

**PM SHRI KENDRIYA VIDYALAYA
BERHAMPUR**



तत् त्वं पूषन् अपावृणु
केन्द्रीय विद्यालय संगठन

COMPUTER SCIENCE PROJECT

2023-24

**PROJECT TOPIC:
LIBRARY MANAGEMENT SYSTEM**

Submitted by:

Name: BIBHASINDHU PRADHAN

Class: XII-A

CBSE Roll Number:

Under the Guidance of:

SAROJ KANTA MISRA, PGT (CS)

CERTIFICATE

This is to certify that BIBHASINDHU PRADHAN of class: XII - A of KENDRIYA VIDYALAYA BERHAMPUR has done his project on LIBRARY MANAGEMENT SYSTEM (DIGITAL LIBRARY) 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, PGT (CS) 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

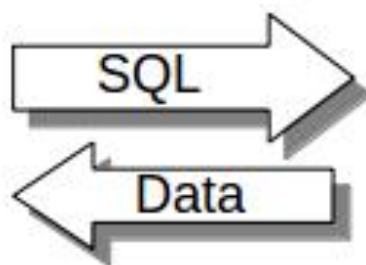
1. Desktop Computer / Laptop
2. Mobile Phone

SOFTWARES

1. Python (Latest Version)
2. MySQL
3. MySQL-Connector-Python, Requests, Wikipedia-API, Datetime, Pyfiglet Modules



Python



Database System

TABLE OF CONTENTS

<u>S.No.</u>	<u>Topic</u>	<u>Page No.</u>
1	Certificate	1
2	Acknowledgement	2
3	Hardware and Softwares Required	3
4	Introduction	5
5	Python Source Code	10
6	MySQL Database	45
7	Outputs	48
8	References	56

INTRODUCTION

The project LIBRARY MANAGEMENT SYSTEM (DIGITAL LIBRARY) includes enrolment of users, adding of books into the library system. The software has the facility to search for news, Wikipedia articles. It includes an 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.

LIBRARY

A library is a collection of books, and possibly other materials and media, that is accessible for use by its members and members of allied institutions. Libraries provide physical or digital materials, and may be a physical location, a virtual space, or both. A library's collection normally includes printed materials which may be borrowed, and usually also includes a reference section of publications which may only be utilized inside the premises.

Libraries can vary widely in size and may be organised and maintained by a public body such as a government, an institution (such as a school or museum), a corporation, or a private individual. In addition to providing materials, libraries also provide the services of librarians who are trained experts in finding, selecting, circulating and organising information while interpreting information needs and navigating and analysing large amounts of information with a variety of resources.

LIBRARY MANAGEMENT SYSTEM **(DIGITAL LIBRARY)**

Library management system (LMS), is an enterprise resource planning system for a library, used to track enrolled users, available books, books issued and to whom, books returned and it's fines, etc.

The purpose of a library management system is to operate a library with efficiency and at reduced costs. The system being entirely automated streamlines all the tasks involved in operations of the library.

The library management system software helps in reducing operational costs. Managing a library manually is labour intensive and an immense amount of paperwork is involved.

The system saves time for both the user and the librarian.

With just a click the user can search for the books available in the library. The librarian can answer queries with ease regarding the availability of books. Adding, removing or editing the database is a simple process. Adding new users or cancelling existing userships can be done with ease.

The automated system saves a considerable amount of time as opposed to the manual system.

FUNCTIONS LIST

❖ Admin

- Login into User Panel
- Modify User
 - Add User
 - Delete User
 - Update User
- Display Users
- Search Users
- Modify Book
 - Add Book
 - Delete Book
 - Update Book
- Issue Book
- Return Book
- Change Admin
- Home
- Back
- Exit

❖ User

- About Library

- **News**
- **Wikipedia Articles**
- **Display Books**
- **Search Books**
- **Issued Books Details**
 - **Notes**
 - **Modify Notes**
 - **Add Notes**
 - **Delete Notes**
 - **Update Notes**
 - **Display Notes**
 - **Search Notes**
 - **Home**
 - **Back**
 - **Exit**

Python Source ***Code***

```

1  # Importing necessary libraries
2  import mysql.connector
3  import pyfiglet
4  import requests
5  import wikipediaapi
6  from datetime import datetime
7
8
9  # Connect to the MySQL database
10 db = mysql.connector.connect(
11     host="localhost",
12     user="root",
13     password="admin",
14     database="library",
15 )
16 c = db.cursor()
17
18
19 # Function to display the return policy information
20 def returnPolicy():
21     print("Return Policy : ")
22     print("The issued book should be returned within 14 days(2 weeks).")
23     print(
24         "If the user kept the issued book for more than 14 days, then the
25         user have to pay ₹5 as fine for each extra day the user kept the issued
26         book."
27     )
28     print("-----")
29
30
31 # Function to calculate the length of a given integer after converting it
32 # to a string
33 def length(i):
34     s = str(i)
35     length = len(s) + 2
36
37     return length
38
39
40 # Function to display a message for an invalid option
41 def validOption():
42     print("Please enter a valid option!")
43     print("-----")
44
45
46 # Function to handle program exit
47 def exiting():
48     print("\033[3;34m-----\033[0;0m")
49     print("\033[3;33mExiting the program.")
50     print("Thank You!\033[0;0m")
51     print("\033[3;34m-----\033[0;0m")
52     exit()
53
54
55 # Function to display the user menu and handle user choices
56 def userMenu():
57     # Displaying options for the user
58     print("1. Add Note")
59     print("2. Home")
60     print("3. Back")
61     print("4. Exit")
62     # Taking user choice as input
63     userChoice = int(input("Enter your Choice to Continue : "))
64     print("-----")
65
66     # Handle user choices

```

```

67     if userChoice == 1:
68         addNote()
69     elif userChoice == 2:
70         home()
71     elif userChoice == 3:
72         user()
73     elif userChoice == 4:
74         exiting()
75     else:
76         validOption()
77
78
79 # Function to display information about the library
80 def aboutLibrary():
81     # Retrieve the name of the librarian who is also an admin
82     c.execute("SELECT userName FROM users WHERE adminStatus='admin'")
83     userName = c.fetchall()
84
85     # Retrieve the total number of books and users in the library
86     c.execute("SELECT * FROM books")
87     totalBooks = c.fetchall()
88
89     c.execute("SELECT * FROM users")
90     totalUsers = c.fetchall()
91     db.commit()
92
93     print("-----")
94     print("About Library")
95     print("-----")
96     # Display library information
97     print("Year of Library's Establishment : ", 2023)
98     print("Name of the Librarian : ", userName[0][0])
99     print("Total Number of Books Available in the Library : ",
100 len(totalBooks))
101     print("Total Number of Users Enrolled in the Library : ",
102 len(totalUsers))
103     print("-----")
104     userMenu()
105
106
107 # Function to display the list of books in the library
108 def displayBooks():
109     print("-----")
110     print("Display Books")
111     print("-----")
112     # Retrieve all books from the database
113     c.execute("SELECT * FROM books ORDER BY bookId")
114     result = c.fetchall()
115     db.commit()
116
117     # Display books if available, otherwise notify the user
118     if result:
119         print("Books available in the Digital Library are :")
120         print("-----")
121         i = 0
122         for row in result:
123             i += 1
124             r = length(i)
125             print(f"{i}. Book ID : {row[0]}")
126             print(" " * r + f"Book Name : {row[1]}")
127             print(" " * r + f"Publication Year : {row[2]}")
128             print(" " * r + f"Author Name : {row[7]}")
129             print(" " * r + f"Issue Status : {row[8]}")
130             print("-----")
131         userMenu()
132     else:

