SALWAN PUBLIC SCHOOL

G.L, Salwan Rd, Rajinder Nagar,
New Delhi, Delhi 110060

SERVICE BEFORE SELF

SALWAN PUBLIC SCHOOL

A PROJECT REPORT ON

LIBRARY MANAGEMENT SYSTEM

SUBMITTED TO:	**SUBMITTED BY:**
MS. SUPREET KAUR	**HARDIK BHATIA**
PGT(COMP.SCI)	**CLASS: XII(Non-Medical)**
SALWAN PUBLIC SCHOOL	**ROLL NO.:**
	SUBJECT – COMPUTER SCIENCE (083)

TABLE OF CONTENT

CERTIFICATE

This is to certify that **HARDIK BHATIA** of class **XII (Non-Medical), (board roll no.)** of **SALWAN PUBLIC SCHOOL** has done his/her project on **LIBRARY MANAGEMENT SIMULATION** under my supervision.

He has taken interest and has shown at most sincerity in completion of this project.

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

(MS. SUPREET KAUR)

PGT(COMPUTER SCIENCE)

SALWAN PUBLIC SCHOOL

ACKNOWLEDGEMENT

It is with pleasure that I acknowledge my sincere gratitude to our teacher, Ms. ***Supreet Kaur*** who taught and undertook the responsibility of teaching the subject of computer science. I have been greatly benefited from her classes.

My sincere thanks goes to our Principal ***Ms. Jyotsna Grover*** who has always been a source of encouragement and support and without whose inspiration, this project would not have been successful.

Finally, I would like to express my sincere appreciation for all my classmates for their friendship & the fine times that we all shared together.

Last but not least, I would like to thank all those who had helped directly or indirectly towards the completion of this project.

ABOUT PYTHON

Python is a powerful high-level programming language. It is a relatively new language and is used for jobs like automation and, web development, scientific and numeric applications and so on. It is beginner-friendly and easy to learn as compared to languages like C++ and Java owing to it's simpler syntax and diverse libraries like NumPy and SciPy which save a lot of time and effort.

It was developed by Guido van Rossum in the 1980s and till date remains one of the easiest yet powerful programming languages in existence.

Python is an interpreted language and that presents some limitations regarding efficiency but it outshines all other languages in areas where it works best.

Overall Python is an easy-to-learn modern language which plays an important role in many day-to-day applications we use yet do not fully appreciate.

ABOUT MYSQL

MySQL is a relational database management system based on SQL (structured query language). It is used to store data in an orderly fashion and has a powerful query system which can be used to select specific data.

It allows for creation of tables to store information which further allow for constraints and checks that then allow the table to hold data of specific kind which allows for storage of characterized data according to the needs of the project or simply user

MySQL is one of the most widely used database software which can be further linked with languages like Python to yield even more use in application-based software such as this one.

ABOUT THIS PROJECT

This project represents a functioning model of a typical library. It is complete with a database of users, administrator accounts and a fully functioning ecosystem of books which is highly interlinked to facilitate the complexities that go into making a library usable for both the user and the librarian.

My primary focus while making Librarium was not just to make a functioning library management system, but to make it a complete program of its own. The functions are just a part of the program. A major effort was put into error handling. I introduced conformations in the form of (y/n) inputs along the important parts of the program which made significant changes to the database and made adequate use of try/except blocks throughout the program to avoid any abrupt errors which lead to the program stopping.

When you start the program there is a log-in screen which can register new users, a home screen you can access only once you’ve logged in. Functions are suitably placed and administrator privileges are disabled for regular users, thereby creating a realistic experience.

My goal with this project was to make a functioning library and I accomplished that by going beyond the scope of books by actually talking to real librarians and applying my own experience with libraries to make it feel like this program would be actually useful, if it were to be implemented. I not only got to learn a lot about just programming but also got an insight into what it feels like to develop an application from scratch: to make, modify and optimize.

During this process I found myself searching for errors in the code, looking for things to improve upon and finding the scope of errors which could lead to malfunctioning of the program to minimize the risk of loss of information. Even though all books I tested this program with were completely imaginary, I found myself working as if they represented real entities and that every time my code malfunctioned, it caused inconvenience to either the user or the librarian which made this a great challenge at hand and eventually an even greater learning opportunity.

