s",
                    (bookId,),
                )
                result = c.fetchall()
                c.execute(
                    "UPDATE issuedBooksDetails SET returnDate = %s, returnTime
= %s WHERE userId = %s AND bookId = %s",
                    (result[0][1], result[0][2], result[0][0], bookId),
                )

                db.commit()
                c.execute(
                    "UPDATE books SET issuedUserId = NULL WHERE bookId = %s",
(bookId,)
                )
                db.commit()

                print(f'The book of book id "{bookId}" is returned
successfully.')

                c.execute("select issueDate from books WHERE bookId = %s",
(bookId,))
                issueDate = c.fetchall()
                c.execute("select returnDate from books WHERE bookId = %s",
(bookId,))
                returnDate = c.fetchall()
                db.commit()

                c.execute("UPDATE books SET issueDate = NULL WHERE bookId =
%s", (bookId,))
                c.execute("UPDATE books SET issueTime = NULL WHERE bookId =
%s", (bookId,))
                c.execute("UPDATE books SET returnDate = NULL WHERE bookId =
%s", (bookId,))
                c.execute("UPDATE books SET returnTime = NULL WHERE bookId =
%s", (bookId,))
                db.commit()

                d1 = datetime.strptime(f"{issueDate[0][0]}", "%Y-%m-%d")
                d2 = datetime.strptime(f"{returnDate[0][0]}", "%Y-%m-%d")
                dateDifference = d1 - d2

                if dateDifference.days > 14:
                    extraDays = dateDifference.days - 14
                    fine = extraDays * 5
                    print("Fine(in Rs.) : ", fine)
                    c.execute(
```

```
                    "update issuedBooksDetails set fineInRs=%s where
userId=%s and bookId=%s",
                    (fine, result[0][0], bookId),
                )
                db.commit()
            else:
                fine = 0 * 5
                print("Fine(in Rs.) : ", fine)
                c.execute(
                    "update issuedBooksDetails set fineInRs=%s where
userId=%s and bookId=%s",
                    (fine, result[0][0], bookId),
                )
                db.commit()

                print("--------------------------")
                returnBookMenu()
        else:
            # Notify the user that the book is not issued
            print(f'The book of book id "{bookId}" is not issued by any
user.')
            print("--------------------------")
            returnBookMenu()
    else:
        print(f"Book with book id {bookId} does not available in the
digital library.")
        print("--------------------------")
        returnBookMenu()


# Function to display the add user menu and handle user choices
def addUserMenu():
    # Add user menu options
    print("1. Home")
    print("2. Back")
    print("3. Exit")
    userChoice = int(input("Enter your Choice to Continue : "))
    print("--------------------------")

    # User choices handling
    if userChoice == 1:
        home()
    elif userChoice == 2:
        modifyUser()
    elif userChoice == 3:
        exiting()
    else:
        validOption()


# Function to add a new user
def addUser():
    print("--------------------------")
    print("Add User")
    print("--------------------------")
    # Get user input for user details
    userId = int(input("Enter the User ID : "))
    userName = input("Enter the User Name : ")
    userPhoneNumber = input("Enter the User Phone Number : ")
    userEmailId = input("Enter the User Email ID : ")
    password = input("Enter the User Password : ")
    print("--------------------------")

    c.execute("SELECT userId FROM users")
    result = c.fetchall()
    db.commit()
```

```
793
794         if (userId,) in result:
795             print(
796                 f'The user of user number "{userId}" is already enrolled in the
797     digital library.'
798             )
799             print("--------------------------")
800             addUserMenu()
801         else:
802             # Execute SQL query to insert the new user into the database
803             c.execute(
804                 "INSERT INTO users (userId, userName, phoneNumber, emailId,
805     password) VALUES (%s, %s, %s, %s, %s)",
806                 (userId, userName, userPhoneNumber, userEmailId, password),
807             )
808             db.commit()
809
810             # Notify the user that the user has been added successfully
811             print("--------------------------")
812             print("User added successfully!")
813             print("--------------------------")
814             addUserMenu()
815
816
817     # Function to display the delete user menu and handle user choices
818     def deleteUserMenu():
819         # Delete user menu options
820         print("1. Home")
821         print("2. Back")
822         print("3. Exit")
823         userChoice = int(input("Enter your Choice to Continue : "))
824         print("--------------------------")
825
826         # User choices handling
827         if userChoice == 1:
828             home()
829         elif userChoice == 2:
830             modifyUser()
831         elif userChoice == 3:
832             exiting()
833         else:
834             validOption()
835
836
837     # Function to delete a user
838     def deleteUser():
839         print("--------------------------")
840         print("Delete User")
841         print("--------------------------")
842         # Get user input for the user ID to be deleted
843         userId = int(input("Enter the User ID : "))
844         choice = input("Are you sure to delete the User? (Yes/No) : ")
845
846         c.execute("SELECT userId FROM users")
847         result = c.fetchall()
848         db.commit()
849
850         if choice.lower() in ["yes", "y"]:
851             if (userId,) in result:
852                 c.execute("DELETE FROM users WHERE userId=%s", (userId,))
853                 db.commit()
854
855                 # Notify the user that the user has been deleted successfully
856                 print("User deleted successfully!")
857                 print("--------------------------")
858                 deleteUserMenu()
```

```
            else:
                print(
                    f'The user of user id "{userId}" does not enrolled in the
digital library.'
                )
                print("--------------------------")
                deleteUserMenu()
        elif choice.lower() in ["no", "n"]:
            print("--------------------------")
            print("User Not Deleted!")
            print("--------------------------")
            deleteUserMenu()
        else:
            validOption()


    # Function to display the update user menu and handle user choices
    def updateUserMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))

        # User choices handling
        if userChoice == 1:
            home()
        elif userChoice == 2:
            updateUser()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    def notUser(userId):
        print(f'The user of user id "{userId}" does not enrolled in the digital
library.')
        print("--------------------------")
        updateBookMenu()


    # Function to update user details
    def updateUser():
        print("--------------------------")
        print("Update User Details")
        print("--------------------------")
        # Display user update options
        print("1. Update the User ID")
        print("2. Update the User Name")
        print("3. Update the User Phone Number")
        print("4. Update the User Email ID")
        print("5. Update the User Password")
        print("6. Home")
        print("7. Back")
        print("8. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        c.execute("SELECT userId FROM users")
        result = c.fetchall()
        db.commit()

        if userChoice == 1:
            # Update user ID
            currentUserId = int(input("Enter the Current User ID : "))
```