```

```

133         # Notify the user if no books are found
134         print("No books found.")
135         print("-----")
136         userMenu()
137
138
139     # Search books menu options
140     def searchBooksMenu():
141         print("1. Add Note")
142         print("2. Home")
143         print("3. Back")
144         print("4. Exit")
145         userChoice = int(input("Enter your Choice to Continue : "))
146
147         # User choices handling
148         if userChoice == 1:
149             addNote()
150         elif userChoice == 2:
151             home()
152         elif userChoice == 3:
153             searchBooks()
154         elif userChoice == 4:
155             exiting()
156         else:
157             validOption()
158
159
160     # Function to search books by Book ID
161     def searchBooksbyId():
162         print("-----")
163         print("Search Books by Book ID")
164         print("-----")
165         # Get user input for Book ID
166         bookId = int(input("Enter the Book ID to search the Book : "))
167         print("-----")
168
169         # Execute SQL query to retrieve book information by Book ID
170         c.execute("SELECT * FROM books WHERE bookId=%s", (bookId,))
171         result = c.fetchall()
172         db.commit()
173
174         # Display search results if books are found, otherwise notify the user
175         if result:
176             print(f'Book available in the Digital Library with the Book ID
177 "{bookId}" is :')
178             print("-----")
179             i = 0
180             for row in result:
181                 i += 1
182                 r = length(i)
183                 print(f"{i}. Book ID : {row[0]}")
184                 print(" " * r + f"Book Name : {row[1]}")
185                 print(" " * r + f"Publication Year : {row[2]}")
186                 print(" " * r + f"Author Name : {row[7]}")
187                 print(" " * r + f"Issue Status : {row[8]}")
188                 print("-----")
189                 searchBooksMenu()
190         else:
191             print(f'No book found with the book id "{bookId}.'.')
192             print("-----")
193             searchBooksMenu()
194
195
196     # Function to search books by keyword
197     def searchBooksbyKeyword():
198         print("-----")

```

```

199     print("Search Books by Keyword")
200     print("-----")
201     # Get user input for keyword
202     keyword = input("Enter a Keyword to search Books : ")
203     print("-----")
204
205     # Execute SQL query to retrieve books by keyword
206     c.execute(
207         "SELECT * FROM books WHERE bookName LIKE '%{}%' ORDER BY
208 bookId".format(keyword)
209     )
210     result = c.fetchall()
211     db.commit()
212
213     # Display search results if books are found, otherwise notify the user
214     if result:
215         print(
216             f'Books available in the Digital Library with the Keyword
217 "{keyword}" are :'
218         )
219         print("-----")
220         i = 0
221         for row in result:
222             i += 1
223             r = length(i)
224             print(f"{i}. Book ID : {row[0]}")
225             print(" " * r + f"Book Name : {row[1]}")
226             print(" " * r + f"Publication Year : {row[2]}")
227             print(" " * r + f"Author Name : {row[7]}")
228             print(" " * r + f"Issue Status : {row[8]}")
229             print("-----")
230         searchBooksMenu()
231     else:
232         print(f'No books found with the keyword "{keyword}.'.)
233         print("-----")
234         searchBooksMenu()
235
236
237 # Function to display search options for books
238 def searchBooks():
239     print("-----")
240     print("Search Books")
241     print("-----")
242     print("1. Search by Book ID")
243     print("2. Search by Keyword")
244     print("3. Home")
245     print("4. Back")
246     print("5. Exit")
247     userChoice = int(input("Enter your Choice to Continue : "))
248     print("-----")
249
250     # User choices handling
251     if userChoice == 1:
252         searchBooksbyId()
253     elif userChoice == 2:
254         searchBooksbyKeyword()
255     elif userChoice == 3:
256         home()
257     elif userChoice == 4:
258         user()
259     elif userChoice == 5:
260         exiting()
261     else:
262         validOption()
263
264

```

```

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:

```

```

397         exiting()
398     else:
399         validOption()
400
401
402 def notBook(bookId):
403     print(f'The book of book id "{bookId}" does not available in the
404 digital library.')
405     print("-----")
406     updateBookMenu()
407
408
409 # Function to update book details
410 def updateBook():
411     print("-----")
412     print("Update Book Details")
413     print("-----")
414     print("1. Update the Book ID")
415     print("2. Update the Book Name")
416     print("3. Update the Book Publication Year")
417     print("4. Update the Book Author Name")
418     print("5. Home")
419     print("6. Back")
420     print("7. Exit")
421     userChoice = int(input("Enter your Choice to Continue : "))
422     print("-----")
423
424     c.execute("SELECT bookId FROM books")
425     result = c.fetchall()
426     db.commit()
427
428     # User choices handling
429     if userChoice == 1:
430         currentBookId = int(input("Enter the Current Book ID : "))
431         newBookId = int(input("Enter the New Book ID : "))
432
433         if (currentBookId,) in result:
434             # Execute SQL query to update the Book ID
435             c.execute(
436                 "UPDATE books SET bookId=%s WHERE bookId=%s", (newBookId,
437 currentBookId)
438             )
439             db.commit()
440
441             print("Book ID changed Successfully!")
442             print("-----")
443             updateBookMenu()
444         else:
445             notBook(currentBookId)
446
447     elif userChoice == 2:
448         bookId = int(input("Enter the Book ID : "))
449         newBookName = input("Enter the New Book Name : ")
450
451         if (bookId,) in result:
452             # Execute SQL query to update the Book Name
453             c.execute(
454                 "UPDATE books SET bookName=%s WHERE bookId=%s",
455 (newBookName, bookId)
456             )
457             db.commit()
458
459             print("Book Name changed Successfully!")
460             print("-----")
461             updateBookMenu()
462         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

```

```

529 def issueBook():
530     print("-----")
531     print("Issue Book")
532     print("-----")
533     bookId = int(input("Enter the Book ID to be Issued: "))
534     userId = int(input("Enter the User ID to whom Book will be Issued: "))
535
536     # Execute SQL query to check the issue status of the book
537     c.execute("SELECT userId FROM users")
538     result1 = c.fetchall()
539     c.execute("SELECT bookId FROM books")
540     result2 = c.fetchall()
541     c.execute("SELECT issueStatus FROM books WHERE bookId=%s", (bookId,))
542     result3 = c.fetchall()
543     db.commit()
544
545     if (userId,) in result1:
546         if (bookId,) in result2:
547             # Check if the book is not already issued
548             if result3[0][0] == "not issued":
549                 # Execute SQL queries to update book details and mark it as
550 issued
551                 c.execute(
552                     "UPDATE books SET issueDate = CURRENT_DATE WHERE bookId
553 = %s",
554                     (bookId,),
555                 )
556                 c.execute(
557                     "UPDATE books SET issueTime = CURRENT_TIME WHERE bookId
558 = %s",
559                     (bookId,),
560                 )
561                 c.execute(
562                     "UPDATE books SET issueStatus = 'issued' WHERE bookId =
563 %s",
564                     (bookId,),
565                 )
566                 c.execute(
567                     "UPDATE books SET returnDate = NULL WHERE bookId = %s",
568 (bookId,)
569                 )
570                 c.execute(
571                     "UPDATE books SET returnTime = NULL WHERE bookId = %s",
572 (bookId,)
573                 )
574                 c.execute(
575                     "UPDATE books SET issuedUserId = %s WHERE bookId = %s",
576                     (userId, bookId),
577                 )
578                 db.commit()
579                 c.execute(
580                     "select issuedUserId,bookName,issueDate,issueTime from
581 books where bookId=%s",
582                     (bookId,),
583                 )
584                 result = c.fetchall()
585                 c.execute(
586                     "INSERT INTO issuedBooksDetails (userId,
587 bookId,bookName,issueDate,issueTime) VALUES (%s, %s, %s, %s, %s)",
588                     (result[0][0], bookId, result[0][1], result[0][2],
589 result[0][3]),
590                 )
591                 db.commit()
592
593                 print("-----")
594                 print(