CODING

```
from os import system
import mysql.connector
import datetime
import random
import time
db = mysql.connector.connect(
        host="localhost",
        user="root",
        password="root",
        database="library")
mycursor = db.cursor()
#Secondary Functions


def guidelines():
    print("General guidelines: \n"
       "-> All issued books must be returned within 14 days of issuing.\n"
       "-> A fine of ruppees 30 will be levied from users who do not return the book within a month of issuing and ruppees 200 for every month after that.\n"
       "-> If a book is not returned within 5 months of issuing it will lead to a permanent cancellation of your library card followed by a potential visit from a collecting officer.\n"
       "-> Any defacement to the book may result in suspension of your library card or a fine or both.\n")


#Main Functions
def bookWrite():
    global AdminLogin
    if AdminLogin == True:
        print("*" * 50, "ENTER BOOKS", "*" * 50)
    else:
        print("ERROR: Access denied.")
        raise NameError
    system('cls')
    while True:
        try:
            print("Please enter the details of book\n")
            book_id = int(input("Book ID: "))
            book_name = str(input("Book name: "))
            book_genre = input("Book genre: ")
            book_author = input("Name of the author: ")
            book_publisher = input("Name of publisher: ")
            book_count = int(input("Number of copies: "))
            book_summary = input("Enter a short summary for the book: ")
            print("\n")


            cmnd = "INSERT into books values (%s,%s,%s,%s,%s,%s,%s,0)"
            val = (book_id, book_name, book_genre, book_author, book_publisher, book_count, book_summary)
            mycursor.execute(cmnd, val)
            ch1 = input("Confirm entry? (y/n): ")
            if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                db.commit()
                print("SUCCESS: The new records were stored in the database.")
            elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':
                print("ABORTED: This record was not entered into the database.")
            else:
                print("Interpreting input as 'no'. This record was not entered into database.")
        except:
            print("ERROR: You have entered an invalid input for one of the fields. Please check your input and try again.\n"
                 "-> Book ID should be unique and less than 3 digits.\n"
                 "-> Only integer inputs are allowed in Book ID and number of copies\n"
                 "-> The string inputs may not exceed 40 characters.\n\n")
        ch = input("Would you like to enter another record? (y/n): ")
        ("\n\n")
        if ch=='y':
            system('cls')
            continue
        elif ch=='n' :
            break
        else:
            print("Interpretting vague input as NO.")
            break
    ch2 = input("Press enter to return to main menu...")
    system('cls')


def bookEdit():
    if AdminLogin == True:
        pass
    else:
        erc = 1
    system('cls')
    print("*" * 50, "UPDATE BOOK", "*" * 50)
    db = mysql.connector.connect(
        host="localhost",
        user="root",
        password ="root",
        database="library")
    mycursor = db.cursor()
    try:
        id = int(input("Enter the ID of the book you'd like to edit: "))
        cmnd = "select * from books where id = %s"
        id1 = (id,)
        mycursor.execute(cmnd,id1)
        p = mycursor.fetchall()
        print(p)
        if len(p) == 0:
            raise NameError
        else:
            print("Please enter NEW details of book")
            book_name = input("Book name: ")
            book_genre = input("Book genre: ")
            book_author = input("Name of the author: ")
            book_publisher = input("Name of publisher: ")
            book_count = input("Number of copies: ")
            book_summary = input("Enter new summary: ")

            book_issued = int(input("Enter number of times the book was issued: "))

            cmnd = "update books set name=%s, genre=%s, author=%s, publisher=%s, count=%s, summary = %s, issued = %s where id=%s"

            val = (book_name, book_genre, book_author, book_publisher, book_count, book_summary, book_issued, id)

            mycursor.execute(cmnd,val)

            ch1 = input("Save changes (y/n): ")

            if ch1 == 'y':

                db.commit()

                print("SUCCESS: Changes were saved\n\n")

            elif ch1 == 'n':

                print("ABORTED: Changes were not saved")

            else:

                print("ABORTED: Interpreting answer as NO. Changes were not saved.")

            ch = input("Press enter to continue...")

    except:

        if erc == '1':

            print("ERROR: Access denied.")

        else:

            print("ERROR: The record was not updated due to some error.\n"

            "-> The book ID you entered might not exist.\n\n")

        ch = input("Press enter to continue...")

    system('cls')


def allBooks():

    system('cls')

    db = mysql.connector.connect(

        host="localhost",

        user="root",

        password ="root",

        database="library")

    mycursor = db.cursor()

    mycursor.execute("select * from books")

    l1 = mycursor.fetchall()

    print("ID   Name")
    for i in l1:
        print(i[0],"  ",i[1])
    ch=input("Press enter to continue...")
    system('cls')


def bookInfo():
    system('cls')
    print("*" * 50, "BOOK INFORMATION", "*" * 50)
    db = mysql.connector.connect(
        host="localhost",
        user="root",
        password ="root",
        database="library")
    mycursor = db.cursor()
    try:
        cmnd = "select * from books where id = %s"
        id1 = input("Enter ID: ")
        system('cls')
        if id1 == '':
            raise NameError
        else:
            id = (id1,)
        mycursor.execute(cmnd, id)
        l1 = mycursor.fetchone()
        if l1[5] == 0:
            av = "not available"
            print(" Book name: ", l1[1], "\n", "Written by: ", l1[3], "\n", "Publisher: ", l1[4] , "\n", "Genre", l1[2], 
"\n", "Status: ", av, "\n", "Number of times issued: ", l1[7], "\n\n",l1[6], "\n")
        else:
            av = "available"
            print(" Book name: ", l1[1], "\n", "Written by: ", l1[3], "\n", "Publisher: ", l1[4] , "\n", "Genre", l1[2], 
"\n", "Status: ", av, "\n", "Number of copies available: ", l1[5], "\n", "Number of times issued: ", l1[7],"\n\n", 
l1[6])
    except NameError:
        print("ERROR: This input cannot be left empty.")
    except:
        print("ERROR: This book ID is not registered in the library. Please try again.")
    print('\n\n')
    ch=input("Press enter to continue...")
    system('cls')

def issueBooks():
    system('cls')
    try:
        global cardid
        global AdminLogin
        if AdminLogin == True:
            cardid = input("Please enter cardid: ")
        else:
            pass
        cmnd = "select * from users where cardid = %s"
        val = (cardid,)
        mycursor.execute(cmnd,val)
        userlist = mycursor.fetchone()
        if userlist[4] >=2:
            erc = 1
            raise MathError
        else:
            pass
        print("\n")
        id = input("Enter the book ID of the book you would like to issue: ")
        id1 = (id,)
        cmnd = "select * from books where id = %s"
        mycursor.execute(cmnd, id1)
        blist = mycursor.fetchone()
        if len(blist) == 0:
            erc = 2
            raise MathError
        if blist[5] == 0:
            print("ERROR: This book is not available in the library at this moment. Please try again later.")
        else:
```

```
cmnd = "update books set count = %s where id = %s"
val = (blist[5]-1, id)
mycursor.execute(cmnd, val)
cmnd = "update users set totalbooksissued = %s , booksinposession = %s where cardid = %s"
val = (userlist[3] + 1, userlist[4] + 1, cardid)
mycursor.execute(cmnd, val)
cmnd = "update books set issued = %s where id = %s"
val = (blist[7]+1,id)
mycursor.execute(cmnd,val)
if userlist[6] == -1:
    cmnd = "update users set book1 = %s where cardid = %s"
    val = (id, cardid)
    mycursor.execute(cmnd,val)
else:
    cmnd = "update users set book2 = %s where cardid = %s"
    val = (id, cardid)
    mycursor.execute(cmnd,val)
date_issue = datetime.date.today()
date_return = date_issue + datetime.timedelta(14)
db.commit()
print("SUCCESS: The book", blist[1], " has been issued to the holder of cardID: ", cardid,
   "\n\n")
ch1 = input("Would you like to print a reciept for your issued book? (y/n): ")
if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
    print("\n\n")
    print(" Book ID:  ", id, "\n",
       "Book name: ", blist[1], "\n",
       "Issued to ID: ", cardid, "\n",
       "Issued on: ", date_issue, "\n",
       "Return by: ", date_return, "\n", "\n")
else:
    print('Skipped reciept\n')

guidelines()
```

```
    except:
        if erc == 1:
            print("ALERT: Users may only issue 2 books at a time. Please try again after returning your current books.")
        else:
            print("ERROR: This book ID does not exist in the library.")
    ch=input('Press enter to continue...')
    system('cls')


def returnBooks():
    system('cls')
    print("*" * 50, "RETURN A BOOK", "*" * 50)
    try:
        global cardid
        global AdminLogin
        if AdminLogin == True:
            cardid = input("Please enter cardID: ")
        else:
            pass
        cmnd = "select * from users where cardid = %s"
        val = (cardid,)
        mycursor.execute(cmnd, val)
        userlist = mycursor.fetchone()
        if userlist[6] == -1 and userlist[7] == -1:
            print("ALERT: You have not issued any books currently. Please try again later.")
        elif userlist[6] == -1:
            cmnd = "select * from books where id = %s"
            val = (userlist[7],)
            mycursor.execute(cmnd, val)
            p = mycursor.fetchone()
            print("Enter 1 to return: ", p[1])
            print("Press enter if you do not want to return this book. ")
            ch1 = input("")
            if ch1 == '1':
                cmnd = "update books set count = %s where id = %s"
                val = (p[5]+1,userlist[7])
                mycursor.execute(cmnd,val)
                cmnd = "update users set booksinposession = %s where cardid = %s"
                val = (userlist[4]-1, cardid)
                mycursor.execute(cmnd, val)
                cmnd = "update users set book2 = -1 where cardid = %s"
                val = (cardid,)
                mycursor.execute(cmnd,val)
                db.commit()
                print("SUCCESS: You have successfully returned the book: ", p[1])
            else:
                print("ALERT: Skipped book return.")
                pass
        elif userlist[7] == -1:
            cmnd = "select * from books where id = %s"
            val = (userlist[6],)
            mycursor.execute(cmnd, val)
            p = mycursor.fetchone()
            print("Enter 1 to return: ", p[1])
            print("Press enter if you do not want to return this book. ")
            ch1 = input("")
            if ch1 == '1':
                cmnd = "update books set count = %s where id = %s"
                val = (p[5]+1,userlist[6])
                mycursor.execute(cmnd,val)
                cmnd = "update users set booksinposession = %s where cardid = %s"
                val = (userlist[4]-1, cardid)
                mycursor.execute(cmnd, val)
                cmnd = "update users set book1 = -1 where cardid = %s"
                val = (cardid,)
                mycursor.execute(cmnd, val)
                db.commit()
                print("SUCCESS: You have successfully returned the book: ", p[1])
            else:
                print("ALERT: Skipped book return.")
                pass
        else:
            cmnd = "select * from books where id = %s"
            val = (userlist[6],)
            mycursor.execute(cmnd, val)
            p = mycursor.fetchone()
            print("Enter 1 to return: ", p[1])
            cmnd = "select * from books where id = %s"
            val = (userlist[7],)
            mycursor.execute(cmnd, val)
            k = mycursor.fetchone()
            print("Enter 2 to return: ", k[1])
            print("Press enter if you do not want to return this book. ")
            ch1 = input("")
            if ch1 == '1':
                cmnd = "update books set count = %s where id = %s"
                val = (p[5] + 1, userlist[7])
                mycursor.execute(cmnd, val)
                cmnd = "update users set booksinposession = %s where cardid = %s"
                val = (userlist[4] - 1, cardid)
                mycursor.execute(cmnd, val)
                cmnd = "update users set book2 = -1 where cardid = %s"
                val = (cardid,)
                mycursor.execute(cmnd, val)
                db.commit()
                print("SUCCESS: You have successfully returned the book: ", p[1])
            elif ch1 == '2':
                cmnd = "update books set count = %s where id = %s"
                val = (k[5] + 1, userlist[7])
                mycursor.execute(cmnd, val)
                cmnd = "update users set booksinposession = %s where cardid = %s"
                val = (userlist[4] - 1, cardid)
                mycursor.execute(cmnd, val)
```

```
                cmnd = "update users set book2 = -1 where cardid = %s"
                val = (cardid,)
                mycursor.execute(cmnd, val)
                db.commit()
                print("SUCCESS: You have successfully returned the book: ",k[1])
            else:
                print("ALERT: Skipped book return.")
    except:
        print("ERROR: Please enter a valid book ID.\n\n")
    ch = input('Press enter to continue...')
    system('cls')


def deleteBook():
    global cardid
    global AdminLogin
    if AdminLogin == True:
        print("*" * 50, "DELETE BOOK", "*" * 50)
        print("1 Delete by ID\n2 Delete by name")
        ch1 = input("")
        if ch1 == '1':
            id = input("Enter the ID of the book you want to delete: ")
            cmnd1 = "select id, name from books where id = %s"
            val1 = (id,)
            mycursor.execute(cmnd1, val1)
            p = mycursor.fetchall()
            if len(p) == 0:
                print("ERROR: The book ID you entered does not exist. Please try again.")
            else:
                print("Book ID: ", p[0][0])
                print("Book name: ", p[0][1])
                ch1 = input("Confirm deletion? (y/n): ")
                if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                    cmnd = "delete from books where ID = %s"
                    val = (id,)
                    mycursor.execute(cmnd, val)
                    db.commit()
                    print("SUCCESS: The book with ID: ", id, " has been deleted.\n\n")
                elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':
                    print("ABORTED: Deletion of the book was cancelled.\n\n ")
                else:
                    print("ABORTED: Interpreting vague answer as NO. Deleteion was cancelled.\n\n")
        elif ch1 == '2':
            id = input("Enter the NAME of the book you want to delete: ")
            cmnd1 = "select id, name from books where name = %s"
            val1 = (id,)
            mycursor.execute(cmnd1, val1)
            p = mycursor.fetchall()
            if len(p) == 0:
                print("ERROR: The book name you entered does not exist. Please try again.")
            else:
                print("Book ID: ", p[0][0])
                print("Book name: ", p[0][1])
                ch1 = input("Confirm deletion? (y/n): ")
                if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                    cmnd = "delete from books where ID = %s"
                    val = (id,)
                    mycursor.execute(cmnd, val)
                    db.commit()
                    print("SUCCESS: The book with ID: ", id, " has been deleted.\n\n")
                elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':
                    print("ABORTED: Deletion of the book was cancelled.\n\n ")
                else:
                    print("ABORTED: Interpreting vague answer as NO. Deleteion was cancelled.\n\n")
        else:
            print("ERROR: Please enter a valid option. (1/2)")
        ch = input('Press enter to continue...')
    else:
        print("ERROR: Access denied.")
        ch1 = input("Press any key to continue...")
    system('cls')
```

```
def registerUser():
    system('cls')
    while True:
        try:
            print("*" * 50, "REGISTER NEW USERS", "*" * 50)
            print("Please enter the details of the user\n")

            while True:
                cardID = random.randint(100, 999)
                cmnd2 = "select * from users where cardID = %s"
                val2 = (cardID,)
                mycursor.execute(cmnd2, val2)
                k = mycursor.fetchall()
                if len(k) == 0:
                    break
                else:
                    cardID = random.randint(100, 999)
            print(cardID)
            username = str(input("Enter username: "))
            password = str(input("Enter Password: "))
            cpassword = str(input("Confirm Password: "))
            if password == '':
                raise NameError
            elif password == cpassword:
                print("\n Passwords match. Please proceed")
                if len(password) <= 5:
                    print("Password strength: Low\n")
                elif len(password) <=8:
                    print("Password strength: Medium\n")
                else:
                    print("Password strength: High\n")
                date_issue = datetime.date.today()
                cmnd = "INSERT into users values (%s,%s,%s,0,0,%s,-1,-1)"
                val = (cardID, username, date_issue, password)
                mycursor.execute(cmnd, val)
                ch1 = input("Confirm entry? (y/n): ")
                if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                    db.commit()
                    print("SUCCESS: The new user was registered.")
                    ch1 = input("Would you like to print a reciept? (y/n): ")
                    if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                        print("\n\n")
                        print(" Card ID: ", cardID, "\n",
                              "Password: ", password, "\n",
                              "Username: ", username, "\n",
                              "Issued on: ", date_issue, "\n")
                    else:
                        print('Skipped reciept')
                elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':
                    print("ABORTED: Registration cancelled.\n")
                else:
                    print("Interpreting input as 'no'. Registration cancelled.\n")
            else:
                print("\nPasswords do not match. Please try again.\n")
            print("\n")


        except:
            print(
                "ERROR: You have entered an invalid input for one of the fields. Please check your input and try again.\n"
                "-> Username and Password cannot be left empty\n"
                "-> The string inputs may not exceed 40 characters.\n\n")
        ch = input("Would you like to register another user? (y/n): ")
        ("\n\n")
        if ch == 'y':
            system('cls')
            continue
        elif ch == 'n':
            break
```

```
         else:
             print("Interpretting vague input as NO.")
             break
     ch2 = input("Press enter to return to main menu...")
     system('cls')
def editUser():
 system('cls')
 global AdminLogin
 global cardid
 try:
     print("*" * 50, "EDIT USER INFO", "*" * 50)
     if AdminLogin == True:
         cardid = int(input("Enter cardID: "))
     else:
         pass
     print("\n\n")
     cmnd = "select * from users where cardid = %s"
     val = (cardid,)
     mycursor.execute(cmnd, val)
     k = mycursor.fetchall()
     if len(k) == 0:
         print("ERROR: This user is not registered with us in the library.")
     else:
         print("Card ID: ", cardid)
         print("Username: ", k[0][1])
         print("Current Password: ", k[0][5])
         print("Date of Issue: ", k[0][2])
         print("Total books issued: ", k[0][3])
         print("Books currently in posession: ", k[0][4])
         print("*"*100)
         print("Please enter NEW details of the user\n")
         username = input("Enter new username: ")
         npass = input("Enter new password: ")
         if AdminLogin == True:
             tbooks = int(input("Total books issued: "))
```

```
            posession = int(input("Books in posession: "))

            cmnd2 = "update users set username = %s, totalbooksissued = %s, booksinposession = %s, password
= %s where cardid = %s"

            val2 = (username,tbooks,posession,npass,cardid)

            mycursor.execute(cmnd2,val2)

        else:

            cmnd2 = "update users set username = %s, password = %s where cardid = %s"

            val2 = (username, npass, cardid)

            mycursor.execute(cmnd2, val2)

        print("\n\n")

        ch1 = input("Confirm edit? (y/n): ")

        if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':

            db.commit()

            print("SUCCESS: The new records were edited in the database.")

        elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':

            print("ABORTED: This record was not edited into the database.")

        else:

            print("Interpreting vague input as 'no'. This record was not edited into database.")

        ch = input("Press Enter to continue...")

 except MathError:

     print("ERROR: Wrong cardID and/or password combination. Please try again.")

 except:

     print("ERROR: Something went wrong. Please check your inputs and try again.")

     print("-> Card ID must be correctly input")

     print("-> Total books issued and books in posession should be integers")


 system('cls')


def allUser():

    system('cls')

    print("*" * 50, "ALL USERS", "*" * 50)

    mycursor.execute("select * from users")

    l1 = mycursor.fetchall()

    if AdminLogin == True:

        print("CardID   Username    Password\n")
```

```
         for i in l1:
             print(i[0], "  ", i[1], "    ", i[5])
     if AdminLogin == False:
         print("CardID   Username")
         for i in l1:
             print(i[0], "  ", i[1])
     ch = input("Press enter to continue...")
     system('cls')


def infoUser():
 global AdminLogin
 global cardid
 system('cls')
 try:
     if AdminLogin == True:
         cardid = input("Enter cardid: ")
     else:
         pass
     print("*"*50,"USER INFO",'*'*50)
     print("\n")
     cmnd = "select * from users where cardid = %s"
     val = (cardid,)
     mycursor.execute(cmnd, val)
     k = mycursor.fetchall()
     if len(k) == 0:
         print("ERROR: This user is not registered with us in the library.")
     else:
         print("Card ID: ", cardid)
         print("Username: ", k[0][1])
         print("Current Password: ", k[0][5])
         print("Date of Issue: ", k[0][2])
         print("Total books issued: ", k[0][3])
         print("Books currently in posession: ", k[0][4])


 except:
      print("ERROR: CardID is necessary to display user info. Please enter a valid card ID")
 ch = input("Press Enter to continue...")
 system('cls')

def deleteUser():
     global AdminLogin
     if AdminLogin == True:
         system('cls')
         print("*" * 50, "DELETE USER", "*" * 50)
         id = input("Enter the cardID of the user you want to delete: ")
         cmnd1 = "select cardid, username from users where cardid = %s"
         val1 = (id,)
         mycursor.execute(cmnd1, val1)
         p = mycursor.fetchall()
         if len(p) == 0:
             print("ERROR: The cardID you entered does not exist. Please try again.")
         else:
             print("Card ID: ", p[0][0])
             print("Username: ", p[0][1])
             ch1 = input("Confirm deletion? (y/n): ")
             if ch1 == 'y' or ch1 == 'Y' or ch1 == 'YES' or ch1 == 'yes':
                 cmnd = "delete from users where cardid = %s"
                 val = (id,)
                 mycursor.execute(cmnd, val)
                 db.commit()
                 print("SUCCESS: The user with ID: ", id, " has been deleted.\n\n")
             elif ch1 == 'n' or ch1 == 'N' or ch1 == 'NO' or ch1 == 'no':
                 print("ABORTED: Deletion of the user was cancelled.\n\n ")
             else:
                 print("ABORTED: Interpreting vague answer as NO. Deleteion was cancelled.\n\n")
         ch = input("Press Enter to continue...")
         system('cls')
     else:
         print("ERROR: Access denied.")
         ch1 = input("Press enter to continue...")
```

```
#Menu

def Login():
 global p
 erc = 0
 try:
     global AdminLogin
     global loginfo
     print(

"
██╗░░░░░██╗██████╗░██████╗░░█████╗░██████╗░██╗██╗░░░██╗███╗░░░███
╗\n"
"
██║░░░░░██║██╔══██╗██╔══██╗██╔══██╗██╔══██╗██║██║░░░██║████╗░████
║\n"
"
██║░░░░░██║██████╦╝██████╔╝███████║██████╔╝██║██║░░░██║██╔████╔██
║\n"
"
██║░░░░░██║██╔══██╗██╔══██╗██╔══██║██╔══██╗██║██║░░░██║██║╚██╔╝██
║\n"
"
███████╗██║██████╦╝██║░░██║██║░░██║██║░░██║██║╚██████╔╝██║░╚═╝░██
║\n"
"
╚══════╝╚═╝╚═════╝░╚═╝░░╚═╝╚═╝░░╚═╝╚═╝░░╚═╝╚═╝░╚═════╝░╚═╝░░░░░╚═
╝\n\n")
     print("                              Welcome user. Please login or sign up to continue \n\n")
     print("                                    1. PRESS 1 TO LOG IN AS ADMINISTRATOR")
     print('                                    2. PRESS 2 TO LOG IN AS A USER      ')
     print("                                    3. PRESS 3 TO VIEW RULES, TERMS AND CONDITIONS")
     print("                                    4. PRESS 4 TO REGISTER AS A NEW USER.")
     print("                                    5. PRESS 5 TO EXIT.")
     ch = input("                                  ENTER HERE: ")
     system('cls')
     if ch == '1':
         system('cls')
         password = input("Enter administrator password: ")
```