```
            newUserId = int(input("Enter the New User ID : "))

            if (currentUserId,) in result:
                c.execute(
                    "update users set userId=%s where userId=%s", (newUserId,
currentUserId)
                )
                db.commit()

                print("User ID changed Successfully!")
                print("--------------------------")
                updateUserMenu()
            else:
                notUser(currentUserId)

        elif userChoice == 2:
            # Update user name
            userId = int(input("Enter the User ID : "))
            newUserName = input("Enter the New User Name : ")

            if (userId,) in result:
                c.execute(
                    "update users set userName=%s where userId=%s",
(newUserName, userId)
                )
                db.commit()

                print("User Name changed Successfully!")
                print("--------------------------")
                updateUserMenu()
            else:
                notUser(userId)

        elif userChoice == 3:
            # Update user phone number
            userId = int(input("Enter the Current User ID : "))
            newPhoneNumber = input("Enter the New Phone Number : ")

            if (userId,) in result:
                c.execute(
                    "update users set phoneNumber=%s where userId=%s",
                    (newPhoneNumber, userId),
                )
                db.commit()

                print("User Phone Number changed Successfully!")
                print("--------------------------")
                updateUserMenu()
            else:
                notUser(userId)

        elif userChoice == 4:
            # Update user email ID
            userId = int(input("Enter the Current User ID : "))
            newEmailId = input("Enter the New Email ID : ")

            if (userId,) in result:
                c.execute(
                    "update users set emailId=%s where userId=%s", (newEmailId,
userId)
                )
                db.commit()

                print("User Email ID changed Successfully!")
                print("--------------------------")
                updateUserMenu()
```

```
            else:
                notUser(userId)

        elif userChoice == 5:
            # Update user password
            userId = int(input("Enter the Current User ID : "))
            newPassword = input("Enter the New Password : ")
            if (userId,) in result:
                c.execute(
                    "update users set password=%s where userId=%s",
    (newPassword, userId)
                )
                db.commit()

                print("User Password changed Successfully!")
                print("--------------------------")
                updateUserMenu()
            else:
                notUser(userId)

        elif userChoice == 6:
            # Return to home
            home()
        elif userChoice == 7:
            # Go back to the previous menu
            modifyUser()
        elif userChoice == 8:
            # Exit the program
            exiting()
        else:
            validOption()


    # Function to modify user
    def modifyUser():
        print("--------------------------")
        print("Modify User")
        print("--------------------------")
        # Display user modification options
        print("1. Add User")
        print("2. Delete User")
        print("3. Update User Details")
        print("4. Home")
        print("5. Back")
        print("6. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # User choices handling
        if userChoice == 1:
            # Add a new user
            addUser()
        elif userChoice == 2:
            # Delete a user
            deleteUser()
        elif userChoice == 3:
            # Update user details
            updateUser()
        elif userChoice == 4:
            # Return to home
            home()
        elif userChoice == 5:
            # Return to the previous menu
            admin()
        elif userChoice == 6:
```

```
            # Exit the program
            exiting()
        else:
            validOption()


    # Display users menu options
    def displayUsersMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))

        # User choices handling
        if userChoice == 1:
            home()
        elif userChoice == 2:
            admin()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to display all users
    def displayUsers():
        print("--------------------------")
        print("Display Users")
        print("--------------------------")
        # Fetch all users from the database
        c.execute("SELECT * FROM users ORDER BY userId")
        result = c.fetchall()
        db.commit()

        if result:
            # Display user information
            print("Users enrolled in the Digital Library are :")
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. User ID : {row[0]}")
                print(" " * r + f"User Name : {row[1]}")
                print(" " * r + f"Phone Number : {row[2]}")
                print(" " * r + f"Email ID : {row[3]}")
                print(" " * r + f"Admin Status : {row[5]}")
                print("--------------------------")
            displayUsersMenu()

        else:
            print("No users found.")
            print("--------------------------")
            displayUsersMenu()


    # Search user menu options
    def searchUsersMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))

        # User choices handling
        if userChoice == 1:
            home()
        elif userChoice == 2:
```

```
            searchUsers()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to search users by ID
    def searchUsersbyId():
        print("--------------------------")
        print("Search Users by User ID")
        print("--------------------------")
        # Get user ID to search
        userId = int(input("Enter the User ID to search the User : "))

        # Search for the user in the database
        c.execute("SELECT * FROM users WHERE userId=%s", (userId,))
        result = c.fetchall()
        db.commit()

        if result:
            # Display user information if found
            print(f'User enrolled in the Digital Library with the User ID
    "{userId}" is :')
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. User ID : {row[0]}")
                print(" " * r + f"User Name : {row[1]}")
                print(" " * r + f"Phone Number : {row[2]}")
                print(" " * r + f"Email ID : {row[3]}")
                print(" " * r + f"Admin Status : {row[5]}")
                print("--------------------------")
            searchUsersMenu()

        else:
            # Handle case when no user is found
            print(f'No user found with the user id "{userId}".')
            print("--------------------------")
            searchUsersMenu()


    # Function to search users by keyword
    def searchUsersbyKeyword():
        print("--------------------------")
        print("Search Users by Keyword")
        print("--------------------------")
        # Get keyword input from the user
        keyword = input("Enter a Keyword to search Users : ")

        # Search for users with the given keyword in their names
        c.execute(
            "SELECT * FROM users WHERE userName LIKE '%{}%' ORDER BY
    userId".format(keyword)
        )
        result = c.fetchall()
        db.commit()

        if result:
            # Display user information if users are found
            print(
                f'Users enrolled in the Digital Library with the Keyword
    "{keyword}" are :'
            )
            i = 0
```

```
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. User ID : {row[0]}")
                print(" " * r + f"User Name : {row[1]}")
                print(" " * r + f"Phone Number : {row[2]}")
                print(" " * r + f"Email ID : {row[3]}")
                print(" " * r + f"Admin Status : {row[5]}")
                print("--------------------------")
            searchUsersMenu()

        else:
            # Handle case when no user is found
            print(f'No users found with the keyword "{keyword}".')
            print("--------------------------")
            searchUsersMenu()


    # Function to search users
    def searchUsers():
        print("--------------------------")
        print("Search Users")
        print("--------------------------")
        # User search menu
        print("1. Search by User ID")
        print("2. Search by Keyword")
        print("3. Home")
        print("4. Back")
        print("5. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # User choices handling
        if userChoice == 1:
            searchUsersbyId()
        elif userChoice == 2:
            searchUsersbyKeyword()
        elif userChoice == 3:
            home()
        elif userChoice == 4:
            admin()
        elif userChoice == 5:
            exiting()
        else:
            validOption()


    # Function to modify books
    def modifyBook():
        print("--------------------------")
        print("Modify Book")
        print("--------------------------")
        # Book modification menu
        print("1. Add Book")
        print("2. Delete Book")
        print("3. Update Book Details")
        print("4. Home")
        print("5. Back")
        print("6. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # User choices handling
        if userChoice == 1:
            addBook()
        elif userChoice == 2:
```

```
            deleteBook()
        elif userChoice == 3:
            updateBook()
        elif userChoice == 4:
            home()
        elif userChoice == 5:
            admin()
        elif userChoice == 6:
            exiting()
        else:
            validOption()


    # Function to manage notes
    def notes():
        print("---------------------------")
        print("Notes")
        print("---------------------------")
        # Display menu options
        print("1. Modify Note")
        print("2. Display Notes")
        print("3. Search Notes")
        print("4. Home")
        print("5. Back")
        print("6. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("---------------------------")

        # Handle user choices
        if userChoice == 1:
            modifyNote()
        elif userChoice == 2:
            displayNotes()
        elif userChoice == 3:
            searchNotes()
        elif userChoice == 4:
            home()
        elif userChoice == 5:
            user()
        elif userChoice == 6:
            exiting()
        else:
            validOption()