```

```

595         f'Book of Book Id "{bookId}" is issued successfully to
596 the User of User Id "{userId}".'
597     )
598     print("-----")
599     returnPolicy()
600     issueBookMenu()
601     else:
602         # Notify the user that the book is already issued
603         print(
604             f'The book of book id "{bookId}" is already issued by
605 another user.'
606         )
607         print("-----")
608         issueBookMenu()
609     else:
610         print(
611             f"Book with book id {bookId} does not available in the
612 digital library."
613         )
614         print("-----")
615         issueBookMenu()
616     else:
617         print(f"User with user id {userId} does not exists in the digital
618 library.")
619         print("-----")
620         issueBookMenu()
621
622
623 # Function to display the return book menu and handle user choices
624 def returnBookMenu():
625     print("1. Home")
626     print("2. Back")
627     print("3. Exit")
628     userChoice = int(input("Enter your Choice to Continue : "))
629     print("-----")
630
631     # User choices handling
632     if userChoice == 1:
633         home()
634     elif userChoice == 2:
635         admin()
636     elif userChoice == 3:
637         exiting()
638     else:
639         validOption()
640
641
642 # Function to return a book
643 def returnBook():
644     print("-----")
645     print("Return Book")
646     print("-----")
647     bookId = int(input("Enter the Book ID to be Returned: "))
648
649     # Execute SQL query to check the issue status of the book
650     c.execute("SELECT bookId FROM books")
651     result1 = c.fetchall()
652     c.execute("SELECT issueStatus FROM books WHERE bookId=%s", (bookId,))
653     result2 = c.fetchall()
654
655     db.commit()
656
657     if (bookId,) in result1:
658         # Check if the book is issued
659         if result2[0][0] == "issued":
660             # Execute SQL queries to update book details and mark it as

```

```

661     returned
662         c.execute(
663             "UPDATE books SET returnDate = CURRENT_DATE WHERE bookId =
664 %s",
665             (bookId,),
666         )
667         c.execute(
668             "UPDATE books SET returnTime = CURRENT_TIME WHERE bookId =
669 %s",
670             (bookId,),
671         )
672         c.execute(
673             "UPDATE books SET issueStatus = 'not issued' WHERE bookId =
674 %s",
675             (bookId,),
676         )
677         db.commit()
678         c.execute(
679             "select issuedUserId,returnDate,returnTime from books where
680 bookId=%s",
681             (bookId,),
682         )
683         result = c.fetchall()
684         c.execute(
685             "UPDATE issuedBooksDetails SET returnDate = %s, returnTime
686 = %s WHERE userId = %s AND bookId = %s",
687             (result[0][1], result[0][2], result[0][0], bookId),
688         )
689
690         db.commit()
691         c.execute(
692             "UPDATE books SET issuedUserId = NULL WHERE bookId = %s",
693 (bookId,)
694         )
695         db.commit()
696
697         print(f'The book of book id "{bookId}" is returned
698 successfully.')
699
700         c.execute("select issueDate from books WHERE bookId = %s",
701 (bookId,))
702         issueDate = c.fetchall()
703         c.execute("select returnDate from books WHERE bookId = %s",
704 (bookId,))
705         returnDate = c.fetchall()
706         db.commit()
707
708         c.execute("UPDATE books SET issueDate = NULL WHERE bookId =
709 %s", (bookId,))
710         c.execute("UPDATE books SET issueTime = NULL WHERE bookId =
711 %s", (bookId,))
712         c.execute("UPDATE books SET returnDate = NULL WHERE bookId =
713 %s", (bookId,))
714         c.execute("UPDATE books SET returnTime = NULL WHERE bookId =
715 %s", (bookId,))
716         db.commit()
717
718         d1 = datetime.strptime(f"{issueDate[0][0]}", "%Y-%m-%d")
719         d2 = datetime.strptime(f"{returnDate[0][0]}", "%Y-%m-%d")
720         dateDifference = d1 - d2
721
722         if dateDifference.days > 14:
723             extraDays = dateDifference.days - 14
724             fine = extraDays * 5
725             print("Fine(in Rs.) : ", fine)
726         c.execute(

```

```

727         "update issuedBooksDetails set fineInRs=%s where
728     userId=%s and bookId=%s",
729         (fine, result[0][0], bookId),
730     )
731     db.commit()
732     else:
733         fine = 0 * 5
734         print("Fine(in Rs.) : ", fine)
735         c.execute(
736             "update issuedBooksDetails set fineInRs=%s where
737     userId=%s and bookId=%s",
738             (fine, result[0][0], bookId),
739         )
740         db.commit()
741
742         print("-----")
743         returnBookMenu()
744     else:
745         # Notify the user that the book is not issued
746         print(f'The book of book id "{bookId}" is not issued by any
747     user.')
```

```

748         print("-----")
749         returnBookMenu()
750     else:
751         print(f'Book with book id {bookId} does not available in the
752     digital library.')
```

```

753         print("-----")
754         returnBookMenu()
755
756
757 # Function to display the add user menu and handle user choices
758 def addUserMenu():
759     # Add user menu options
760     print("1. Home")
761     print("2. Back")
762     print("3. Exit")
763     userChoice = int(input("Enter your Choice to Continue : "))
764     print("-----")
765
766     # User choices handling
767     if userChoice == 1:
768         home()
769     elif userChoice == 2:
770         modifyUser()
771     elif userChoice == 3:
772         exiting()
773     else:
774         validOption()
775
776
777 # Function to add a new user
778 def addUser():
779     print("-----")
780     print("Add User")
781     print("-----")
782     # Get user input for user details
783     userId = int(input("Enter the User ID : "))
784     userName = input("Enter the User Name : ")
785     userPhoneNumber = input("Enter the User Phone Number : ")
786     userEmailId = input("Enter the User Email ID : ")
787     password = input("Enter the User Password : ")
788     print("-----")
789
790     c.execute("SELECT userId FROM users")
791     result = c.fetchall()
792     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()