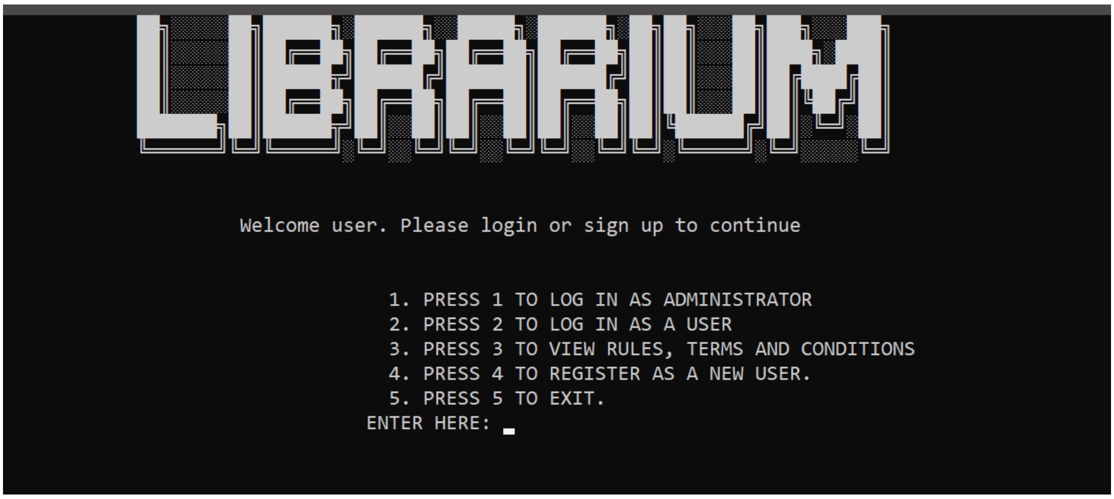
```
    if password == 'abcd':
        print("Fetching data")
        for i in range(5):
            time.sleep(0.3)
            print(".",end='')
        print("\n\n")
        print("Welcome Administrator. Please proceed.\n")


        ch1 = input("Press enter to continue...")
        AdminLogin = True
        loginfo = True
    else:
        print("The password you entered was incorrect. Please return to homescreen and try again.")
        ch1 = input("Press enter to continue...")
elif ch == '2':
    system('cls')
    global cardid
    cardid = input("Please enter your card ID: ")
    password = input("Please enter your password: ")
    cmnd = "select * from users where cardid = %s"
    val = (cardid,)
    mycursor.execute(cmnd, val)
    k = mycursor.fetchone()
    if len(k) == 0:
        erc = 1
        raise MathError
    if k[5] == password:
        print("Welcome, ", k[1], ". Please proceed.\n")
        loginfo = True
        ch1=input("Press enter to continue...")
    else:
        erc=1
        raise MathError
elif ch == '3':
```

```
            print("Hello user, welcome to Librarium.\n"
                  "We at Librarium believe that book serve as a vital source of information and guidance for anyone regardless of age. We believe that a pleasant environment and a good book reading session can have a \n"
                  "massively positive impact on one's life and therefore we strive to present our customers with the same. \n")
            print("Librarium has many features that make it the perfect place to spend time working on yourself or your studies. Some of the sailent features of the library are:\n"
                  "-> 24/7 open reading space for registered customers\n"
                  "-> Over 2,000 books from a wide variety of generes to choose from\n"
                  "-> Coffee and tea served in reading spaces\n"
                  "-> Fully A/C facility\n"
                  "-> Sound proof walls for maximum focus\n")
            guidelines()
            ch = input("Press enter to continue...")
        elif ch == '4':
            registerUser()
        elif ch == '5':
            p = 1
 except:
        print("ERROR: Wrong ID or password. Please try again.")
        ch1 = input("Press enter to return to homescreen...")


 system('cls')


def Menu(choice):
 global loginfo
 global AdminLogin
 system('cls')
 try:
        if choice == 1:
         if AdminLogin == True:
            print("*" * 30, "LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia", "*" * 30)
            print("\n****************** MANAGE BOOKS ******************\n")
            print("1. Enter new books")
            print("2. Edit book information")
            print("3. Delete a book")
            ch1 = int(input(""))
            if ch1 == 1:
                bookWrite()
            elif ch1 == 2:
                bookEdit()
            elif ch1 == 3:
                deleteBook()
            else:
                raise NameError
         else:
            print("Access Denied.")
            ch1 = input("Press enter to return to main menu...")
            system('cls')
        elif choice ==2:
            print("*" * 30, "LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia", "*" * 30)
            print("\n******************* BOOK INFORMATION *******************\n")
            print("1. List of all books")
            print("2. Check full details of a book")
            ch1 = int(input(""))
            if ch1 == 1:
                allBooks()
            elif ch1 == 2:
                bookInfo()
            else:
                raise NameError
        elif choice == 3:
            print("*" * 30, "LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia", "*" * 30)
            print("\n******************* ISSUE / RETURN BOOK *******************\n")
            print("1. Issue a book")
            print("2. Return a book")
            ch1 = int(input(""))
            if ch1 == 1:
                issueBooks()
            elif ch1 == 2:
                returnBooks()
            else:
                raise NameError
        elif choice == 4:
            print("*" * 30, "LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia", "*" * 30)
            print("\n******************* MANAGE USERS *******************\n")
            print("1. Register users")
            print("2. Edit user details")
            print("3. Details of user")
            print("4. List of all users")
            print("5. Delete users")
            ch1 = int(input(""))
            if ch1 == 1:
                registerUser()
            elif ch1 ==2:
                editUser()
            elif ch1 == 3:
                infoUser()
            elif ch1 == 4:
             if AdminLogin == True:
                allUser()
             else:
                print("Access denied.")
                ch1 = input("Press enter to return to main menu")
                system('cls')
            elif ch1 ==5:
                if AdminLogin == True:
                    deleteUser()
                else:
                    print("Access denied.")
                    ch1 = input("Press enter to return to main menu")
                    system('cls')
            else:
                raise NameError
        elif choice == 5:
            loginfo = False
        else:
            print("ERROR: Please enter a valid number to perform an operation.")
            ch = input("Press Enter to continue...")
            system('cls')
     except:
         print("Please enter a valid input according to the given options.")
         ch = input("Press Enter to continue...")
         system('cls')


    def mainmenu():
        global loginfo
        global AdminLogin
        print("*" * 30, "LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia", "*" * 30)
        print("Please enter the serial number of the function you would like to perform.")
        print("1. Manage books")
        print("2. Book Information")
        print("3. Issue / Return a book")
        print("4. Manage users")
        print("5. Log out and return to homescreen")
        try:
            x = int(input(""))
            Menu(x)
        except:
            print("ERROR: Please enter a valid number to perform an operation.")
            ch=input("Press Enter to continue...")
            system('cls')


    p = 0
    while True:
        AdminLogin = False
        loginfo = False
        Login()
        while loginfo == True:
            mainmenu()
        if p == 1:
            break
        else:
            pass
```

TOTAL LINES – 823

OUTPUT SCREENSHOTS

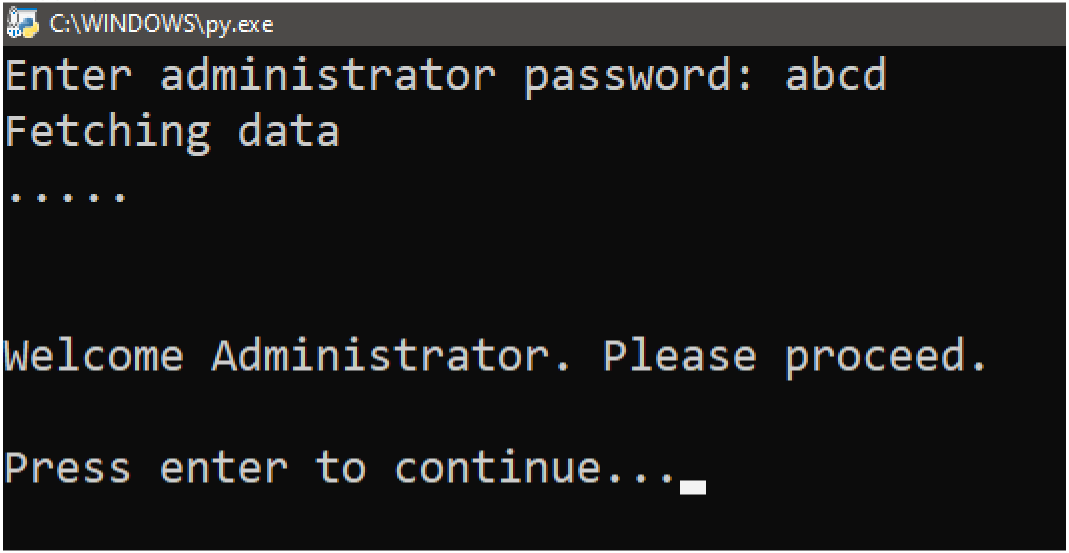
Part 1: The log-in screen



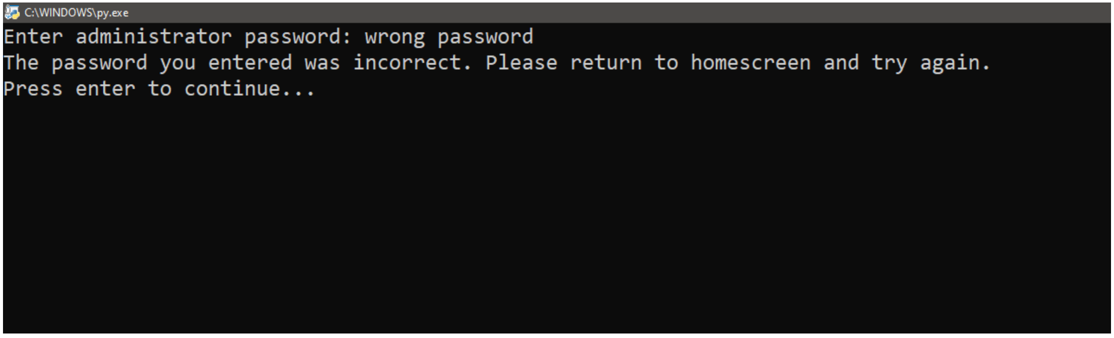
```
              Welcome user. Please login or sign up to continue

                         1. PRESS 1 TO LOG IN AS ADMINISTRATOR
                         2. PRESS 2 TO LOG IN AS A USER
                         3. PRESS 3 TO VIEW RULES, TERMS AND CONDITIONS
                         4. PRESS 4 TO REGISTER AS A NEW USER.
                         5. PRESS 5 TO EXIT.
                       ENTER HERE:
```

This is the home-screen which also serves as the outer main loop for the program

1. Logging in as administrator

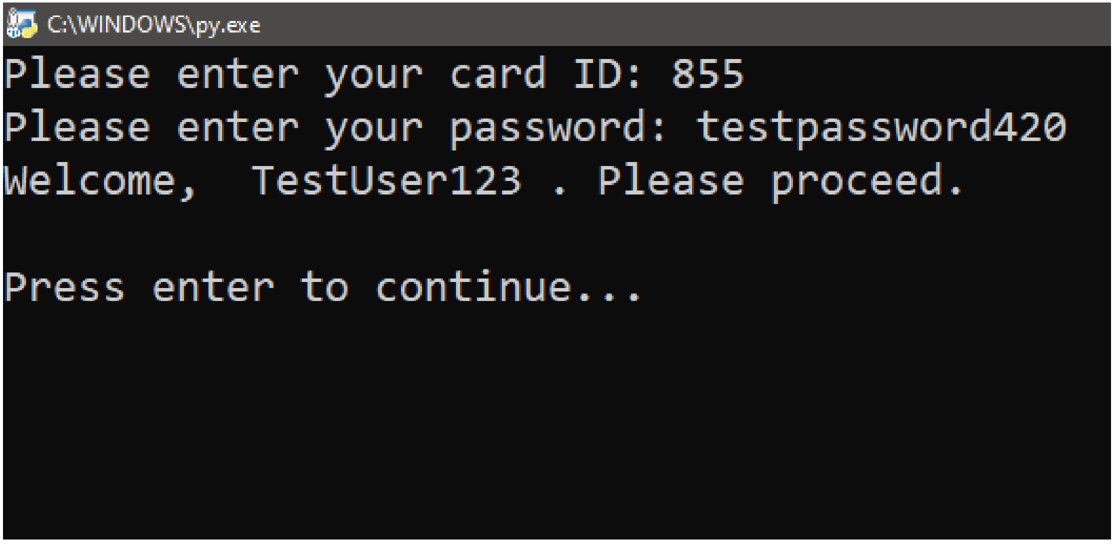


```
Enter administrator password: abcd
Fetching data
.....


Welcome Administrator. Please proceed.

Press enter to continue...
```

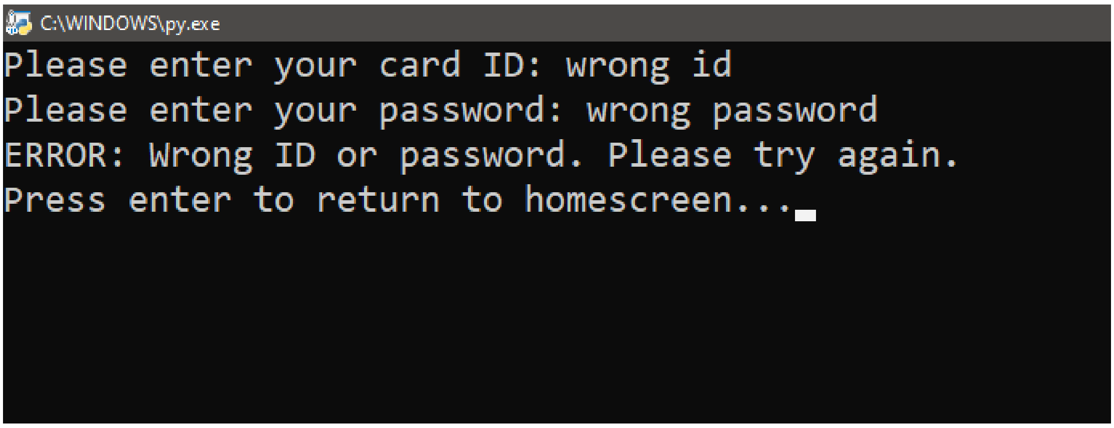


```
Enter administrator password: wrong password
The password you entered was incorrect. Please return to homescreen and try again.
Press enter to continue...
```