    # Function to display the add note menu and handle user choices
    def addNoteMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))

        # Handle user choices
        if userChoice == 1:
            home()
        elif userChoice == 2:
            modifyNote()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to add note
```

```
def addNote():
    print("--------------------------")
    print("Add Note")
    print("--------------------------")
    # Get note details from the user
    noteNumber = int(input("Enter the Note Number : "))
    noteTitle = input("Enter the Note Title : ")
    noteDescription = input("Enter the Note Description : ")
    print("--------------------------")

    c.execute("SELECT noteNumber FROM notes where userId=%s", (USERID,))
    result = c.fetchall()
    db.commit()

    if (noteNumber,) in result:
        print(
            f'The note of note number "{noteNumber}" is already exists in
the digital library.'
        )
        print("--------------------------")
        addNoteMenu()

    else:
        # Execute SQL query to insert the note into the database
        c.execute(
            "INSERT INTO notes (userId, noteNumber, noteTitle,
noteDescription, updateDate, updateTime) VALUES (%s, %s, %s, %s,
CURRENT_DATE, CURRENT_TIME)",
            (USERID, noteNumber, noteTitle, noteDescription),
        )
        db.commit()

        print(f'The note of note number "{noteNumber}" is added
successfully.')
        print("--------------------------")
        addNoteMenu()


# Function to display the delete note menu and handle user choices
def deleteNoteMenu():
    # Display menu options after deleting the note
    print("1. Home")
    print("2. Back")
    print("3. Exit")
    # Get user choice
    userChoice = int(input("Enter your Choice to Continue : "))
    print("--------------------------")

    # Handle user choices
    if userChoice == 1:
        home()
    elif userChoice == 2:
        modifyNote()
    elif userChoice == 3:
        exiting()
    else:
        validOption()


# Function to delete a note
def deleteNote():
    print("--------------------------")
    print("Delete Note")
    print("--------------------------")
    # Get note number to be deleted from the user
    noteNumber = int(input("Enter the Note Number to Delete the Note : "))
```

```
        choice = input("Are you sure to delete the Note? (Yes/No) : ")
        print("--------------------------")

        c.execute("SELECT noteNumber FROM notes where userId=%s", (USERID,))
        result = c.fetchall()
        db.commit()

        if choice.lower() in ["yes", "y"]:
            if (noteNumber,) in result:
                # Execute SQL query to delete the note from the database
                c.execute(
                    "delete FROM notes WHERE userId=%s and noteNumber=%s",
                    (USERID, noteNumber),
                )
                db.commit()

                print(f'The note of note number "{noteNumber}" is deleted
successfully.')
                print("--------------------------")
                deleteNoteMenu()

            else:
                print(
                    f'The note of note number "{noteNumber}" does not exists in
the digital library.'
                )
                print("--------------------------")
                deleteNoteMenu()
        elif choice.lower() in ["no", "n"]:
            print("--------------------------")
            print("Note Not Deleted!")
            print("--------------------------")
            deleteNoteMenu()
        else:
            validOption()


    # Function to display the update notes menu and handle user choices
    def updateNotesMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # Handle user choices
        if userChoice == 1:
            home()
        elif userChoice == 2:
            updateNotes()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    def notNote(noteNumber):
        print(
            f'The note of note number "{noteNumber}" does not exists in the
digital library.'
        )
        print("--------------------------")
        updateNotesMenu()
```

```
1453    # Function to update a note
1454    def updateNotes():
1455        print("--------------------------")
1456        print("Update Notes")
1457        print("--------------------------")
1458        # Display update options
1459        print("1. Update the Note Number")
1460        print("2. Update the Note Title")
1461        print("3. Update the Note Description")
1462        print("4. Home")
1463        print("5. Back")
1464        print("6. Exit")
1465        # Get user choice
1466        userChoice = int(input("Enter your Choice to Continue : "))
1467        print("--------------------------")
1468
1469        c.execute("SELECT noteNumber FROM notes where userId=%s", (USERID,))
1470        result = c.fetchall()
1471        db.commit()
1472
1473        # Handle user choices
1474        if userChoice == 1:
1475            # Update Note Number
1476            currentNoteNumber = int(input("Enter the Current Note Number : "))
1477            newNoteNumber = int(input("Enter the New Note Number : "))
1478
1479            if (currentNoteNumber,) in result:
1480                # Update date and time
1481                c.execute(
1482                    "update notes set updateDate=CURRENT_DATE where userId=%s
1483    and noteNumber=%s",
1484                    (USERID, currentNoteNumber),
1485                )
1486                c.execute(
1487                    "update notes set updateTime=CURRENT_TIME where userId=%s
1488    and noteNumber=%s",
1489                    (USERID, currentNoteNumber),
1490                )
1491                # Update Note Number
1492                c.execute(
1493                    "update notes set noteNumber=%s where userId=%s and
1494    noteNumber=%s",
1495                    (newNoteNumber, USERID, currentNoteNumber),
1496                )
1497                db.commit()
1498
1499                print("Note Number changed Successfully!")
1500                print("--------------------------")
1501                updateNotesMenu()
1502            else:
1503                notNote(currentNoteNumber)
1504
1505        elif userChoice == 2:
1506            # Update Note Title
1507            noteNumber = int(input("Enter the Current Note Number : "))
1508            newTitle = input("Enter the New Note Title : ")
1509
1510            if (noteNumber,) in result:
1511                # Update date and time
1512                c.execute(
1513                    "update notes set updateDate=CURRENT_DATE where userId=%s
1514    and noteNumber=%s",
1515                    (USERID, noteNumber),
1516                )
1517                c.execute(
1518                    "update notes set updateTime=CURRENT_TIME where userId=%s
```

```
1519    and noteNumber=%s",
1520                        (USERID, noteNumber),
1521                )
1522                # Update Note Title
1523                c.execute(
1524                    "update notes set noteTitle=%s where userId=%s and
1525    noteNumber=%s",
1526                        (newTitle, USERID, noteNumber),
1527                )
1528                db.commit()
1529
1530                print("Note Title changed Successfully!")
1531                print("--------------------------")
1532                updateNotesMenu()
1533            else:
1534                notNote(noteNumber)
1535
1536        elif userChoice == 3:
1537            # Update Note Description
1538            noteNumber = int(input("Enter the Current Note Number : "))
1539            newDescription = input("Enter the New Note Description : ")
1540
1541            if (noteNumber,) in result:
1542                # Update date and time
1543                c.execute(
1544                    "update notes set updateDate=CURRENT_DATE where userId=%s
1545    and noteNumber=%s",
1546                        (USERID, noteNumber),
1547                )
1548                c.execute(
1549                    "update notes set updateTime=CURRENT_TIME where userId=%s
1550    and noteNumber=%s",
1551                        (USERID, noteNumber),
1552                )
1553                # Update Note Description
1554                c.execute(
1555                    "update notes set noteDescription=%s where userId=%s and
1556    noteNumber=%s",
1557                        (newDescription, USERID, noteNumber),
1558                )
1559                db.commit()
1560
1561                print("Note Description changed successfully!")
1562                print("--------------------------")
1563                updateNotesMenu()
1564            else:
1565                notNote(noteNumber)
1566
1567        elif userChoice == 5:
1568            home()
1569        elif userChoice == 6:
1570            modifyNote()
1571        elif userChoice == 7:
1572            exiting()
1573        else:
1574            validOption()
1575
1576
1577    # Function to handle note modifications
1578    def modifyNote():
1579        print("--------------------------")
1580        print("Modify Notes")
1581        print("--------------------------")
1582        # Display modification options
1583        print("1. Add Note")
1584        print("2. Delete Note")
```