```



```

859         else:
860             print(
861                 f'The user of user id "{userId}" does not enrolled in the
862 digital library.'
863             )
864             print("-----")
865             deleteUserMenu()
866         elif choice.lower() in ["no", "n"]:
867             print("-----")
868             print("User Not Deleted!")
869             print("-----")
870             deleteUserMenu()
871         else:
872             validOption()
873
874
875 # Function to display the update user menu and handle user choices
876 def updateUserMenu():
877     print("1. Home")
878     print("2. Back")
879     print("3. Exit")
880     userChoice = int(input("Enter your Choice to Continue : "))
881
882     # User choices handling
883     if userChoice == 1:
884         home()
885     elif userChoice == 2:
886         updateUser()
887     elif userChoice == 3:
888         exiting()
889     else:
890         validOption()
891
892
893 def notUser(userId):
894     print(f'The user of user id "{userId}" does not enrolled in the digital
895 library.')
896     print("-----")
897     updateBookMenu()
898
899
900 # Function to update user details
901 def updateUser():
902     print("-----")
903     print("Update User Details")
904     print("-----")
905     # Display user update options
906     print("1. Update the User ID")
907     print("2. Update the User Name")
908     print("3. Update the User Phone Number")
909     print("4. Update the User Email ID")
910     print("5. Update the User Password")
911     print("6. Home")
912     print("7. Back")
913     print("8. Exit")
914     # Get user choice
915     userChoice = int(input("Enter your Choice to Continue : "))
916     print("-----")
917
918     c.execute("SELECT userId FROM users")
919     result = c.fetchall()
920     db.commit()
921
922     if userChoice == 1:
923         # Update user ID
924         currentUserId = int(input("Enter the Current User ID : "))

```

```

925     newUserId = int(input("Enter the New User ID : "))
926
927     if (currentUserId,) in result:
928         c.execute(
929             "update users set userId=%s where userId=%s", (newUserId,
930 currentUserId)
931         )
932         db.commit()
933
934         print("User ID changed Successfully!")
935         print("-----")
936         updateUserMenu()
937     else:
938         notUser(currentUserId)
939
940     elif userChoice == 2:
941         # Update user name
942         userId = int(input("Enter the User ID : "))
943         newUserName = input("Enter the New User Name : ")
944
945         if (userId,) in result:
946             c.execute(
947                 "update users set userName=%s where userId=%s",
948 (newUserName, userId)
949             )
950             db.commit()
951
952             print("User Name changed Successfully!")
953             print("-----")
954             updateUserMenu()
955         else:
956             notUser(userId)
957
958     elif userChoice == 3:
959         # Update user phone number
960         userId = int(input("Enter the Current User ID : "))
961         newPhoneNumber = input("Enter the New Phone Number : ")
962
963         if (userId,) in result:
964             c.execute(
965                 "update users set phoneNumber=%s where userId=%s",
966 (newPhoneNumber, userId),
967             )
968             db.commit()
969
970             print("User Phone Number changed Successfully!")
971             print("-----")
972             updateUserMenu()
973         else:
974             notUser(userId)
975
976     elif userChoice == 4:
977         # Update user email ID
978         userId = int(input("Enter the Current User ID : "))
979         newEmailId = input("Enter the New Email ID : ")
980
981         if (userId,) in result:
982             c.execute(
983                 "update users set emailId=%s where userId=%s", (newEmailId,
984 userId)
985             )
986             db.commit()
987
988             print("User Email ID changed Successfully!")
989             print("-----")
990             updateUserMenu()

```

```

991         else:
992             notUser(userId)
993
994     elif userChoice == 5:
995         # Update user password
996         userId = int(input("Enter the Current User ID : "))
997         newPassword = input("Enter the New Password : ")
998         if (userId,) in result:
999             c.execute(
1000                 "update users set password=%s where userId=%s",
1001                 (newPassword, userId)
1002             )
1003             db.commit()
1004
1005             print("User Password changed Successfully!")
1006             print("-----")
1007             updateUserMenu()
1008         else:
1009             notUser(userId)
1010
1011     elif userChoice == 6:
1012         # Return to home
1013         home()
1014     elif userChoice == 7:
1015         # Go back to the previous menu
1016         modifyUser()
1017     elif userChoice == 8:
1018         # Exit the program
1019         exiting()
1020     else:
1021         validOption()
1022
1023
1024 # Function to modify user
1025 def modifyUser():
1026     print("-----")
1027     print("Modify User")
1028     print("-----")
1029     # Display user modification options
1030     print("1. Add User")
1031     print("2. Delete User")
1032     print("3. Update User Details")
1033     print("4. Home")
1034     print("5. Back")
1035     print("6. Exit")
1036     # Get user choice
1037     userChoice = int(input("Enter your Choice to Continue : "))
1038     print("-----")
1039
1040     # User choices handling
1041     if userChoice == 1:
1042         # Add a new user
1043         addUser()
1044     elif userChoice == 2:
1045         # Delete a user
1046         deleteUser()
1047     elif userChoice == 3:
1048         # Update user details
1049         updateUser()
1050     elif userChoice == 4:
1051         # Return to home
1052         home()
1053     elif userChoice == 5:
1054         # Return to the previous menu
1055         admin()
1056     elif userChoice == 6:

```

```

1057         # Exit the program
1058         exiting()
1059     else:
1060         validOption()
1061
1062
1063     # Display users menu options
1064     def displayUsersMenu():
1065         print("1. Home")
1066         print("2. Back")
1067         print("3. Exit")
1068         userChoice = int(input("Enter your Choice to Continue : "))
1069
1070         # User choices handling
1071         if userChoice == 1:
1072             home()
1073         elif userChoice == 2:
1074             admin()
1075         elif userChoice == 3:
1076             exiting()
1077         else:
1078             validOption()
1079
1080
1081     # Function to display all users
1082     def displayUsers():
1083         print("-----")
1084         print("Display Users")
1085         print("-----")
1086         # Fetch all users from the database
1087         c.execute("SELECT * FROM users ORDER BY userId")
1088         result = c.fetchall()
1089         db.commit()
1090
1091         if result:
1092             # Display user information
1093             print("Users enrolled in the Digital Library are :")
1094             i = 0
1095             for row in result:
1096                 i += 1
1097                 r = length(i)
1098                 print(f"{i}. User ID : {row[0]}")
1099                 print(" " * r + f"User Name : {row[1]}")
1100                 print(" " * r + f"Phone Number : {row[2]}")
1101                 print(" " * r + f"Email ID : {row[3]}")
1102                 print(" " * r + f"Admin Status : {row[5]}")
1103                 print("-----")
1104             displayUsersMenu()
1105
1106         else:
1107             print("No users found.")
1108             print("-----")
1109             displayUsersMenu()
1110
1111
1112     # Search user menu options
1113     def searchUsersMenu():
1114         print("1. Home")
1115         print("2. Back")
1116         print("3. Exit")
1117         userChoice = int(input("Enter your Choice to Continue : "))
1118
1119         # User choices handling
1120         if userChoice == 1:
1121             home()
1122         elif userChoice == 2:

```

```

1123         searchUsers()
1124     elif userChoice == 3:
1125         exiting()
1126     else:
1127         validOption()
1128
1129
1130 # Function to search users by ID
1131 def searchUsersbyId():
1132     print("-----")
1133     print("Search Users by User ID")
1134     print("-----")
1135     # Get user ID to search
1136     userId = int(input("Enter the User ID to search the User : "))
1137
1138     # Search for the user in the database
1139     c.execute("SELECT * FROM users WHERE userId=%s", (userId,))
1140     result = c.fetchall()
1141     db.commit()
1142
1143     if result:
1144         # Display user information if found
1145         print(f'User enrolled in the Digital Library with the User ID
1146 "{userId}" is :')
1147         i = 0
1148         for row in result:
1149             i += 1
1150             r = length(i)
1151             print(f"{i}. User ID : {row[0]}")
1152             print(" " * r + f"User Name : {row[1]}")
1153             print(" " * r + f"Phone Number : {row[2]}")
1154             print(" " * r + f"Email ID : {row[3]}")
1155             print(" " * r + f"Admin Status : {row[5]}")
1156             print("-----")
1157         searchUsersMenu()
1158
1159     else:
1160         # Handle case when no user is found
1161         print(f'No user found with the user id "{userId}.'.)
1162         print("-----")
1163         searchUsersMenu()
1164
1165
1166 # Function to search users by keyword
1167 def searchUsersbyKeyword():
1168     print("-----")
1169     print("Search Users by Keyword")
1170     print("-----")
1171     # Get keyword input from the user
1172     keyword = input("Enter a Keyword to search Users : ")
1173
1174     # Search for users with the given keyword in their names
1175     c.execute(
1176         "SELECT * FROM users WHERE userName LIKE '%{}%' ORDER BY
1177         userId".format(keyword)
1178     )
1179     result = c.fetchall()
1180     db.commit()
1181
1182     if result:
1183         # Display user information if users are found
1184         print(
1185             f'Users enrolled in the Digital Library with the Keyword
1186 "{keyword}" are :')
1187         )
1188         i = 0