2. Logging in as a user



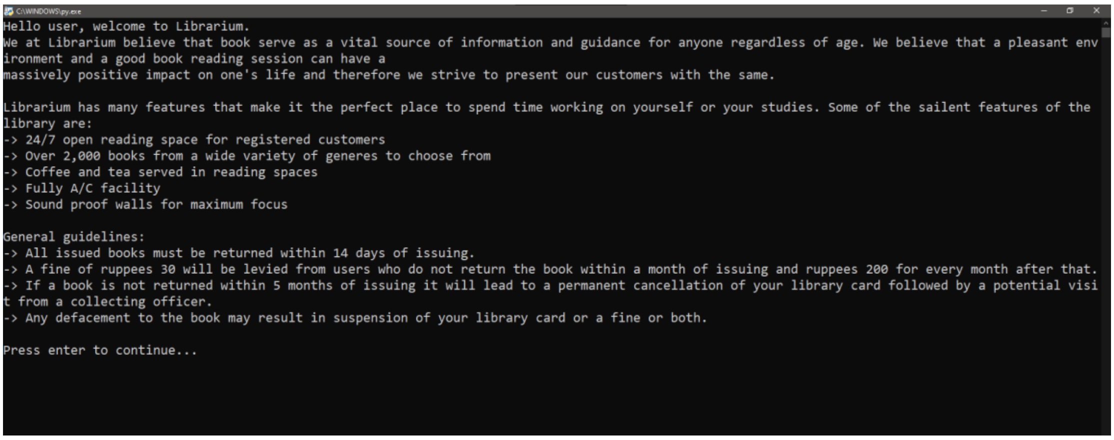
```
Please enter your card ID: 855
Please enter your password: testpassword420
Welcome,  TestUser123 . Please proceed.

Press enter to continue...
```



```
Please enter your card ID: wrong id
Please enter your password: wrong password
ERROR: Wrong ID or password. Please try again.
Press enter to return to homescreen...
```

Shows what went wrong at all steps to make the program user-friendly

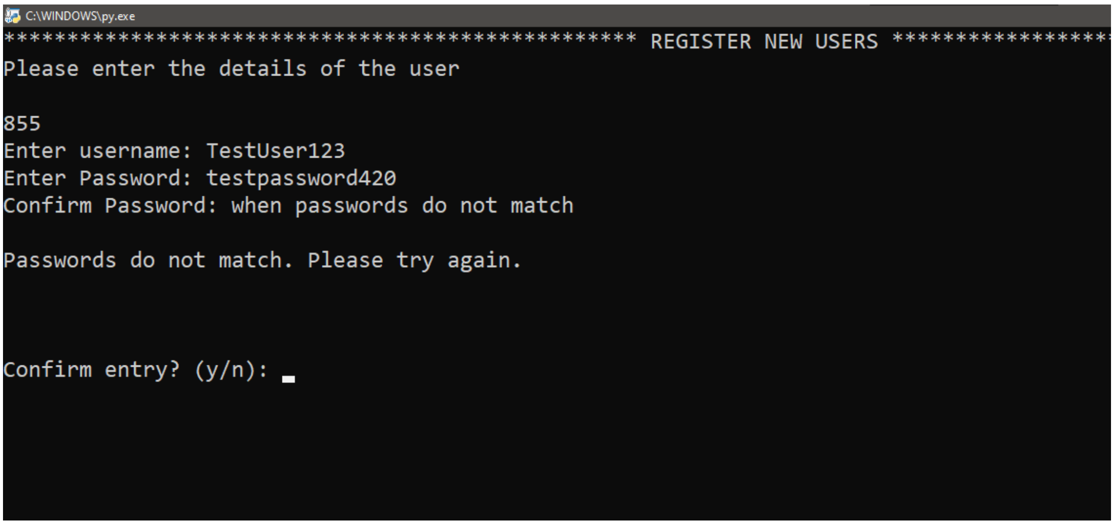
3. Rules, terms and conditions



```
Hello user, welcome to Librarium.
We at Librarium believe that book serve as a vital source of information and guidance for anyone regardless of age. We believe that a pleasant env
ironment and a good book reading session can have a
massively positive impact on one's life and therefore we strive to present our customers with the same.

Librarium has many features that make it the perfect place to spend time working on yourself or your studies. Some of the sailent features of the
library are:
-> 24/7 open reading space for registered customers
-> Over 2,000 books from a wide variety of generes to choose from
-> Coffee and tea served in reading spaces
-> Fully A/C facility
-> Sound proof walls for maximum focus

General guidelines:
-> All issued books must be returned within 14 days of issuing.
-> A fine of ruppees 30 will be levied from users who do not return the book within a month of issuing and ruppees 200 for every month after that.
-> If a book is not returned within 5 months of issuing it will lead to a permanent cancellation of your library card followed by a potential visi
t from a collecting officer.
-> Any defacement to the book may result in suspension of your library card or a fine or both.

Press enter to continue...
```

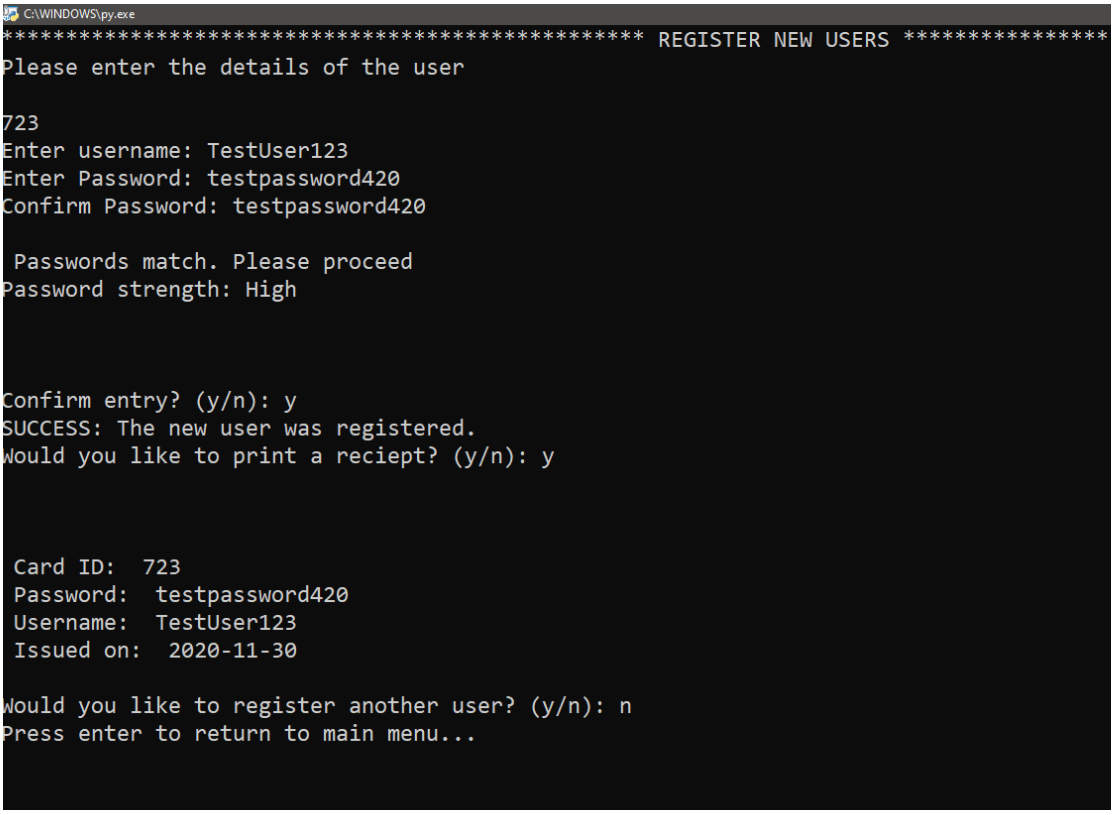
Standard set of rules and regulations for the library

4. Register new users (This is also a feature in the main menu once the user has logged in however they both server the exact same function)



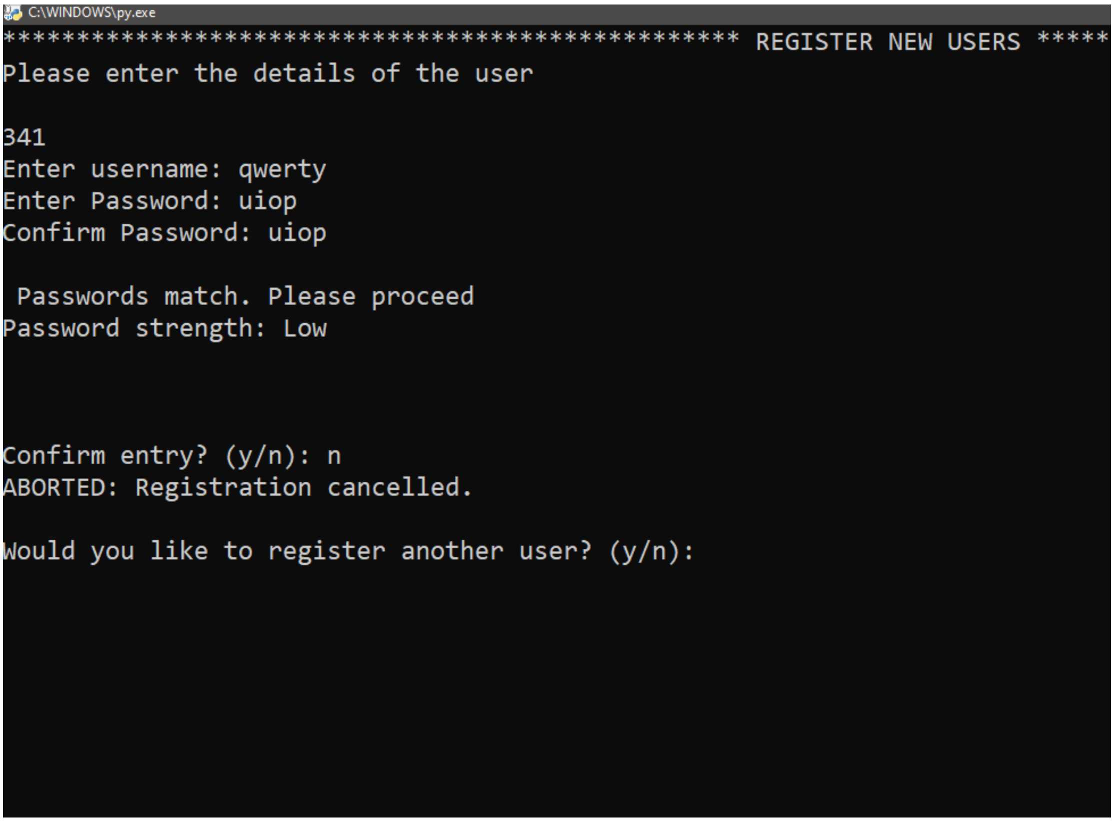
```
************************************************** REGISTER NEW USERS *****************
Please enter the details of the user

855
Enter username: TestUser123
Enter Password: testpassword420
Confirm Password: when passwords do not match

Passwords do not match. Please try again.



Confirm entry? (y/n): _
```

The password conformation system also implemented



```
************************************************** REGISTER NEW USERS ****************
Please enter the details of the user

723
Enter username: TestUser123
Enter Password: testpassword420
Confirm Password: testpassword420

 Passwords match. Please proceed
Password strength: High



Confirm entry? (y/n): y
SUCCESS: The new user was registered.
Would you like to print a reciept? (y/n): y



 Card ID:  723
 Password:  testpassword420
 Username:  TestUser123
 Issued on:  2020-11-30

Would you like to register another user? (y/n): n
Press enter to return to main menu...
```

The system allows for checking of password strength along with an e-receipt. You can also register another user while in the menu for the sake of convenience.

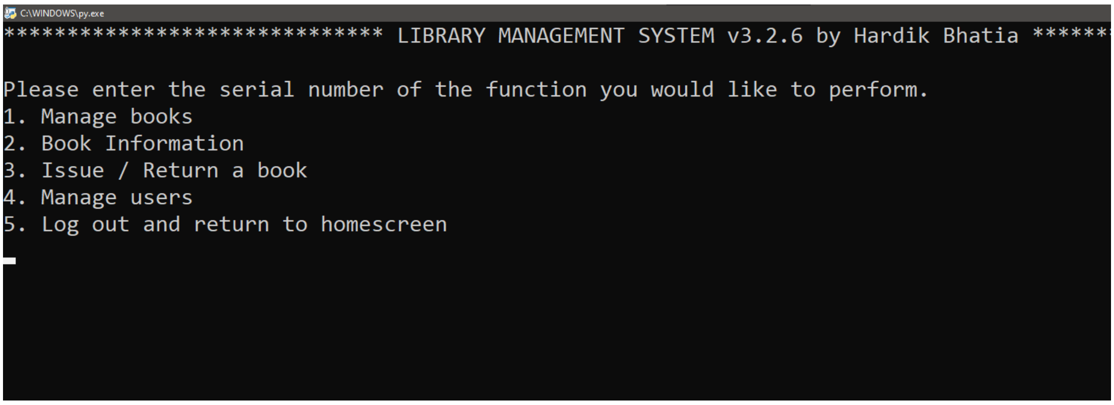


```
************************************************** REGISTER NEW USERS *****
Please enter the details of the user

341
Enter username: qwerty
Enter Password: uiop
Confirm Password: uiop

 Passwords match. Please proceed
Password strength: Low



Confirm entry? (y/n): n
ABORTED: Registration cancelled.

Would you like to register another user? (y/n):
```

Negative input for entry conformation: user is not registered and the inputs are discarded.