```
        print("3. Update Notes")
        print("4. Home")
        print("5. Back")
        print("6. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # Handle user choices
        if userChoice == 1:
            addNote()
        elif userChoice == 2:
            deleteNote()
        elif userChoice == 3:
            updateNotes()
        elif userChoice == 4:
            home()
        elif userChoice == 5:
            admin()
        elif userChoice == 6:
            exiting()
        else:
            validOption()


    # Function to display the display notes menu and handle user choices
    def displayNotesMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # Handle user choices
        if userChoice == 1:
            home()
        elif userChoice == 2:
            user()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to display notes
    def displayNotes():
        # Fetch all notes from the database
        c.execute("SELECT * FROM notes ORDER BY noteNumber")
        result = c.fetchall()
        db.commit()

        # Check if there are notes available
        if result:
            print(f"Notes available in the Digital Library are :")
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Note Number : {row[1]}")
                print(" " * r + f"Note Title : {row[2]}")
                print(" " * r + f"Note Description : {row[3]}")
                print(" " * r + f"Update Date : {row[4]}")
                print(" " * r + f"Update Time : {row[5]}")
                print("--------------------------")
            displayNotesMenu()
```

```
        else:
            # If no notes are found
            print("No notes found.")
            print("--------------------------")
            displayNotesMenu()


    # Function to display the search notes menu and handle user choices
    def searchNotesMenu():
        print("1. Home")
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))

        # Handle user choices
        if userChoice == 1:
            home()
        elif userChoice == 2:
            searchNotes()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to search notes by note number
    def searchNotesbynoteNumber():
        # Get the note number to search
        noteNumber = int(input("Enter the Note Number to search the Note : "))

        # Execute SQL query to fetch notes with the given note number
        c.execute("SELECT * FROM notes WHERE bookId=%s", (noteNumber,))
        result = c.fetchall()
        db.commit()

        # Check if notes are found
        if result:
            print(
                f'Note available in the Digital Library with the Note Number
    "{noteNumber}" is :'
            )
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Note Number : {row[1]}")
                print(" " * r + f"Note Title : {row[2]}")
                print(" " * r + f"Note Description : {row[3]}")
                print("--------------------------")
            searchNotesMenu()

        else:
            # If no notes are found with the given note number
            print(f'No note found with the note number "{noteNumber}".')
            print("--------------------------")
            searchNotesMenu()


    # Function to search notes by keyword
    def searchNotesbyKeyword():
        print("--------------------------")
        print("Search Notes by Keyword")
        print("--------------------------")
        # Get keyword from user
        keyword = input("Enter a Keyword to search Notes : ")
```

```
        # Execute SQL query to fetch notes with the given keyword in the title
        c.execute(
            "SELECT * FROM notes WHERE noteTitle LIKE '%{}%' ORDER BY
    noteNumber".format(
                keyword
            )
        )
        result = c.fetchall()
        db.commit()

        # Check if notes are found
        if result:
            print(
                f'Notes available in the Digital Library with the Keyword
    "{keyword}" are :'
            )
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Note Number : {row[1]}")
                print(" " * r + f"Note Title : {row[2]}")
                print(" " * r + f"Note Description : {row[3]}")
                print("---------------------------")
            searchNotesMenu()

        else:
            # If no notes are found with the given keyword
            print(f'No notes found with the keyword "{keyword}".')
            print("---------------------------")
            searchNotesMenu()


    # Function to handle note searching
    def searchNotes():
        print("---------------------------")
        print("Search Notes")
        print("---------------------------")
        # Display search options
        print("1. Search by Note Number")
        print("2. Search by Keyword")
        print("3. Home")
        print("4. Back")
        print("5. Exit")
        # Get user choice
        userChoice = int(input("Enter your Choice to Continue : "))
        print("---------------------------")

        # Handle user choices
        if userChoice == 1:
            searchNotesbynoteNumber()
        elif userChoice == 2:
            searchNotesbyKeyword()
        elif userChoice == 3:
            notes()
        elif userChoice == 4:
            modifyNote()
        elif userChoice == 5:
            exiting()
        else:
            validOption()


    # Function to display the change admin menu and handle user choices
    def changeAdminMenu():
        print("1. Home")
```

```
        print("2. Back")
        print("3. Exit")
        userChoice = int(input("Enter your Choice to Continue : "))
        print("--------------------------")

        # Handle user choices
        if userChoice == 1:
            home()
        elif userChoice == 2:
            admin()
        elif userChoice == 3:
            exiting()
        else:
            validOption()


    # Function to change the admin status
    def changeAdmin():
        print("--------------------------")
        print("Change Admin")
        print("--------------------------")
        # Get new admin's ID and password from the user
        newAdminId = int(input("Enter the New Admin's User ID : "))
        newAdminPassword = input("Enter the New Admin's Password : ")
        choice = input("Are you sure to change the Admin? (Yes/No) : ")
        print("--------------------------")

        # Check if the entered user ID exists
        c.execute("SELECT password FROM users WHERE userId=%s", (newAdminId,))
        result = c.fetchall()
        db.commit()

        # Check the user's choice to proceed or cancel
        if choice.lower() in ["yes", "y"]:
            # If the user ID is not valid, print an error message
            if len(result) == 0:
                print("Please enter a valid user id!")
            else:
                # If the entered password matches the user's password
                if newAdminPassword == result[0][0]:
                    # Update admin status for all users
                    c.execute(
                        "UPDATE users SET adminStatus='not admin' WHERE
    adminStatus ='admin'"
                    )
                    c.execute(
                        "UPDATE users SET adminStatus='admin' WHERE userId
    =%s",
                        (newAdminId,),
                    )
                    db.commit()

                    print("Admin Changed Successfully!")
                    print("--------------------------")
                    changeAdminMenu()

                else:
                    print("Please enter a valid password!")
        elif choice.lower() in ["no", "n"]:
            print("Admin Not Changed!")
            print("--------------------------")
            changeAdminMenu()
        else:
            validOption()
```

```
# Function to authenticate admin
def authAdmin():
    print("--------------------------")
    print("Admin Authentication")
    print("--------------------------")
    adminId = int(input("Enter the Admin's User ID : "))
    adminPassword = input("Enter the Admin's User Password : ")

    # Check if the entered admin ID exists
    c.execute("SELECT password FROM users WHERE userId=%s", (adminId,))
    result = c.fetchall()
    db.commit()