```

```

1189     for row in result:
1190         i += 1
1191         r = length(i)
1192         print(f"{i}. User ID : {row[0]}")
1193         print(" " * r + f"User Name : {row[1]}")
1194         print(" " * r + f"Phone Number : {row[2]}")
1195         print(" " * r + f"Email ID : {row[3]}")
1196         print(" " * r + f"Admin Status : {row[5]}")
1197         print("-----")
1198     searchUsersMenu()
1199
1200     else:
1201         # Handle case when no user is found
1202         print(f'No users found with the keyword "{keyword}.'.')
1203         print("-----")
1204         searchUsersMenu()
1205
1206
1207 # Function to search users
1208 def searchUsers():
1209     print("-----")
1210     print("Search Users")
1211     print("-----")
1212     # User search menu
1213     print("1. Search by User ID")
1214     print("2. Search by Keyword")
1215     print("3. Home")
1216     print("4. Back")
1217     print("5. Exit")
1218     userChoice = int(input("Enter your Choice to Continue : "))
1219     print("-----")
1220
1221     # User choices handling
1222     if userChoice == 1:
1223         searchUsersbyId()
1224     elif userChoice == 2:
1225         searchUsersbyKeyword()
1226     elif userChoice == 3:
1227         home()
1228     elif userChoice == 4:
1229         admin()
1230     elif userChoice == 5:
1231         exiting()
1232     else:
1233         validOption()
1234
1235
1236 # Function to modify books
1237 def modifyBook():
1238     print("-----")
1239     print("Modify Book")
1240     print("-----")
1241     # Book modification menu
1242     print("1. Add Book")
1243     print("2. Delete Book")
1244     print("3. Update Book Details")
1245     print("4. Home")
1246     print("5. Back")
1247     print("6. Exit")
1248     userChoice = int(input("Enter your Choice to Continue : "))
1249     print("-----")
1250
1251     # User choices handling
1252     if userChoice == 1:
1253         addBook()
1254     elif userChoice == 2:

```

```

1255         deleteBook()
1256     elif userChoice == 3:
1257         updateBook()
1258     elif userChoice == 4:
1259         home()
1260     elif userChoice == 5:
1261         admin()
1262     elif userChoice == 6:
1263         exiting()
1264     else:
1265         validOption()
1266
1267
1268 # Function to manage notes
1269 def notes():
1270     print("-----")
1271     print("Notes")
1272     print("-----")
1273     # Display menu options
1274     print("1. Modify Note")
1275     print("2. Display Notes")
1276     print("3. Search Notes")
1277     print("4. Home")
1278     print("5. Back")
1279     print("6. Exit")
1280     # Get user choice
1281     userChoice = int(input("Enter your Choice to Continue : "))
1282     print("-----")
1283
1284     # Handle user choices
1285     if userChoice == 1:
1286         modifyNote()
1287     elif userChoice == 2:
1288         displayNotes()
1289     elif userChoice == 3:
1290         searchNotes()
1291     elif userChoice == 4:
1292         home()
1293     elif userChoice == 5:
1294         user()
1295     elif userChoice == 6:
1296         exiting()
1297     else:
1298         validOption()
1299
1300
1301 # Function to display the add note menu and handle user choices
1302 def addNoteMenu():
1303     print("1. Home")
1304     print("2. Back")
1305     print("3. Exit")
1306     # Get user choice
1307     userChoice = int(input("Enter your Choice to Continue : "))
1308
1309     # Handle user choices
1310     if userChoice == 1:
1311         home()
1312     elif userChoice == 2:
1313         modifyNote()
1314     elif userChoice == 3:
1315         exiting()
1316     else:
1317         validOption()
1318
1319
1320 # Function to add note

```

```

1321 def addNote():
1322     print("-----")
1323     print("Add Note")
1324     print("-----")
1325     # Get note details from the user
1326     noteNumber = int(input("Enter the Note Number : "))
1327     noteTitle = input("Enter the Note Title : ")
1328     noteDescription = input("Enter the Note Description : ")
1329     print("-----")
1330
1331     c.execute("SELECT noteNumber FROM notes where userId=%s", (USERID,))
1332     result = c.fetchall()
1333     db.commit()
1334
1335     if (noteNumber,) in result:
1336         print(
1337             f'The note of note number "{noteNumber}" is already exists in
1338 the digital library.'
1339         )
1340         print("-----")
1341         addNoteMenu()
1342
1343     else:
1344         # Execute SQL query to insert the note into the database
1345         c.execute(
1346             "INSERT INTO notes (userId, noteNumber, noteTitle,
1347 noteDescription, updateDate, updateTime) VALUES (%s, %s, %s, %s,
1348 CURRENT_DATE, CURRENT_TIME)",
1349             (USERID, noteNumber, noteTitle, noteDescription),
1350         )
1351         db.commit()
1352
1353         print(f'The note of note number "{noteNumber}" is added
1354 successfully.')
1355         print("-----")
1356         addNoteMenu()
1357
1358
1359 # Function to display the delete note menu and handle user choices
1360 def deleteNoteMenu():
1361     # Display menu options after deleting the note
1362     print("1. Home")
1363     print("2. Back")
1364     print("3. Exit")
1365     # Get user choice
1366     userChoice = int(input("Enter your Choice to Continue : "))
1367     print("-----")
1368
1369     # Handle user choices
1370     if userChoice == 1:
1371         home()
1372     elif userChoice == 2:
1373         modifyNote()
1374     elif userChoice == 3:
1375         exiting()
1376     else:
1377         validOption()
1378
1379
1380 # Function to delete a note
1381 def deleteNote():
1382     print("-----")
1383     print("Delete Note")
1384     print("-----")
1385     # Get note number to be deleted from the user
1386     noteNumber = int(input("Enter the Note Number to Delete the Note : "))