Part 2: The Main menu (User)

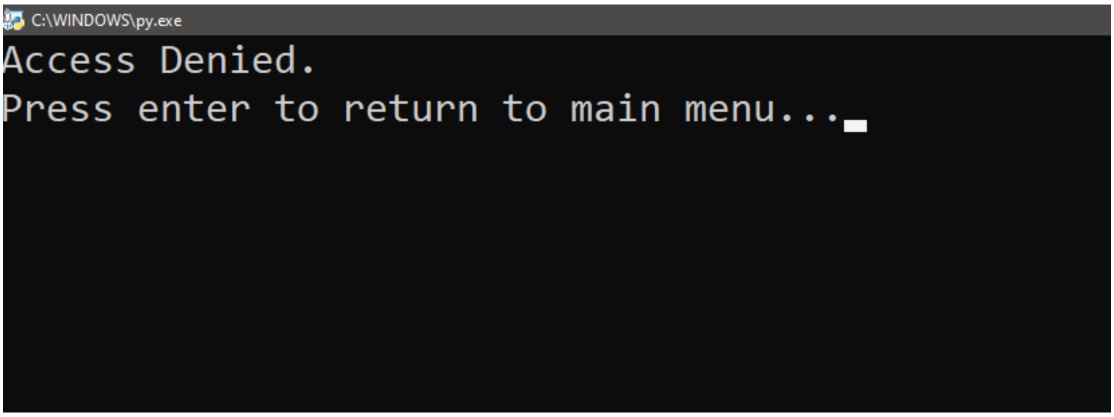
The main menu for the user is a rather simple set of available operations to automate jobs like issuing and returning of books. Of course, the user may only perform operations for his/her self and to avoid security risks the card ID of the user is automatically declared as a global variable throughout the main menu (as long as the user logs out) and is automatically assumed each time he/she wishes to do an operation



```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ******

Please enter the serial number of the function you would like to perform.
1. Manage books
2. Book Information
3. Issue / Return a book
4. Manage users
5. Log out and return to homescreen
```

1. Manage books

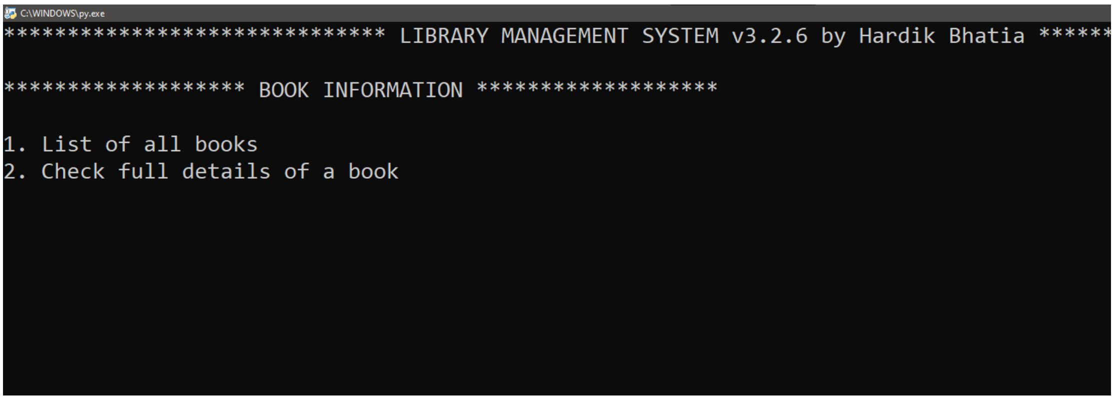
Now of course we don’t want users to interact with the database of books directly causing undesirable changes, to prevent this, in user-mode the manage books option is disabled entirely



```
C:\WINDOWS\py.exe
Access Denied.
Press enter to return to main menu...
```

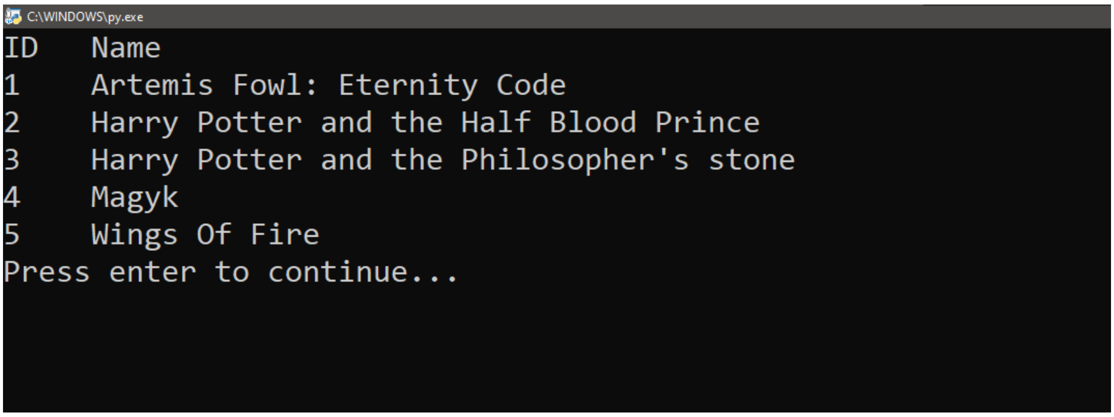
2. Book Information

Book information on the other hand is a necessary tool for any user to find the correct choice of book to issue.



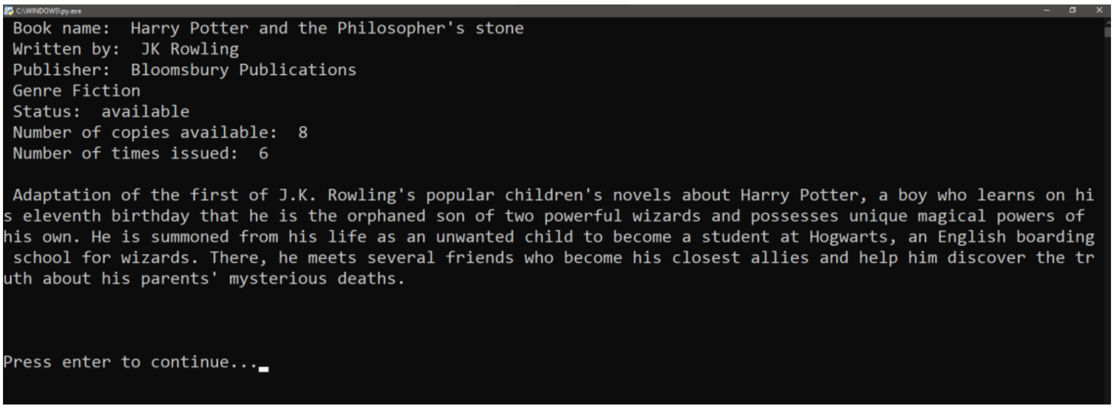
```
C:\WINDOWS\py.exe
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ******

****************** BOOK INFORMATION ******************

1. List of all books
2. Check full details of a book
```

a. List of all books



```
C:\WINDOWS\py.exe
ID    Name
1     Artemis Fowl: Eternity Code
2     Harry Potter and the Half Blood Prince
3     Harry Potter and the Philosopher's stone
4     Magyk
5     Wings Of Fire
Press enter to continue...
```

b. Check full details of a book

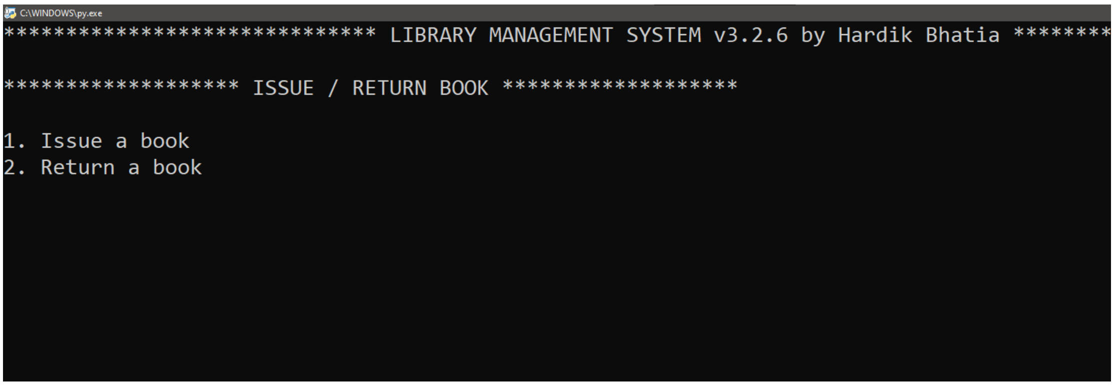


```
 Book name:  Harry Potter and the Philosopher's stone
 Written by:  JK Rowling
 Publisher:  Bloomsbury Publications
 Genre Fiction
 Status:  available
 Number of copies available:  8
 Number of times issued:  6

 Adaptation of the first of J.K. Rowling's popular children's novels about Harry Potter, a boy who learns on hi
s eleventh birthday that he is the orphaned son of two powerful wizards and possesses unique magical powers of
his own. He is summoned from his life as an unwanted child to become a student at Hogwarts, an English boarding
 school for wizards. There, he meets several friends who become his closest allies and help him discover the tr
uth about his parents' mysterious deaths.


Press enter to continue...
```

3. Issue / Return books

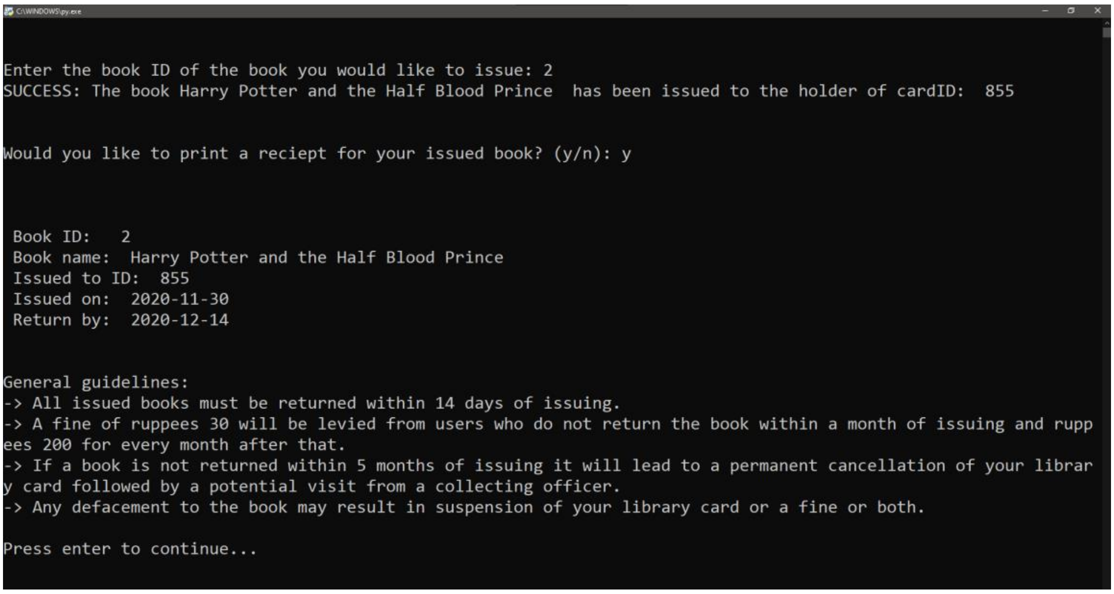
Working:
The issue / return system works by assigning values to the columns of each book. The count feature tells you how many copies of a book are available in the library currently. When someone issues a book it reduces the count by one and upon return the count is increased by one. Similarly, whenever someone issues a book the book ID column in the users table which tells the program later what book was issued is checked for book IDs or empty slots and since the book can only be returned by the user using a defined system that shows you what books you issued, the count remains unchanged. **It's basically fool-proof**. Upon return a book, your book ID column is assigned -1 value which, by my methods, represents an empty slot.

This is one of the most important functions of the whole program which captures the essence of the entire project.



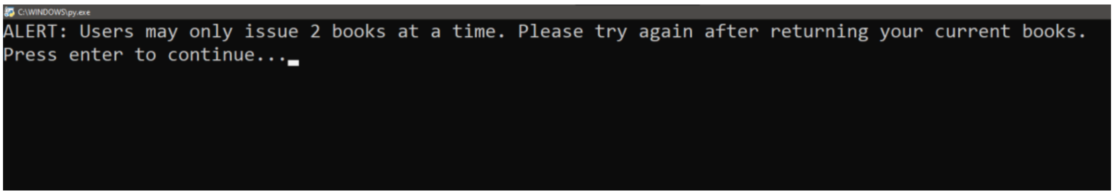
```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ********

****************** ISSUE / RETURN BOOK *******************

1. Issue a book
2. Return a book
```

1. Issuing a book



```
Enter the book ID of the book you would like to issue: 2
SUCCESS: The book Harry Potter and the Half Blood Prince  has been issued to the holder of cardID:  855


Would you like to print a reciept for your issued book? (y/n): y



 Book ID:   2
 Book name:  Harry Potter and the Half Blood Prince
 Issued to ID:  855
 Issued on:  2020-11-30
 Return by:  2020-12-14


General guidelines:
-> All issued books must be returned within 14 days of issuing.
-> A fine of ruppees 30 will be levied from users who do not return the book within a month of issuing and rupp
ees 200 for every month after that.
-> If a book is not returned within 5 months of issuing it will lead to a permanent cancellation of your librar
y card followed by a potential visit from a collecting officer.
-> Any defacement to the book may result in suspension of your library card or a fine or both.

Press enter to continue...
```

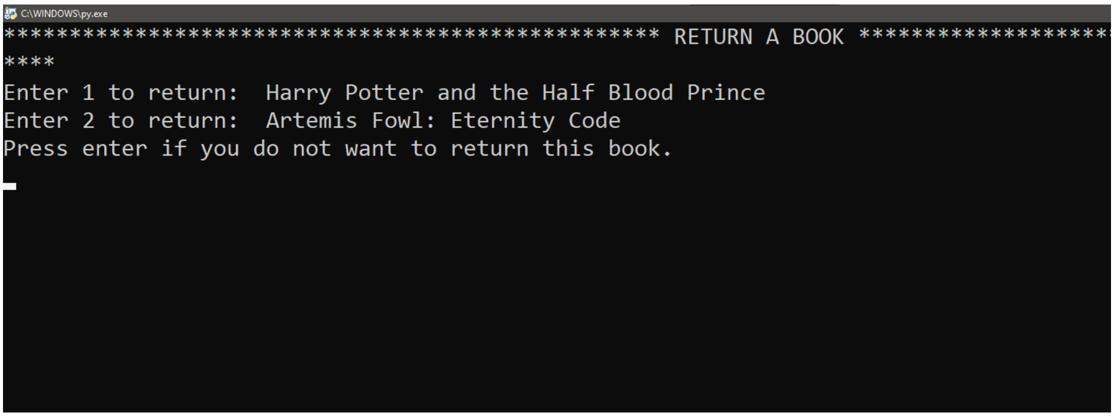
When you issue a book the program writes the book IDs in 2 specific columns in each row. Using this, we can check for the number of books one user can issue



```
ALERT: Users may only issue 2 books at a time. Please try again after returning your current books.
Press enter to continue...
```