    # If the entered admin ID is not valid, print an error message
    if len(result) == 0:
        print("--------------------------")
        print("Please enter a valid user id!")
        print("--------------------------")
    else:
        # If the entered password matches the admin's password
        if adminPassword == result[0][0]:
            global USERID
            USERID = adminId
            print("\033[0;35m--------------------------\033[0;0m")
            print("\033[0;36mAdmin is verified successfully.\033[0;0m")
            print("\033[0;35m--------------------------\033[0;0m")
            admin()  # Call the admin menu
        else:
            print("Please enter a valid password!")
            print("--------------------------")


# Function to display the admin menu
def admin():
    print("--------------------------")
    print("Admin")
    print("--------------------------")
    print("1.  Login into User Panel")
    print("2.  Modify User")
    print("3.  Display Users")
    print("4.  Search Users")
    print("5.  Modify Book")
    print("6.  Issue Book")
    print("7.  Return Book")
    print("8.  Change Admin")
    print("9.  Home")
    print("10. Back")
    print("11. Exit")
    userChoice = int(input("Enter your Choice to Continue : "))
    print("--------------------------")

    # Handle user choices
    if userChoice == 1:
        print("You are successfully login into user panel.")
        print("--------------------------")

        user()
    elif userChoice == 2:
        modifyUser()
    elif userChoice == 3:
        displayUsers()
    elif userChoice == 4:
        searchUsers()
    elif userChoice == 5:
        modifyBook()
    elif userChoice == 6:
```

```
            issueBook()
        elif userChoice == 7:
            returnBook()
        elif userChoice == 8:
            changeAdmin()
        elif userChoice == 9:
            home()
        elif userChoice == 10:
            authAdmin()
        elif userChoice == 11:
            exiting()
        else:
            validOption()


    # Function to authenticate a user
    def authUser():
        print("---------------------------")
        print("User Authentication")
        print("---------------------------")
        userId = int(input("Enter the User ID : "))
        password = input("Enter the User Password : ")

        # Check if the entered user ID exists
        c.execute("SELECT password FROM users WHERE userId=%s", (userId,))
        result = c.fetchall()
        db.commit()

        # If the entered user ID is not valid, print an error message
        if len(result) == 0:
            print("---------------------------")
            print("Please enter a valid user id!")
            print("---------------------------")
        else:
            # If the entered password matches the user's password
            if password == result[0][0]:
                global USERID
                USERID = userId
                print("\033[0;35m---------------------------\033[0;0m")
                print("\033[0;36mUser is verified successfully.\033[0;0m")
                print("\033[0;35m---------------------------\033[0;0m")
                user()  # Call the user menu
            else:
                print("Please Enter a Valid Password!")
                print("---------------------------")


    # Function to search & display the wikipedia articles
    def wikipediaArticles():
        # Function to fetch article details
        def fetchingArticle(keyword, articleLength=1500):
            # Creating a Wikipedia API object
            wiki = wikipediaapi.Wikipedia(language="en", user_agent="digital-
    library/1.1")
            # Fetching the page for the given search query
            page = wiki.page(keyword)

            # Checking if the page exists
            if not page.exists():
                print(
                    f'Sorry, the Wikipedia Article for the keyword "{keyword}"
    does not exists.'
                )
                print("---------------------------")
            else:
                # Displaying article title
```

```
                print("Title : ")
                print(page.title)
                print("URL : ")
                print(page.fullurl)
                # Displaying a summary of the article within the specified
length
                print("Summary : ")

                start = 0
                end = 157
                article = page.summary[:articleLength]

                while end <= articleLength:
                    print(article[start:end])
                    start += 157
                    end += 157
                else:
                    print("--------------------------")

    print("--------------------------")
    print("Search Articles")
    print("--------------------------")
    # Taking user input for the keyword and article length
    keyword = input("Enter the Keyword for searching the Wikipedia Article
: ")
    articleLength = int(input("Enter the Article Length : "))
    print("--------------------------")

    # Calling the function to fetch and display the article
    fetchingArticle(keyword, articleLength)

    userMenu()


# Function to search & display the news
def news():
    def fetchNews(apiKey, country="in", category="science", numArticles=5):
        url = f"https://newsapi.org/v2/top-headlines"
        params = {
            "apiKey": apiKey,
            "country": country,
            "category": category,
            "pageSize": numArticles,
        }
        response = requests.get(url, params=params)

        if response.status_code == 200:
            news_data = response.json()
            articles = news_data.get("articles", [])

            for i, article in enumerate(articles, start=1):
                print(f"{i}. {article['title']}")
                print(f"   Source: {article['source']['name']}")
                print(f"   URL: {article['url']}")
                print("--------------------------")

        else:
            print(f"Error {response.status_code}: {response.text}")
            print("--------------------------")

    API_KEY = "YOUR_API_KEY"

    print("--------------------------")
    print("News")
    print("--------------------------")
    print("Country codes are : ")
```

```
        print("https://newsapi.org/sources")
        print("Categories are : ")
        print("business, entertainment, general, health, science, sports,
technology")
        print("--------------------------")
        country = input("Enter the Country Code : ")
        category = input("Enter the Category : ")
        numArticles = int(input("Enter the Number of Articles : "))
        print("--------------------------")

        fetchNews(API_KEY, country, category, numArticles)

        userMenu()


    # Function to display the issued books details of a user
    def issuedBooksDetails():
        print("--------------------------")
        print("Issued Books Details")
        print("--------------------------")
        returnPolicy()

        c.execute(
            "SELECT * FROM issuedBooksDetails WHERE userId=%s ORDER BY bookId",
(USERID,)
        )
        result = c.fetchall()
        db.commit()

        if result == []:
            print("No Books Issued!")
            print("--------------------------")
            userMenu()

        else:
            i = 0
            for row in result:
                i += 1
                r = length(i)
                print(f"{i}. Book ID : ", row[1])
                print(" " * r + "Book Name : ", row[2])
                print(" " * r + "Issue Date : ", row[3])
                print(" " * r + "Issue Time : ", row[4])
                print(" " * r + "Return Date : ", row[5])
                print(" " * r + "Return Time : ", row[6])
                print(" " * r + "Fine(in Rs.) : ", row[7])
                print("--------------------------")
            userMenu()


    # Function to display the user menu
    def user():
        print("--------------------------")
        print("User")
        print("--------------------------")
        # Check if the entered user ID exists
        c.execute('SELECT userId FROM users WHERE adminStatus="admin"')
        result = c.fetchall()
        db.commit()

        if result[0][0] == USERID:
            print("1.  Login into Admin Panel")
            print("2.  About the Library")
            print("3.  News")
            print("4.  Wikipedia Articles")
            print("5.  Display Books")
```

```
            print("6.  Search Books")
            print("7.  Issued Books Details")
            print("8.  Notes")
            print("9.  Home")
            print("10. Back")
            print("11. Exit")
            userChoice = int(input("Enter your Choice to Continue : "))
            print("--------------------------")