```



```

1387     choice = input("Are you sure to delete the Note? (Yes/No) : ")
1388     print("-----")
1389
1390     c.execute("SELECT noteNumber FROM notes where userId=%s", (USERID,))
1391     result = c.fetchall()
1392     db.commit()
1393
1394     if choice.lower() in ["yes", "y"]:
1395         if (noteNumber,) in result:
1396             # Execute SQL query to delete the note from the database
1397             c.execute(
1398                 "delete FROM notes WHERE userId=%s and noteNumber=%s",
1399                 (USERID, noteNumber),
1400             )
1401             db.commit()
1402
1403             print(f'The note of note number "{noteNumber}" is deleted
1404 successfully.')
1405             print("-----")
1406             deleteNoteMenu()
1407
1408         else:
1409             print(
1410                 f'The note of note number "{noteNumber}" does not exists in
1411 the digital library.'
1412             )
1413             print("-----")
1414             deleteNoteMenu()
1415         elif choice.lower() in ["no", "n"]:
1416             print("-----")
1417             print("Note Not Deleted!")
1418             print("-----")
1419             deleteNoteMenu()
1420         else:
1421             validOption()
1422
1423
1424 # Function to display the update notes menu and handle user choices
1425 def updateNotesMenu():
1426     print("1. Home")
1427     print("2. Back")
1428     print("3. Exit")
1429     # Get user choice
1430     userChoice = int(input("Enter your Choice to Continue : "))
1431     print("-----")
1432
1433     # Handle user choices
1434     if userChoice == 1:
1435         home()
1436     elif userChoice == 2:
1437         updateNotes()
1438     elif userChoice == 3:
1439         exiting()
1440     else:
1441         validOption()
1442
1443
1444 def notNote(noteNumber):
1445     print(
1446         f'The note of note number "{noteNumber}" does not exists in the
1447 digital library.'
1448     )
1449     print("-----")
1450     updateNotesMenu()
1451
1452

```

```

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")

```

```

1585     print("3. Update Notes")
1586     print("4. Home")
1587     print("5. Back")
1588     print("6. Exit")
1589     # Get user choice
1590     userChoice = int(input("Enter your Choice to Continue : "))
1591     print("-----")
1592
1593     # Handle user choices
1594     if userChoice == 1:
1595         addNote()
1596     elif userChoice == 2:
1597         deleteNote()
1598     elif userChoice == 3:
1599         updateNotes()
1600     elif userChoice == 4:
1601         home()
1602     elif userChoice == 5:
1603         admin()
1604     elif userChoice == 6:
1605         exiting()
1606     else:
1607         validOption()
1608
1609
1610 # Function to display the display notes menu and handle user choices
1611 def displayNotesMenu():
1612     print("1. Home")
1613     print("2. Back")
1614     print("3. Exit")
1615     userChoice = int(input("Enter your Choice to Continue : "))
1616     print("-----")
1617
1618     # Handle user choices
1619     if userChoice == 1:
1620         home()
1621     elif userChoice == 2:
1622         user()
1623     elif userChoice == 3:
1624         exiting()
1625     else:
1626         validOption()
1627
1628
1629 # Function to display notes
1630 def displayNotes():
1631     # Fetch all notes from the database
1632     c.execute("SELECT * FROM notes ORDER BY noteNumber")
1633     result = c.fetchall()
1634     db.commit()
1635
1636     # Check if there are notes available
1637     if result:
1638         print(f"Notes available in the Digital Library are :")
1639         i = 0
1640         for row in result:
1641             i += 1
1642             r = length(i)
1643             print(f"{i}. Note Number : {row[1]}")
1644             print(" " * r + f"Note Title : {row[2]}")
1645             print(" " * r + f"Note Description : {row[3]}")
1646             print(" " * r + f"Update Date : {row[4]}")
1647             print(" " * r + f"Update Time : {row[5]}")
1648             print("-----")
1649         displayNotesMenu()
1650

```

```

1651     else:
1652         # If no notes are found
1653         print("No notes found.")
1654         print("-----")
1655         displayNotesMenu()
1656
1657
1658 # Function to display the search notes menu and handle user choices
1659 def searchNotesMenu():
1660     print("1. Home")
1661     print("2. Back")
1662     print("3. Exit")
1663     userChoice = int(input("Enter your Choice to Continue : "))
1664
1665     # Handle user choices
1666     if userChoice == 1:
1667         home()
1668     elif userChoice == 2:
1669         searchNotes()
1670     elif userChoice == 3:
1671         exiting()
1672     else:
1673         validOption()
1674
1675
1676 # Function to search notes by note number
1677 def searchNotesbynoteNumber():
1678     # Get the note number to search
1679     noteNumber = int(input("Enter the Note Number to search the Note : "))
1680
1681     # Execute SQL query to fetch notes with the given note number
1682     c.execute("SELECT * FROM notes WHERE bookId=%s", (noteNumber,))
1683     result = c.fetchall()
1684     db.commit()
1685
1686     # Check if notes are found
1687     if result:
1688         print(
1689             f'Note available in the Digital Library with the Note Number
1690 {noteNumber}" is :'
1691         )
1692         i = 0
1693         for row in result:
1694             i += 1
1695             r = length(i)
1696             print(f"{i}. Note Number : {row[1]}")
1697             print(" " * r + f"Note Title : {row[2]}")
1698             print(" " * r + f"Note Description : {row[3]}")
1699             print("-----")
1700         searchNotesMenu()
1701
1702     else:
1703         # If no notes are found with the given note number
1704         print(f'No note found with the note number "{noteNumber}.'.')
1705         print("-----")
1706         searchNotesMenu()
1707
1708
1709 # Function to search notes by keyword
1710 def searchNotesbyKeyword():
1711     print("-----")
1712     print("Search Notes by Keyword")
1713     print("-----")
1714     # Get keyword from user
1715     keyword = input("Enter a Keyword to search Notes : ")
1716

```

```

1717     # Execute SQL query to fetch notes with the given keyword in the title
1718     c.execute(
1719         "SELECT * FROM notes WHERE noteTitle LIKE '%{}%' ORDER BY
1720     noteNumber".format(
1721         keyword
1722     )
1723     )
1724     result = c.fetchall()
1725     db.commit()
1726
1727     # Check if notes are found
1728     if result:
1729         print(
1730             f'Notes available in the Digital Library with the Keyword
1731     "{keyword}" are :'
1732         )
1733         i = 0
1734         for row in result:
1735             i += 1
1736             r = length(i)
1737             print(f"{i}. Note Number : {row[1]}")
1738             print(" " * r + f>Note Title : {row[2]}")
1739             print(" " * r + f>Note Description : {row[3]}")
1740             print("-----")
1741         searchNotesMenu()
1742
1743     else:
1744         # If no notes are found with the given keyword
1745         print(f'No notes found with the keyword "{keyword}.'.)
1746         print("-----")
1747         searchNotesMenu()
1748
1749
1750     # Function to handle note searching
1751     def searchNotes():
1752         print("-----")
1753         print("Search Notes")
1754         print("-----")
1755         # Display search options
1756         print("1. Search by Note Number")
1757         print("2. Search by Keyword")
1758         print("3. Home")
1759         print("4. Back")
1760         print("5. Exit")
1761         # Get user choice
1762         userChoice = int(input("Enter your Choice to Continue : "))
1763         print("-----")
1764
1765         # Handle user choices
1766         if userChoice == 1:
1767             searchNotesbynoteNumber()
1768         elif userChoice == 2:
1769             searchNotesbyKeyword()
1770         elif userChoice == 3:
1771             notes()
1772         elif userChoice == 4:
1773             modifyNote()
1774         elif userChoice == 5:
1775             exiting()
1776         else:
1777             validOption()
1778
1779
1780     # Function to display the change admin menu and handle user choices
1781     def changeAdminMenu():
1782         print("1. Home")