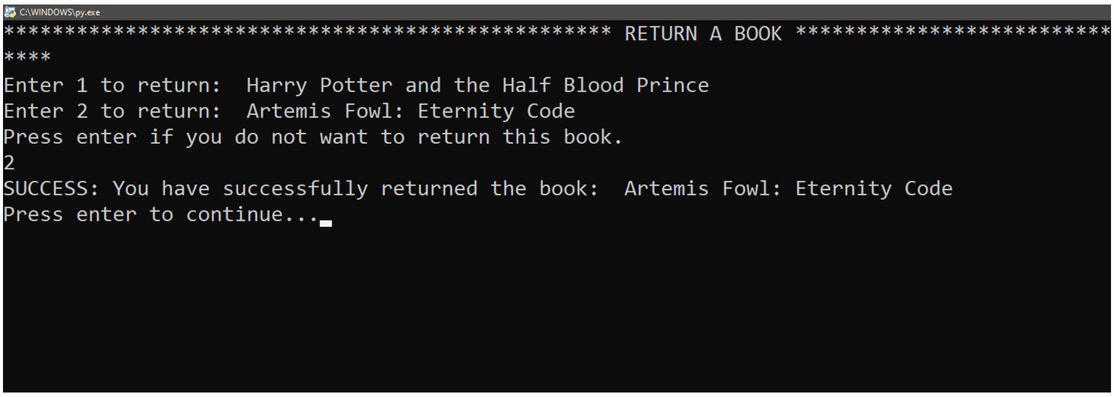
If you try to issue another book with both your book ID issued columns already occupied, you get this message

2. Return a book



```
************************************************** RETURN A BOOK ********************
****
Enter 1 to return:  Harry Potter and the Half Blood Prince
Enter 2 to return:  Artemis Fowl: Eternity Code
Press enter if you do not want to return this book.
```

The program takes input directly from the table to know which books you issued and allows for the return of only those books.

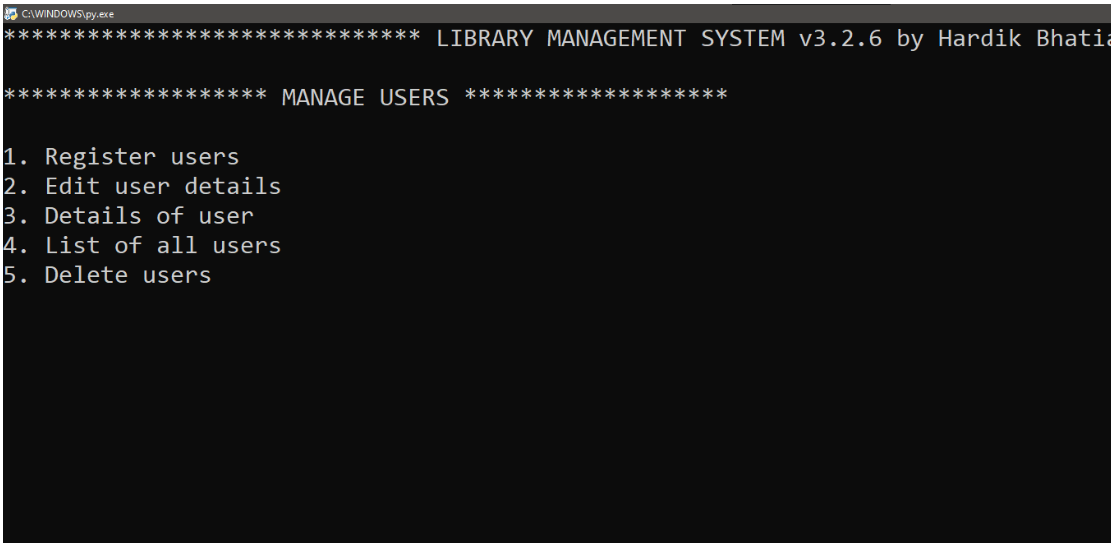


```
************************************************** RETURN A BOOK **************************
****
Enter 1 to return:  Harry Potter and the Half Blood Prince
Enter 2 to return:  Artemis Fowl: Eternity Code
Press enter if you do not want to return this book.
2
SUCCESS: You have successfully returned the book:  Artemis Fowl: Eternity Code
Press enter to continue...
```

And just like that your book was returned to the library.

4. Manage users

Working: This function is there to inform the users about their own data. In case of administrator account, it can show you accounts of all the users. It can be used to edit the information of the user which, again, the admin may edit for anyone yet the user may only edit for themselves (however some details for even the logged in users may only be edited by the admin like the number of books in possession in a real case scenario where something might go wrong.)



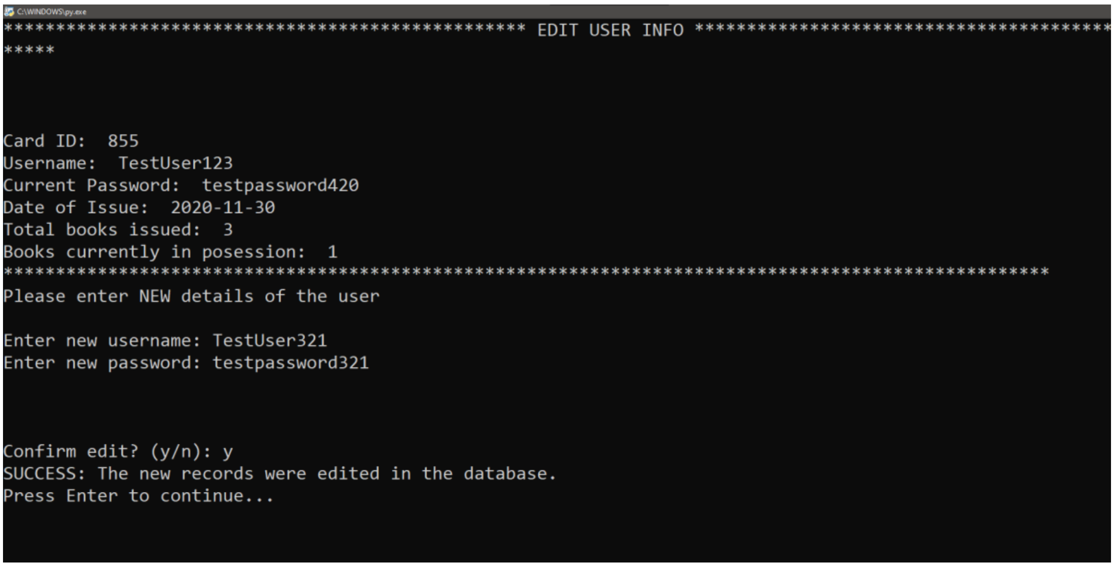
```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhati

******************* MANAGE USERS *******************

1. Register users
2. Edit user details
3. Details of user
4. List of all users
5. Delete users
```

1. Register users

As shown before this function can be used to register new users in the database.

2. Edit user details

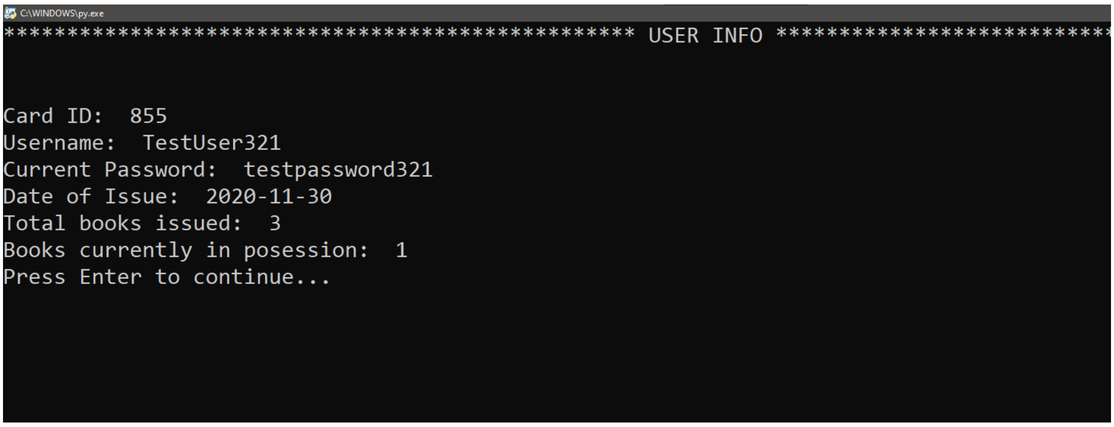
This function has key differences for the user and admin as described above.



```
************************************************** EDIT USER INFO ****************************************
*****



Card ID:  855
Username:  TestUser123
Current Password:  testpassword420
Date of Issue:  2020-11-30
Total books issued:  3
Books currently in posession:  1
****************************************************************************************************
Please enter NEW details of the user

Enter new username: TestUser321
Enter new password: testpassword321



Confirm edit? (y/n): y
SUCCESS: The new records were edited in the database.
Press Enter to continue...
```

3. Details of user

This shows you the details of the user. Pretty basic yet necessary function.



```
************************************************** USER INFO ***************************


Card ID:  855
Username:  TestUser321
Current Password:  testpassword321
Date of Issue:  2020-11-30
Total books issued:  3
Books currently in posession:  1
Press Enter to continue...
```

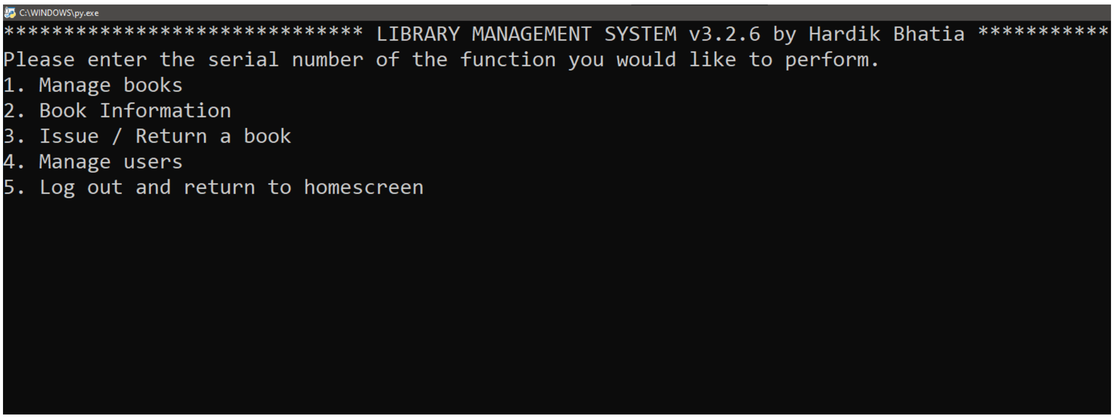
4. List of all users

This function is disabled in the user-mode but will be covered later in admin mode

5. Delete users

This function too is disabled for users.

Part 3: The Main Menu (Administrator)

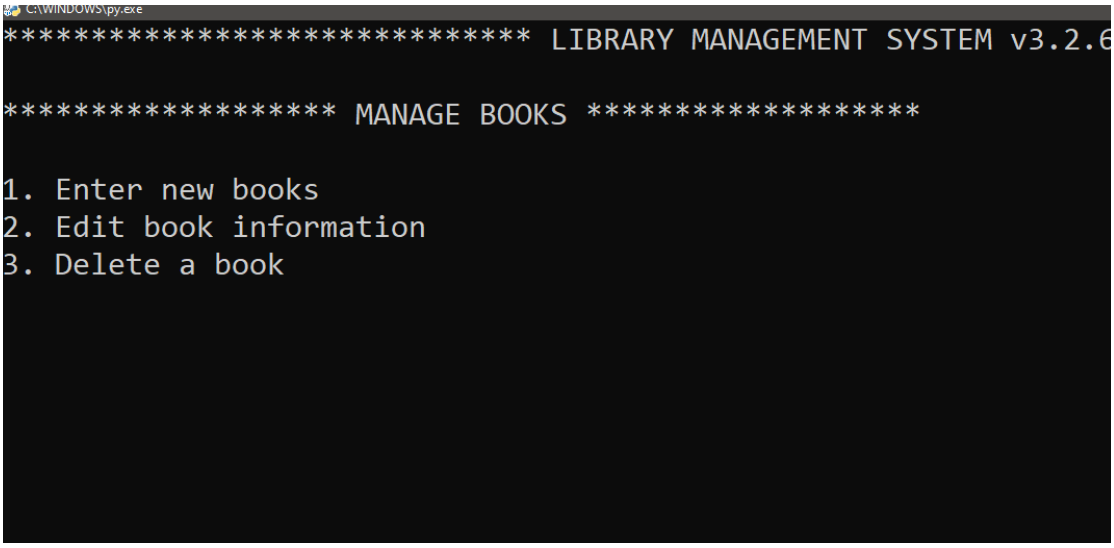


```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ***********
Please enter the serial number of the function you would like to perform.
1. Manage books
2. Book Information
3. Issue / Return a book
4. Manage users
5. Log out and return to homescreen
```