            # Handle user choices
            if userChoice == 1:
                print("You are successfully login into admin panel.")
                print("--------------------------")

                admin()
            elif userChoice == 2:
                aboutLibrary()
            elif userChoice == 3:
                news()
            elif userChoice == 4:
                wikipediaArticles()
            elif userChoice == 5:
                displayBooks()
            elif userChoice == 6:
                searchBooks()
            elif userChoice == 7:
                issuedBooksDetails()
            elif userChoice == 8:
                notes()
            elif userChoice == 9:
                home()
            elif userChoice == 10:
                authUser()
            elif userChoice == 11:
                exiting()
            else:
                validOption()
        else:
            print("1.  About Library")
            print("2.  News")
            print("3.  Wikipedia Articles")
            print("4.  Display Books")
            print("5.  Search Books")
            print("6.  Issued Books Details")
            print("7.  Notes")
            print("8.  Home")
            print("9.  Back")
            print("10. Exit")
            userChoice = int(input("Enter your Choice to Continue : "))
            print("--------------------------")

            # Handle user choices
            if userChoice == 1:
                aboutLibrary()
            elif userChoice == 2:
                news()
            elif userChoice == 3:
                wikipediaArticles()
            elif userChoice == 4:
                displayBooks()
            elif userChoice == 5:
                searchBooks()
            elif userChoice == 6:
                issuedBooksDetails()
            elif userChoice == 7:
                notes()
```

```
2179            elif userChoice == 8:
2180                home()
2181            elif userChoice == 9:
2182                authUser()
2183            elif userChoice == 10:
2184                exiting()
2185            else:
2186                validOption()
2187
2188
2189    # Function to display the main menu
2190    def home():
2191        while True:
2192            print("============================")
2193            print("\033[1;32m~~~~~~~~~~~~~~~~~~~~~~~~~~\033[0;0m")
2194            print(
2195                "\033[1;31m"
2196                + pyfiglet.figlet_format("Welcome to the", font="banner3",
2197    width=1000)
2198            )
2199            print(
2200                pyfiglet.figlet_format("Digital Library", font="banner3",
2201    width=1000)
2202                + "\033[0;0m"
2203            )
2204            print("\033[1;32m~~~~~~~~~~~~~~~~~~~~~~~~~~\033[0;0m")
2205            print("============================")
2206            print("----------------------------")
2207            print("Home")
2208            print("----------------------------")
2209            print("1. Admin")
2210            print("2. User")
2211            print("3. Exit")
2212            userChoice = int(input("Enter your Choice to Continue : "))
2213            print("----------------------------")
2214
2215            # Handle user choices
2216            if userChoice == 1:
2217                authAdmin()
2218            elif userChoice == 2:
2219                authUser()
2220            elif userChoice == 3:
2221                exiting()
2222            else:
2223                validOption()
2224
2225
2226    # Call the main menu function
2227    home()
2228
2229
```

MySQL Database

Library Database:

```
Enter password: ************
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use library;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_library  |
+--------------------+
| books              |
| issuedbooksdetails |
| notes              |
| users              |
+--------------------+
4 rows in set (0.07 sec)
```

Books Table:

```
mysql> desc books;
+-----------------+-------------+------+-----+------------+-------+
| Field           | Type        | Null | Key | Default    | Extra |
+-----------------+-------------+------+-----+------------+-------+
| bookId          | int         | NO   | PRI | NULL       |       |
| bookName        | varchar(50) | NO   |     | NULL       |       |
| publicationYear | int         | YES  |     | NULL       |       |
| issueDate       | date        | YES  |     | NULL       |       |
| issueTime       | time        | YES  |     | NULL       |       |
| returnDate      | date        | YES  |     | NULL       |       |
| returnTime      | time        | YES  |     | NULL       |       |
| author          | varchar(40) | YES  |     | NULL       |       |
| issueStatus     | varchar(10) | NO   |     | not issued |       |
| issuedUserId    | int         | YES  | MUL | NULL       |       |
+-----------------+-------------+------+-----+------------+-------+
10 rows in set (0.02 sec)

mysql> select * from books;
+--------+-------------+-----------------+------------+-----------+------------+------------+--------------+-------------+--------------+
| bookId | bookName    | publicationYear | issueDate  | issueTime | returnDate | returnTime | author       | issueStatus | issuedUserId |
+--------+-------------+-----------------+------------+-----------+------------+------------+--------------+-------------+--------------+
|   3000 | English     |            2005 | NULL       | NULL      | NULL       | NULL       | Sman         | not issued  |         NULL |
|   4000 | Hindi       |            2011 | NULL       | NULL      | NULL       | NULL       | Rman         | not issued  |         NULL |
|   5000 | Histroy     |            2010 | NULL       | NULL      | NULL       | NULL       | NULL         | not issued  |         NULL |
|   5263 | physics     |            2003 | 2023-12-14 | 14:31:21  | NULL       | NULL       | H.C. Verma   | issued      |         1025 |
|   5658 | cs          |            2016 | NULL       | NULL      | NULL       | NULL       | Sunita Arora | not issued  |         NULL |
|  12305 | mathematics |            2011 | NULL       | NULL      | NULL       | NULL       | V.V. Acharya | not issued  |         NULL |
+--------+-------------+-----------------+------------+-----------+------------+------------+--------------+-------------+--------------+
6 rows in set (0.01 sec)
```

Users Table:

```
mysql> desc users;
+-------------+-------------+------+-----+-----------+-------+
| Field       | Type        | Null | Key | Default   | Extra |
+-------------+-------------+------+-----+-----------+-------+
| userId      | int         | NO   | PRI | NULL      |       |
| userName    | varchar(50) | NO   |     | NULL      |       |
| phoneNumber | varchar(13) | YES  |     | NULL      |       |
| emailId     | varchar(40) | NO   |     | NULL      |       |
| password    | varchar(40) | NO   |     | NULL      |       |
| adminStatus | varchar(9)  | NO   |     | not admin |       |
+-------------+-------------+------+-----+-----------+-------+
6 rows in set (0.00 sec)

mysql> select * from users;
+--------+----------+-------------+--------------------+-----------+-------------+
| userId | userName | phoneNumber | emailId            | password  | adminStatus |
+--------+----------+-------------+--------------------+-----------+-------------+
|   1023 | Aman     | 9564823675  | abc@gmail.com      | Aman@1023 | not admin   |
|   1024 | Cman     | 8564526545  | cman1024@gmail.com | Cman@1024 | not admin   |
|   1025 | Bman     | 8623254587  | def@gmail.com      | Bman@1025 | admin       |
|   1026 | Dman     | 9456875462  | dman1026@gmail.com | Dman@1026 | not admin   |
|   1028 | Eman     | 8564231547  | eman1028@gmail.com | Eman@1028 | not admin   |
+--------+----------+-------------+--------------------+-----------+-------------+
5 rows in set (0.00 sec)
```

Notes Table:

```
mysql> desc notes;
+-----------------+----------------+------+-----+---------+-------+
| Field           | Type           | Null | Key | Default | Extra |
+-----------------+----------------+------+-----+---------+-------+
| userId          | int            | NO   | MUL | NULL    |       |
| noteNumber      | int            | NO   |     | NULL    |       |
| noteTitle       | varchar(50)    | YES  |     | NULL    |       |
| noteDescription | varchar(10000) | YES  |     | NULL    |       |
| updateDate      | date           | NO   |     | NULL    |       |
| updateTime      | time           | NO   |     | NULL    |       |
+-----------------+----------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> select * from notes;
+--------+------------+----------------------------------------+--------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------+------------+------------+
| userId | noteNumber | noteTitle                              | noteDescription