```

```

1783     print("2. Back")
1784     print("3. Exit")
1785     userChoice = int(input("Enter your Choice to Continue : "))
1786     print("-----")
1787
1788     # Handle user choices
1789     if userChoice == 1:
1790         home()
1791     elif userChoice == 2:
1792         admin()
1793     elif userChoice == 3:
1794         exiting()
1795     else:
1796         validOption()
1797
1798
1799 # Function to change the admin status
1800 def changeAdmin():
1801     print("-----")
1802     print("Change Admin")
1803     print("-----")
1804     # Get new admin's ID and password from the user
1805     newAdminId = int(input("Enter the New Admin's User ID : "))
1806     newAdminPassword = input("Enter the New Admin's Password : ")
1807     choice = input("Are you sure to change the Admin? (Yes/No) : ")
1808     print("-----")
1809
1810     # Check if the entered user ID exists
1811     c.execute("SELECT password FROM users WHERE userId=%s", (newAdminId,))
1812     result = c.fetchall()
1813     db.commit()
1814
1815     # Check the user's choice to proceed or cancel
1816     if choice.lower() in ["yes", "y"]:
1817         # If the user ID is not valid, print an error message
1818         if len(result) == 0:
1819             print("Please enter a valid user id!")
1820         else:
1821             # If the entered password matches the user's password
1822             if newAdminPassword == result[0][0]:
1823                 # Update admin status for all users
1824                 c.execute(
1825                     "UPDATE users SET adminStatus='not admin' WHERE
1826 adminStatus ='admin'"
1827                 )
1828                 c.execute(
1829                     "UPDATE users SET adminStatus='admin' WHERE userId
1830 =%s",
1831                     (newAdminId,),
1832                 )
1833                 db.commit()
1834
1835                 print("Admin Changed Successfully!")
1836                 print("-----")
1837                 changeAdminMenu()
1838
1839             else:
1840                 print("Please enter a valid password!")
1841     elif choice.lower() in ["no", "n"]:
1842         print("Admin Not Changed!")
1843         print("-----")
1844         changeAdminMenu()
1845     else:
1846         validOption()
1847
1848

```

```

1849 # Function to authenticate admin
1850 def authAdmin():
1851     print("-----")
1852     print("Admin Authentication")
1853     print("-----")
1854     adminId = int(input("Enter the Admin's User ID : "))
1855     adminPassword = input("Enter the Admin's User Password : ")
1856
1857     # Check if the entered admin ID exists
1858     c.execute("SELECT password FROM users WHERE userId=%s", (adminId,))
1859     result = c.fetchall()
1860     db.commit()
1861
1862     # If the entered admin ID is not valid, print an error message
1863     if len(result) == 0:
1864         print("-----")
1865         print("Please enter a valid user id!")
1866         print("-----")
1867     else:
1868         # If the entered password matches the admin's password
1869         if adminPassword == result[0][0]:
1870             global USERID
1871             USERID = adminId
1872             print("\033[0;35m-----\033[0;0m")
1873             print("\033[0;36mAdmin is verified successfully.\033[0;0m")
1874             print("\033[0;35m-----\033[0;0m")
1875             admin() # Call the admin menu
1876         else:
1877             print("Please enter a valid password!")
1878             print("-----")
1879
1880
1881 # Function to display the admin menu
1882 def admin():
1883     print("-----")
1884     print("Admin")
1885     print("-----")
1886     print("1. Login into User Panel")
1887     print("2. Modify User")
1888     print("3. Display Users")
1889     print("4. Search Users")
1890     print("5. Modify Book")
1891     print("6. Issue Book")
1892     print("7. Return Book")
1893     print("8. Change Admin")
1894     print("9. Home")
1895     print("10. Back")
1896     print("11. Exit")
1897     userChoice = int(input("Enter your Choice to Continue : "))
1898     print("-----")
1899
1900     # Handle user choices
1901     if userChoice == 1:
1902         print("You are successfully login into user panel.")
1903         print("-----")
1904
1905         user()
1906     elif userChoice == 2:
1907         modifyUser()
1908     elif userChoice == 3:
1909         displayUsers()
1910     elif userChoice == 4:
1911         searchUsers()
1912     elif userChoice == 5:
1913         modifyBook()
1914     elif userChoice == 6:

```



```

1915         issueBook()
1916     elif userChoice == 7:
1917         returnBook()
1918     elif userChoice == 8:
1919         changeAdmin()
1920     elif userChoice == 9:
1921         home()
1922     elif userChoice == 10:
1923         authAdmin()
1924     elif userChoice == 11:
1925         exiting()
1926     else:
1927         validOption()
1928
1929
1930 # Function to authenticate a user
1931 def authUser():
1932     print("-----")
1933     print("User Authentication")
1934     print("-----")
1935     userId = int(input("Enter the User ID : "))
1936     password = input("Enter the User Password : ")
1937
1938     # Check if the entered user ID exists
1939     c.execute("SELECT password FROM users WHERE userId=%s", (userId,))
1940     result = c.fetchall()
1941     db.commit()
1942
1943     # If the entered user ID is not valid, print an error message
1944     if len(result) == 0:
1945         print("-----")
1946         print("Please enter a valid user id!")
1947         print("-----")
1948     else:
1949         # If the entered password matches the user's password
1950         if password == result[0][0]:
1951             global USERID
1952             USERID = userId
1953             print("\033[0;35m-----\033[0;0m")
1954             print("\033[0;36mUser is verified successfully.\033[0;0m")
1955             print("\033[0;35m-----\033[0;0m")
1956             user() # Call the user menu
1957         else:
1958             print("Please Enter a Valid Password!")
1959             print("-----")
1960
1961
1962 # Function to search & display the wikipedia articles
1963 def wikipediaArticles():
1964     # Function to fetch article details
1965     def fetchingArticle(keyword, articleLength=1500):
1966         # Creating a Wikipedia API object
1967         wiki = wikipediaapi.Wikipedia(language="en", user_agent="digital-
1968 library/1.1")
1969         # Fetching the page for the given search query
1970         page = wiki.page(keyword)
1971
1972         # Checking if the page exists
1973         if not page.exists():
1974             print(
1975                 f'Sorry, the Wikipedia Article for the keyword "{keyword}"
1976 does not exists.'
1977             )
1978             print("-----")
1979         else:
1980             # Displaying article title

```

```

1981         print("Title : ")
1982         print(page.title)
1983         print("URL : ")
1984         print(page.fullurl)
1985         # Displaying a summary of the article within the specified
1986 length
1987         print("Summary : ")
1988
1989         start = 0
1990         end = 157
1991         article = page.summary[:articleLength]
1992
1993         while end <= articleLength:
1994             print(article[start:end])
1995             start += 157
1996             end += 157
1997         else:
1998             print("-----")
1999
2000     print("-----")
2001     print("Search Articles")
2002     print("-----")
2003     # Taking user input for the keyword and article length
2004     keyword = input("Enter the Keyword for searching the Wikipedia Article
2005 : ")
2006     articleLength = int(input("Enter the Article Length : "))
2007     print("-----")
2008
2009     # Calling the function to fetch and display the article
2010     fetchingArticle(keyword, articleLength)
2011
2012     userMenu()
2013
2014
2015 # Function to search & display the news
2016 def news():
2017     def fetchNews(apiKey, country="in", category="science", numArticles=5):
2018         url = f"https://newsapi.org/v2/top-headlines"
2019         params = {
2020             "apiKey": apiKey,
2021             "country": country,
2022             "category": category,
2023             "pageSize": numArticles,
2024         }
2025         response = requests.get(url, params=params)
2026
2027         if response.status_code == 200:
2028             news_data = response.json()
2029             articles = news_data.get("articles", [])
2030
2031             for i, article in enumerate(articles, start=1):
2032                 print(f"{i}. {article['title']}")
2033                 print(f"    Source: {article['source']['name']}")
2034                 print(f"    URL: {article['url']}")
2035                 print("-----")
2036
2037             else:
2038                 print(f"Error {response.status_code}: {response.text}")
2039                 print("-----")
2040
2041     API_KEY = "YOUR_API_KEY"
2042
2043     print("-----")
2044     print("News")
2045     print("-----")
2046     print("Country codes are : ")