The main menu is essentially the same for administrator

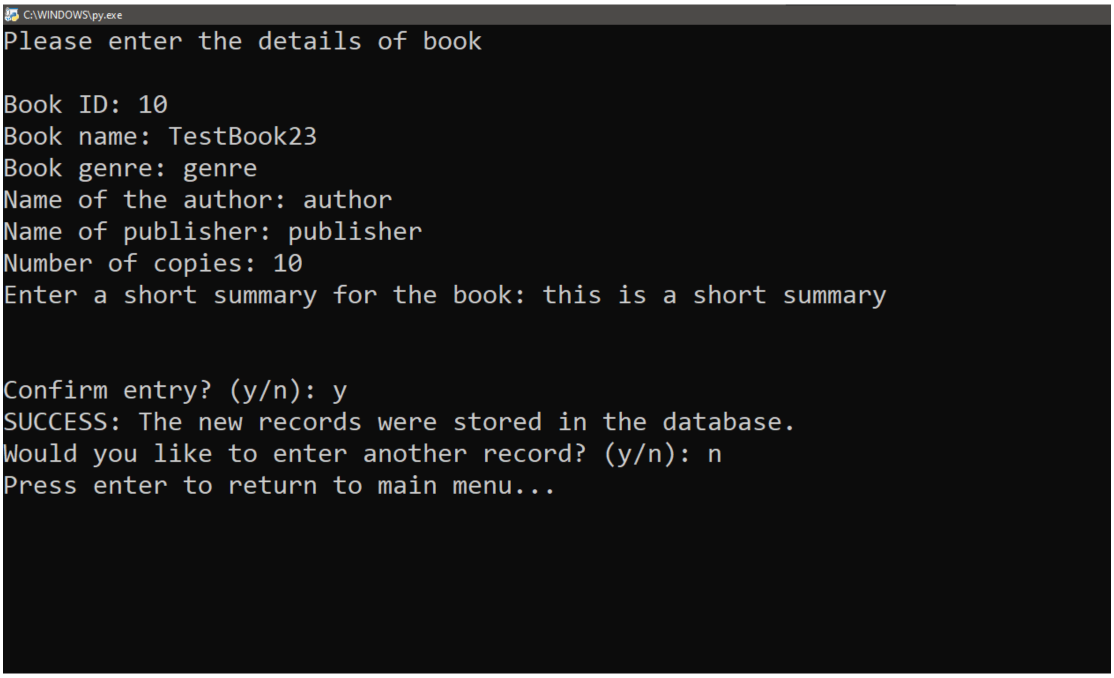
1. Manage books

This function was disabled for users but is open for administrators



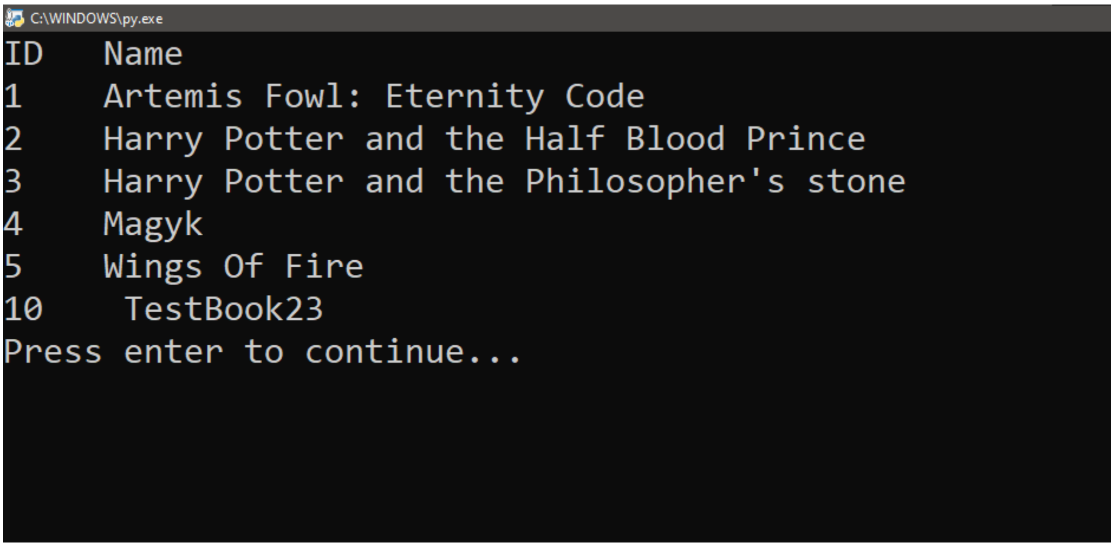
```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6

****************** MANAGE BOOKS ******************

1. Enter new books
2. Edit book information
3. Delete a book
```

a. Enter new books



C:\WINDOWS\py.exe

```
Please enter the details of book

Book ID: 10
Book name: TestBook23
Book genre: genre
Name of the author: author
Name of publisher: publisher
Number of copies: 10
Enter a short summary for the book: this is a short summary


Confirm entry? (y/n): y
SUCCESS: The new records were stored in the database.
Would you like to enter another record? (y/n): n
Press enter to return to main menu...
```

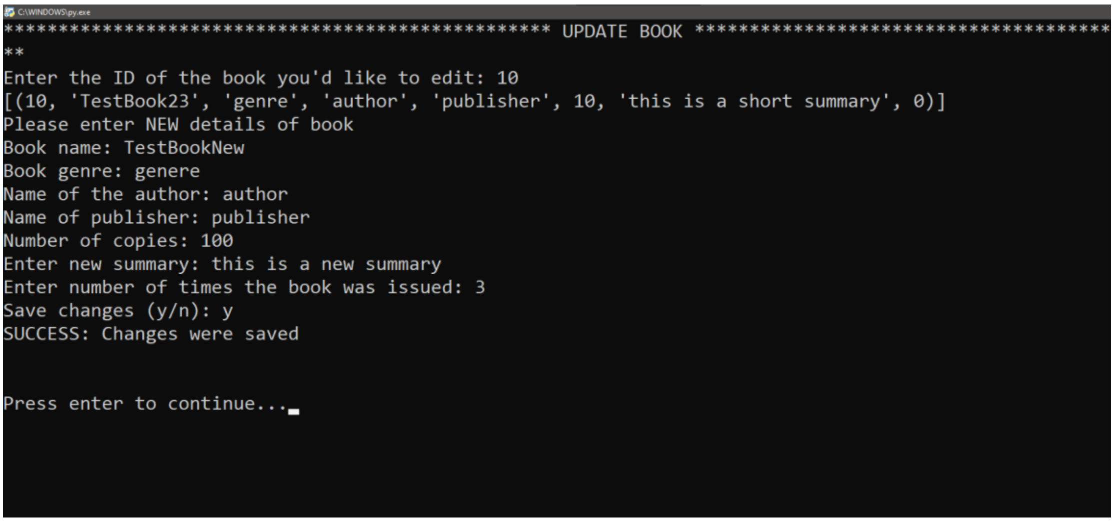


C:\WINDOWS\py.exe

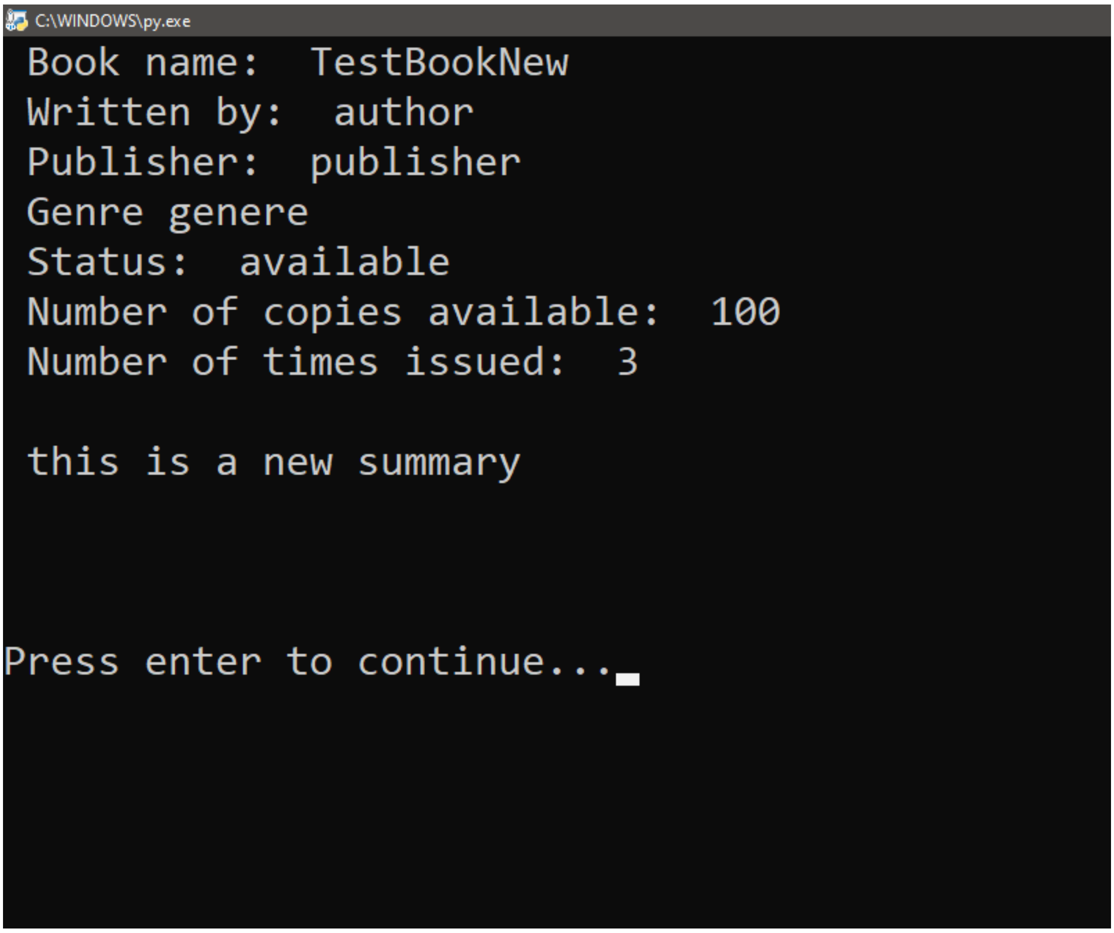
```
ID    Name
1     Artemis Fowl: Eternity Code
2     Harry Potter and the Half Blood Prince
3     Harry Potter and the Philosopher's stone
4     Magyk
5     Wings Of Fire
10     TestBook23
Press enter to continue...
```

The new book was added

b. Edit books



```
************************************************** UPDATE BOOK **************************************
**
Enter the ID of the book you'd like to edit: 10
[(10, 'TestBook23', 'genre', 'author', 'publisher', 10, 'this is a short summary', 0)]
Please enter NEW details of book
Book name: TestBookNew
Book genre: genere
Name of the author: author
Name of publisher: publisher
Number of copies: 100
Enter new summary: this is a new summary
Enter number of times the book was issued: 3
Save changes (y/n): y
SUCCESS: Changes were saved


Press enter to continue...
```

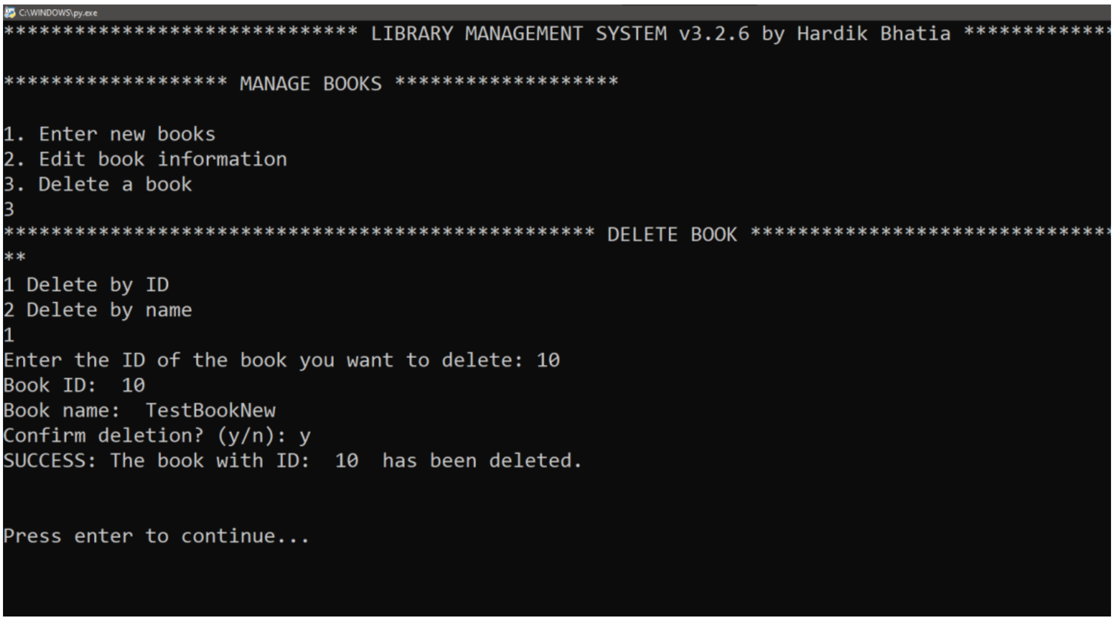


```
 Book name:  TestBookNew
 Written by:  author
 Publisher:  publisher
 Genre genere
 Status:  available
 Number of copies available:  100
 Number of times issued:  3

 this is a new summary



Press enter to continue...
```

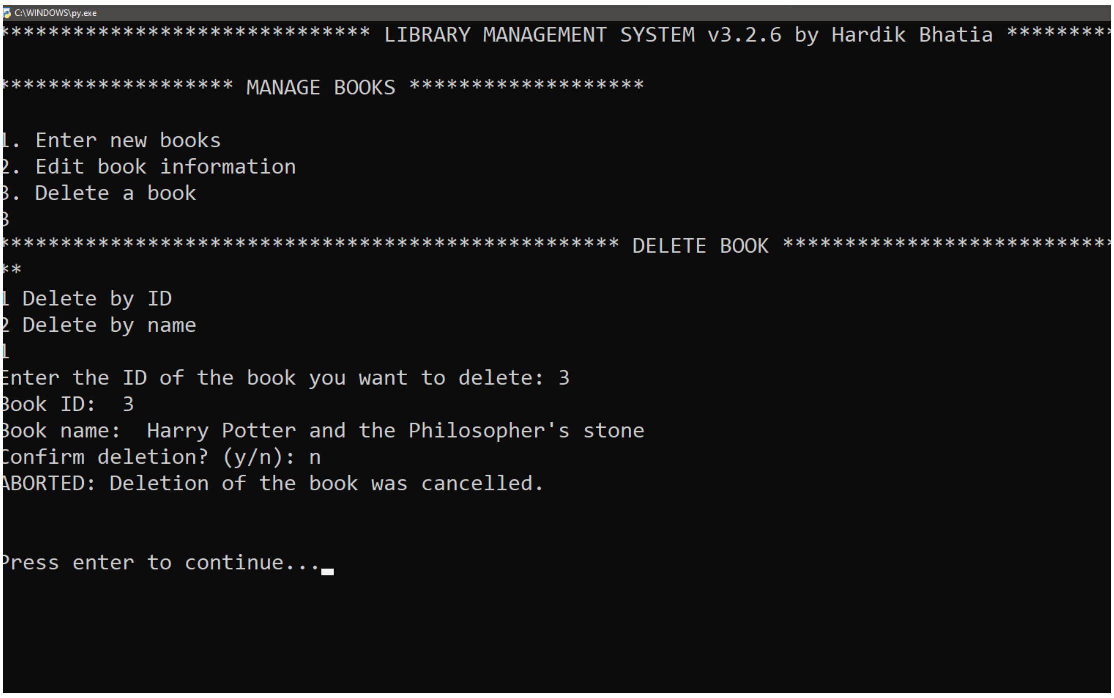
New details of the book using the “Check full details of book” function

c. Delete books

This can be done in 2 ways, for the sake of simplicity I will stick to deleting by ID however the “Delete by name function operates in much the same way”