                                                                                                               | updateDate | updateTime |
+--------+------------+----------------------------------------+--------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------+------------+------------+
|   1025 |          1 | ISRO                                   | The Indian Space Research Organisation (ISRO) is the national space agency of India. It oper
ates as the primary research and development arm of the Department of Space (DoS), which is directly overseen by the Prime Minister of India, while the Chai
rman of ISRO also acts as the executive of DoS.                                                                | 2023-12-14 | 18:03:18   |
|   1023 |          1 | Massachusetts Institute of Technology | The Massachusetts Institute of Technology (MIT) is a private land-grant research university
in Cambridge, Massachusetts. Established in 1861, MIT has played a significant role in the development of many areas of modern technology and science. Its r
eputation for innovation and rankings have made it one of the most prestigious universities in the world. | 2023-12-14 | 17:35:40   |
+--------+------------+----------------------------------------+--------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------+------------+------------+
2 rows in set (0.01 sec)
```

Issued Books Details Table:

```
mysql> desc issuedbooksdetails;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| userId     | int         | NO   | MUL | NULL    |       |
| bookId     | int         | NO   | MUL | NULL    |       |
| bookName   | varchar(50) | NO   |     | NULL    |       |
| issueDate  | date        | YES  |     | NULL    |       |
| issueTime  | time        | YES  |     | NULL    |       |
| returnDate | date        | YES  |     | NULL    |       |
| returnTime | time        | YES  |     | NULL    |       |
| fineInRs   | int         | NO   |     | 0       |       |
+------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

mysql> select * from issuedbooksdetails;
+--------+--------+-------------+------------+-----------+------------+------------+----------+
| userId | bookId | bookName    | issueDate  | issueTime | returnDate | returnTime | fineInRs |
+--------+--------+-------------+------------+-----------+------------+------------+----------+
|   1025 |   5263 | physics     | 2023-12-14 | 14:31:21  | NULL       | NULL       |        0 |
|   1023 |   5658 | cs          | 2023-12-14 | 14:31:53  | 2023-12-14 | 14:32:04   |        0 |
|   1023 |  12305 | mathematics | 2023-12-14 | 14:32:28  | 2023-12-14 | 14:32:34   |        0 |
+--------+--------+-------------+------------+-----------+------------+------------+----------+
3 rows in set (0.01 sec)
```

Outputs

Starting of the program:

Admin Authentication:

Adding a new user:

```
--------------------------
--------------------------
User added successfully!
--------------------------
1. Home
2. Back
3. Exit
Enter your Choice to Continue : 3
--------------------------
```

Updating user details:

```
Enter your Choice to Continue : 3
--------------------------
--------------------------
Update User Details
--------------------------
1. Update the User ID
2. Update the User Name
3. Update the User Phone Number
4. Update the User Email ID
5. Update the User Password
6. Home
7. Back
8. Exit
Enter your Choice to Continue : 7
--------------------------
--------------------------
Modify User
--------------------------
1. Add User
2. Delete User
3. Update User Details
4. Home
5. Back
6. Exit
Enter your Choice to Continue : 5
--------------------------
--------------------------
Admin
--------------------------
1.  Login into User Panel
2.  Modify User
3.  Display Users
4.  Search Users
5.  Modify Book
6.  Issue Book
7.  Return Book
8.  Change Admin
9.  Home
10. Back
11. Exit
Enter your Choice to Continue : 5
--------------------------
--------------------------
Modify Book
--------------------------
1. Add Book
2. Delete Book
3. Update Book Details
4. Home
5. Back
6. Exit
Enter your Choice to Continue : 1
--------------------------
```

Adding a new book:

```
--------------------------
Add Book
--------------------------
Enter the Book ID : 6000
Enter the Book Name : Physical Education
Enter the Book Publication Year : 2016
Enter the Book Author Name : Tman
--------------------------
Book added Successfully!
--------------------------
1. Home
```

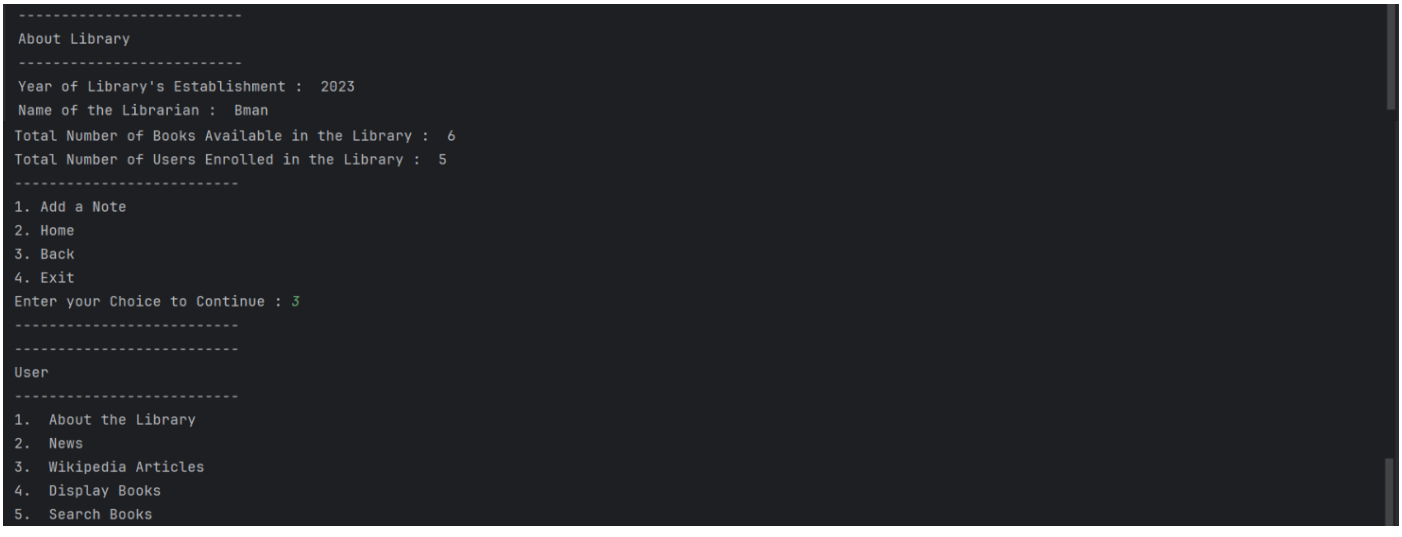
```
2. Back
3. Exit
Enter your Choice to Continue : 3
-------------------------
```

Updating book details:

```
-------------------------
Update Book Details
-------------------------
1. Update the Book ID
2. Update the Book Name
3. Update the Book Publication Year
4. Update the Book Author Name
5. Home
6. Back
7. Exit
Enter your Choice to Continue : 6
-------------------------
-------------------------
Modify Book
-------------------------
1. Add Book
2. Delete Book
3. Update Book Details
4. Home
5. Back
6. Exit
Enter your Choice to Continue : 5
-------------------------
-------------------------
Admin
-------------------------
1.  Login into User Panel
2.  Modify User
3.  Display Users
4.  Search Users
5.  Modify Book
6.  Issue Book
7.  Return Book
8.  Change Admin
9.  Home
10. Back
11. Exit
Enter your Choice to Continue : 1
-------------------------
You are successfully login into user panel.
-------------------------
-------------------------
User
-------------------------
1.  Login into Admin Panel
2.  About the Library
3.  News
4.  Wikipedia Articles
5.  Display Books
6.  Search Books
7.  Issued Books Details
8.  Notes
9.  Home
10. Back
11. Exit
Enter your Choice to Continue : 1
-------------------------
You are successfully login into admin panel.
-------------------------
-------------------------
Admin
-------------------------
1.  Login into User Panel
2.  Modify User
3.  Display Users
4.  Search Users
5.  Modify Book
6.  Issue Book
7.  Return Book
8.  Change Admin
9.  Home
10. Back
11. Exit
Enter your Choice to Continue : 9
-------------------------
=========================
```

User Authentication:

About Library:



```
6.  Issued Books Details
7.  Notes
Enter your Choice to Continue : 2
-------------------------
```

News:

```
-------------------------
News
-------------------------
Country codes are :
https://newsapi.org/sources
Categories are :
business, entertainment, general, health, science, sports, technology
-------------------------
Enter the Country Code : in
Enter the Category : science
Enter the Number of Articles : 5
-------------------------
1. DART Asteroid Impact Aftermath Caught On Webb and Hubble Space Telescopes - MSN
   Source: msnNOW
   URL: https://www.msn.com/en-us/news/technology/dart-asteroid-impact-aftermath-caught-on-webb-and-hubble-space-telescopes/vi-AA1lHnyH
-------------------------
2. Watch: NASA Sends Cat Video To Earth From Spaceship 31 Million Km Away - NDTV
   Source: NDTV News
   URL: https://www.ndtv.com/world-news/nasa-sends-cat-video-to-earth-from-spaceship-31-million-km-away-4701041
-------------------------
3. Scientists now know what happened to a chunk of Earth's crust missing for millions of years - WION
   Source: WION
   URL: https://www.wionews.com/science/scientists-reveal-how-oceans-ate-up-large-part-of-earths-crust-671110
-------------------------
4. Here's how astronauts workout in space and why is it important - IndiaTimes
   Source: The Times of India
   URL: https://timesofindia.indiatimes.com/etimes/trending/heres-how-astronauts-workout-in-space-and-why-is-it-important/articleshow/106099552.cms
-------------------------
5. Year-Ender 2023: A Look at the Top 11 Space Moments of 2023 | Mint - Mint
   Source: Livemint
   URL: https://www.livemint.com/science/news/yearender-2023-a-look-at-the-top-11-space-moments-of-2023-11702710071902.html
-------------------------
1. Add Note
2. Home
3. Back
4. Exit
Enter your Choice to Continue : 3
-------------------------
-------------------------
User
-------------------------
1.  About the Library
2.  News
3.  Wikipedia Articles
4.  Display Books
5.  Search Books
6.  Issued Books Details
7.  Notes
8.  Home
9.  Back
10. Exit
Enter your Choice to Continue : 3
-------------------------
```

Wikipedia Articles:

```
-------------------------
Wikipedia Articles
-------------------------
Enter the Keyword for searching the Wikipedia Article : nasa
Enter the Article Length : 2000
-------------------------
Title :
NASA
URL :
https://en.wikipedia.org/wiki/NASA
```

```
Summary :
The National Aeronautics and Space Administration (NASA ) is an independent agency of the U.S. federal government responsible for the civil space program, ae
ronautics research, and space research. Established in 1958, NASA succeeded the National Advisory Committee for Aeronautics (NACA) to give the U.S. space dev
elopment effort a distinctly civilian orientation, emphasizing peaceful applications in space science. NASA has since led most American space exploration, in
cluding Project Mercury, Project Gemini, the 1968–1972 Apollo Moon landing missions, the Skylab space station, and the Space Shuttle. NASA currently supports
 the International Space Station and oversees the development of the Orion spacecraft and the Space Launch System for the crewed lunar Artemis program, the C
ommercial Crew spacecraft, and the planned Lunar Gateway space station.
NASA's science is focused on better understanding Earth through the Earth Observing S
ystem; advancing heliophysics through the efforts of the Science Mission Directorate's Heliophysics Research Program; exploring bodies throughout the Solar S
ystem with advanced robotic spacecraft such as New Horizons and planetary rovers such as Perseverance; and researching astrophysics topics, such as the Big B
ang, through the James Webb Space Telescope, the Great Observatories and associated programs. NASA's Launch Services Program provides oversight of launch ope
rations and countdown management for its uncrewed launches.


-------------------------
1. Add Note
2. Home
3. Back
```

```
Enter your Choice to Continue : 3
-------------------------
-------------------------
User
-------------------------
1.  About the Library
2.  News
3.  Wikipedia Articles
4.  Display Books
5.  Search Books
6.  Issued Books Details
7.  Notes
8.  Home
9.  Back
10. Exit
```

```
Enter your Choice to Continue : 6
-------------------------
```

Issued Books Details:

```
-------------------------
Issued Books Details
-------------------------
Return Policy :
The issued book should be returned within 14 days(2 weeks).
If the user kept the issued book for more than 14 days, then the user have to pay ₹5 as fine for each extra day the user kept the issued book.
-------------------------
1. Book ID :  5658
   Book Name :  cs
   Issue Date :  2023-12-14
   Issue Time :  14:31:53
   Return Date :  2023-12-14
   Return Time :  14:32:04
   Fine(in Rs.) :  0
-------------------------
1. Add Note
2. Home
3. Back
4. Exit
Enter your Choice to Continue : 3
-------------------------
-------------------------
User
-------------------------
1.  About the Library
2.  News
3.  Wikipedia Articles
4.  Display Books
5.  Search Books
6.  Issued Books Details
7.  Notes
8.  Home
9.  Back
10. Exit
Enter your Choice to Continue : 7
-------------------------
-------------------------
Notes
-------------------------
1. Modify Note
2. Display Notes
3. Search Notes
4. Home
```

```
5. Back
6. Exit
Enter your Choice to Continue : 5
-------------------------
-------------------------
User
-------------------------
1.  About the Library
2.  News
3.  Wikipedia Articles
4.  Display Books
5.  Search Books
6.  Issued Books Details
7.  Notes
8.  Home
9.  Back
10. Exit
Enter your Choice to Continue : 10
-------------------------
```

Existing the program:

```
-------------------------
Exiting the program.
Thank You!
-------------------------

Process finished with exit code 0
```

REFERENCES

1. News API
   https://newsapi.org/
2. Wikipedia
   https://www.wikipedia.org/
3. Python
   https://www.python.org/
4. MySQL
   https://www.mysql.com/
5. ANSI Escape Codes in Python
   - https://pypi.org/project/ansi/
   - https://replit.com/talk/learn/ANSI-Escape-Codes-in-Python/22803
   - https://gist.github.com/rene-d/9e584a7dd2935d0f461904b9f2950007
6. Class 11th & 12th Computer Science Arihant Books