```

```

2047     print("https://newsapi.org/sources")
2048     print("Categories are : ")
2049     print("business, entertainment, general, health, science, sports,
2050     technology")
2051     print("-----")
2052     country = input("Enter the Country Code : ")
2053     category = input("Enter the Category : ")
2054     numArticles = int(input("Enter the Number of Articles : "))
2055     print("-----")
2056
2057     fetchNews(API_KEY, country, category, numArticles)
2058
2059     userMenu()
2060
2061
2062     # Function to display the issued books details of a user
2063     def issuedBooksDetails():
2064         print("-----")
2065         print("Issued Books Details")
2066         print("-----")
2067         returnPolicy()
2068
2069         c.execute(
2070             "SELECT * FROM issuedBooksDetails WHERE userId=%s ORDER BY bookId",
2071             (USERID,)
2072         )
2073         result = c.fetchall()
2074         db.commit()
2075
2076         if result == []:
2077             print("No Books Issued!")
2078             print("-----")
2079             userMenu()
2080
2081         else:
2082             i = 0
2083             for row in result:
2084                 i += 1
2085                 r = length(i)
2086                 print(f"{i}. Book ID : ", row[1])
2087                 print(" " * r + "Book Name : ", row[2])
2088                 print(" " * r + "Issue Date : ", row[3])
2089                 print(" " * r + "Issue Time : ", row[4])
2090                 print(" " * r + "Return Date : ", row[5])
2091                 print(" " * r + "Return Time : ", row[6])
2092                 print(" " * r + "Fine(in Rs.) : ", row[7])
2093                 print("-----")
2094             userMenu()
2095
2096
2097     # Function to display the user menu
2098     def user():
2099         print("-----")
2100         print("User")
2101         print("-----")
2102         # Check if the entered user ID exists
2103         c.execute('SELECT userId FROM users WHERE adminStatus="admin"')
2104         result = c.fetchall()
2105         db.commit()
2106
2107         if result[0][0] == USERID:
2108             print("1. Login into Admin Panel")
2109             print("2. About the Library")
2110             print("3. News")
2111             print("4. Wikipedia Articles")
2112             print("5. Display Books")

```

```

2113     print("6.  Search Books")
2114     print("7.  Issued Books Details")
2115     print("8.  Notes")
2116     print("9.  Home")
2117     print("10. Back")
2118     print("11. Exit")
2119     userChoice = int(input("Enter your Choice to Continue : "))
2120     print("-----")
2121
2122     # Handle user choices
2123     if userChoice == 1:
2124         print("You are successfully login into admin panel.")
2125         print("-----")
2126
2127         admin()
2128     elif userChoice == 2:
2129         aboutLibrary()
2130     elif userChoice == 3:
2131         news()
2132     elif userChoice == 4:
2133         wikipediaArticles()
2134     elif userChoice == 5:
2135         displayBooks()
2136     elif userChoice == 6:
2137         searchBooks()
2138     elif userChoice == 7:
2139         issuedBooksDetails()
2140     elif userChoice == 8:
2141         notes()
2142     elif userChoice == 9:
2143         home()
2144     elif userChoice == 10:
2145         authUser()
2146     elif userChoice == 11:
2147         exiting()
2148     else:
2149         validOption()
2150 else:
2151     print("1.  About Library")
2152     print("2.  News")
2153     print("3.  Wikipedia Articles")
2154     print("4.  Display Books")
2155     print("5.  Search Books")
2156     print("6.  Issued Books Details")
2157     print("7.  Notes")
2158     print("8.  Home")
2159     print("9.  Back")
2160     print("10. Exit")
2161     userChoice = int(input("Enter your Choice to Continue : "))
2162     print("-----")
2163
2164     # Handle user choices
2165     if userChoice == 1:
2166         aboutLibrary()
2167     elif userChoice == 2:
2168         news()
2169     elif userChoice == 3:
2170         wikipediaArticles()
2171     elif userChoice == 4:
2172         displayBooks()
2173     elif userChoice == 5:
2174         searchBooks()
2175     elif userChoice == 6:
2176         issuedBooksDetails()
2177     elif userChoice == 7:
2178         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",
width=1000)
2197             )
2198             print(
2199                 pyfiglet.figlet_format("Digital Library", font="banner3",
width=1000)
2200             + "\033[0;0m"
2201             )
2202             print("\033[1;32m~::~::~::~::~::~::~::~::~::~::~\033[0;0m")
2203             print("=====")
2204             print("-----")
2205             print("Home")
2206             print("-----")
2207             print("1. Admin")
2208             print("2. User")
2209             print("3. Exit")
2210             userChoice = int(input("Enter your Choice to Continue : "))
2211             print("-----")
2212
2213         # Handle user choices
2214         if userChoice == 1:
2215             authAdmin()
2216         elif userChoice == 2:
2217             authUser()
2218         elif userChoice == 3:
2219             exiting()
2220         else:
2221             validOption()
2222
2223     # Call the main menu function
2224     home()
2225
2226
2227
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 operates 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 Chairman 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 reputation 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:

```
=====
#####
## ##### ## ##### ## ##### ##### ## ## #####
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ##### ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ##### ##### ##### ## ## ##### ## ## #####

##### ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
## ## ## ## ## ## ## ## ## ## ## ## ## ## ##
##### ## ## ## ## ## ## ## ## ## ## ## ## ## ##

=====
-----
Home
-----
1. Admin
2. User
3. Exit
Enter your Choice to Continue : 1
-----
```

Admin Authentication:

```
-----
Admin Authentication
-----
Enter the Admin's User ID : 1025
Enter the Admin's User Password : Bman@1025
-----
Admin is verified successfully.
-----
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 : 2
-----
Modify User
-----
1. Add User
2. Delete User
3. Update User Details
4. Home
5. Back
6. Exit
Enter your Choice to Continue : 1
-----
```

Adding a new user:

```
-----
Add User
-----
Enter the User ID : 1028
Enter the User Name : Eman
Enter the User Phone Number : 8564231547
Enter the User Email ID : eman1028@gmail.com
Enter the User Password : Eman@1028
-----
```

```
-----  
-----  
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
-----
=====
```



```
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-Enders 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, aeronautics research, and space research. Established in 1958, NASA succeeded the National Advisory Committee for Aeronautics (NACA) to give the U.S. space development effort a distinctly civilian orientation, emphasizing peaceful applications in space science. NASA has since led most American space exploration, including 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 Commercial Crew spacecraft, and the planned Lunar Gateway space station.

NASA's science is focused on better understanding Earth through the Earth Observing System; advancing heliophysics through the efforts of the Science Mission Directorate's Heliophysics Research Program; exploring bodies throughout the Solar System with advanced robotic spacecraft such as New Horizons and planetary rovers such as Perseverance; and researching astrophysics topics, such as the Big Bang, through the James Webb Space Telescope, the Great Observatories and associated programs. NASA's Launch Services Program provides oversight of launch operations 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