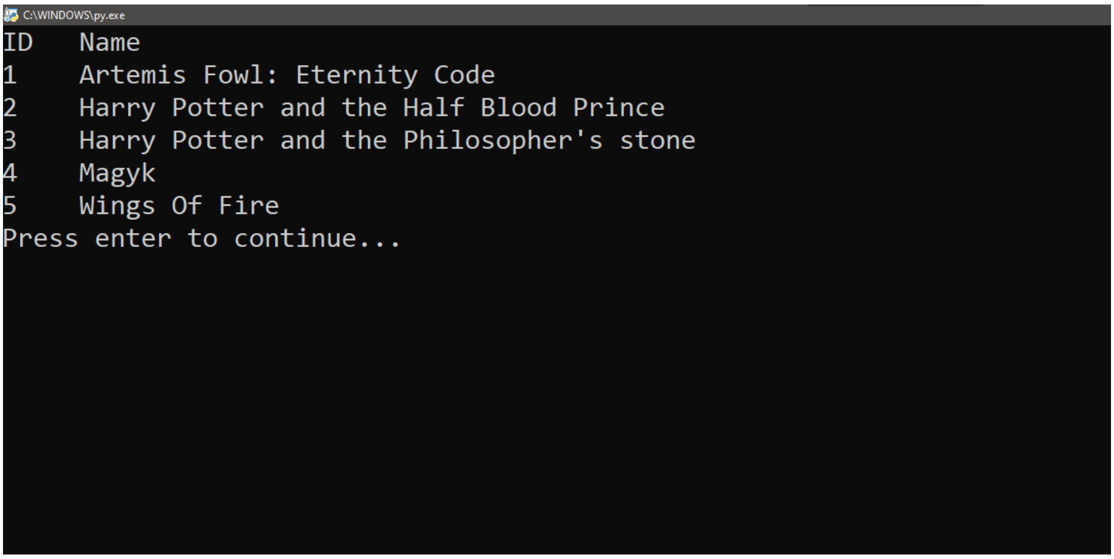


```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ************

****************** MANAGE BOOKS ******************

1. Enter new books
2. Edit book information
3. Delete a book
3
************************************************** DELETE BOOK ******************************
**
1 Delete by ID
2 Delete by name
1
Enter the ID of the book you want to delete: 10
Book ID:  10
Book name:  TestBookNew
Confirm deletion? (y/n): y
SUCCESS: The book with ID:  10  has been deleted.


Press enter to continue...
```



```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik Bhatia ********

****************** MANAGE BOOKS ******************

1. Enter new books
2. Edit book information
3. Delete a book
3
************************************************** DELETE BOOK **************************
**
1 Delete by ID
2 Delete by name
1
Enter the ID of the book you want to delete: 3
Book ID:  3
Book name:  Harry Potter and the Philosopher's stone
Confirm deletion? (y/n): n
ABORTED: Deletion of the book was cancelled.


Press enter to continue...
```

Deletion can be aborted if you accidentally entered the wrong ID



```
ID   Name
1    Artemis Fowl: Eternity Code
2    Harry Potter and the Half Blood Prince
3    Harry Potter and the Philosopher's stone
4    Magyk
5    Wings Of Fire
Press enter to continue...
```

Changes can always be confirmed using other functions

2. Book Information

Same as user-mode no additional features here for administrators

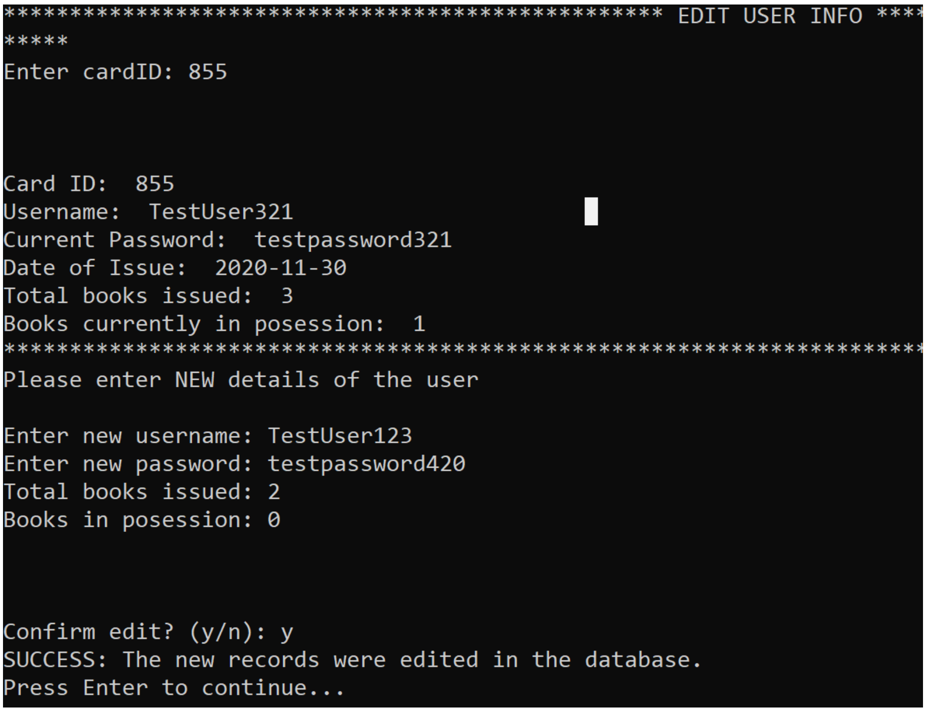
3. Issue / Return a book

Administrator can perform this function but the ID of the user is not specified during log-in so it is asked for before issuing/returning. Other than that the function is the same for both admin and users.

4. Manage users

    a. Register users
    Same for both admin and users

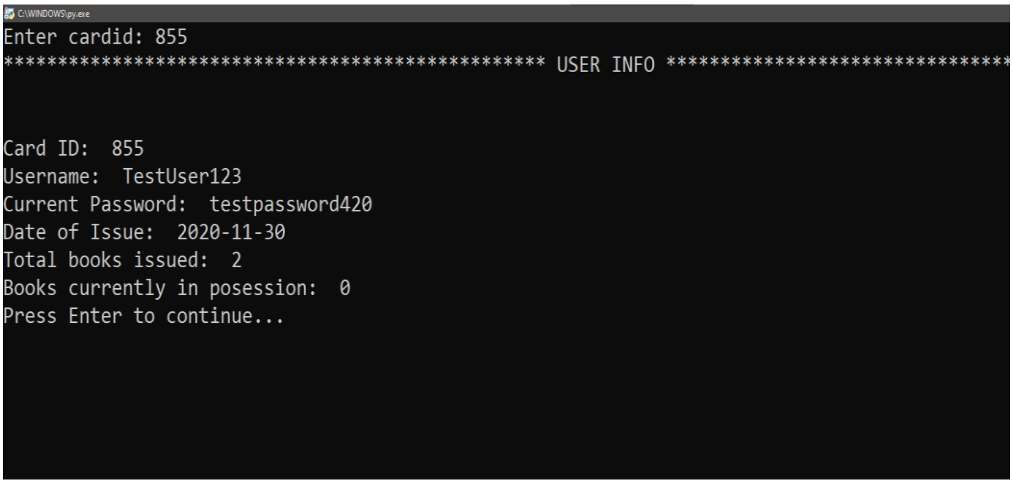
    b. Edit user details
    There are some differences here for the admin and users like asking for total books issued and books in possession in case something unexpected happens in real life and for some reason it requires changes, the option is there. If no changes are required then they can be re-entered as they are displayed above as-well.



```
************************************************** EDIT USER INFO ****
*****
Enter cardID: 855



Card ID:  855
Username:  TestUser321
Current Password:  testpassword321
Date of Issue:  2020-11-30
Total books issued:  3
Books currently in posession:  1
**********************************************************************
Please enter NEW details of the user

Enter new username: TestUser123
Enter new password: testpassword420
Total books issued: 2
Books in posession: 0



Confirm edit? (y/n): y
SUCCESS: The new records were edited in the database.
Press Enter to continue...
```

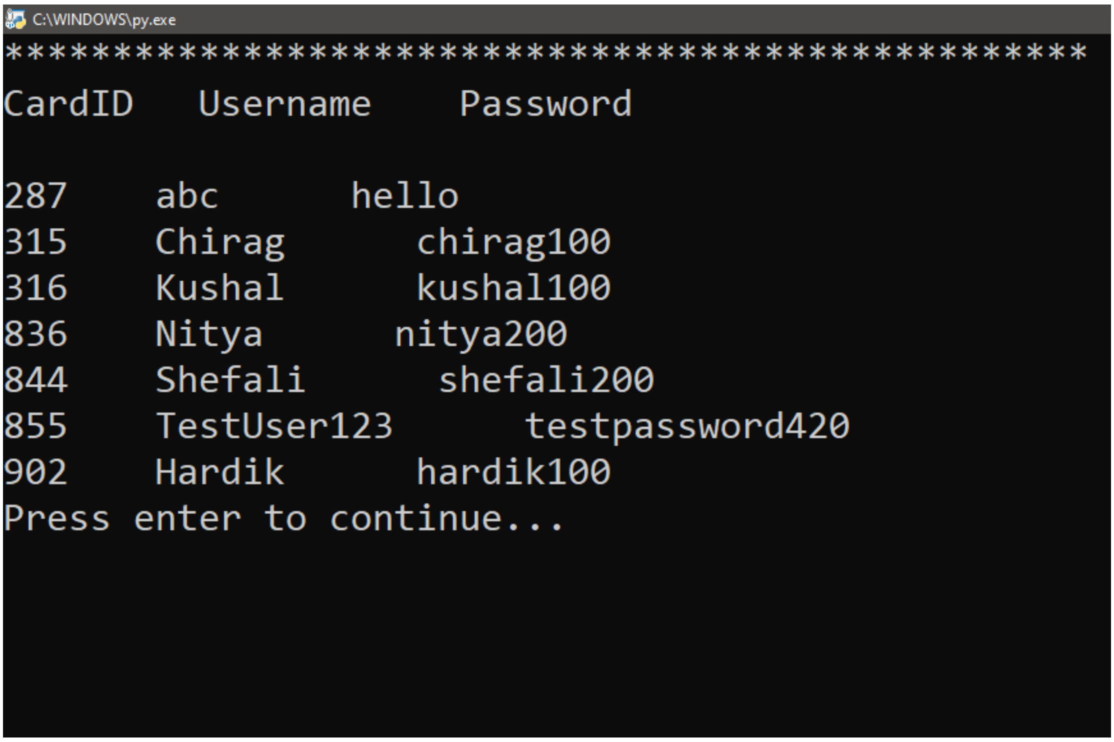
c. Details of user

This is shown in much the same way except the admin can see anyone’s user details by choosing the correct card ID



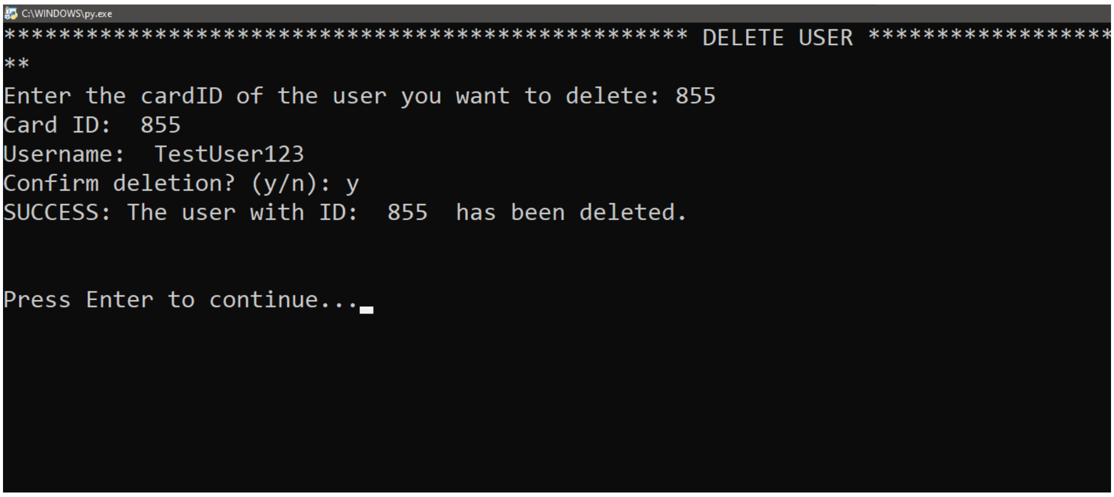
```
Enter cardid: 855
************************************************** USER INFO ********************************


Card ID:  855
Username:  TestUser123
Current Password:  testpassword420
Date of Issue:  2020-11-30
Total books issued:  2
Books currently in posession:  0
Press Enter to continue...
```

d. List of all users
This is an admin-only function which allows admin to see the entire database of users with their passwords and usernames

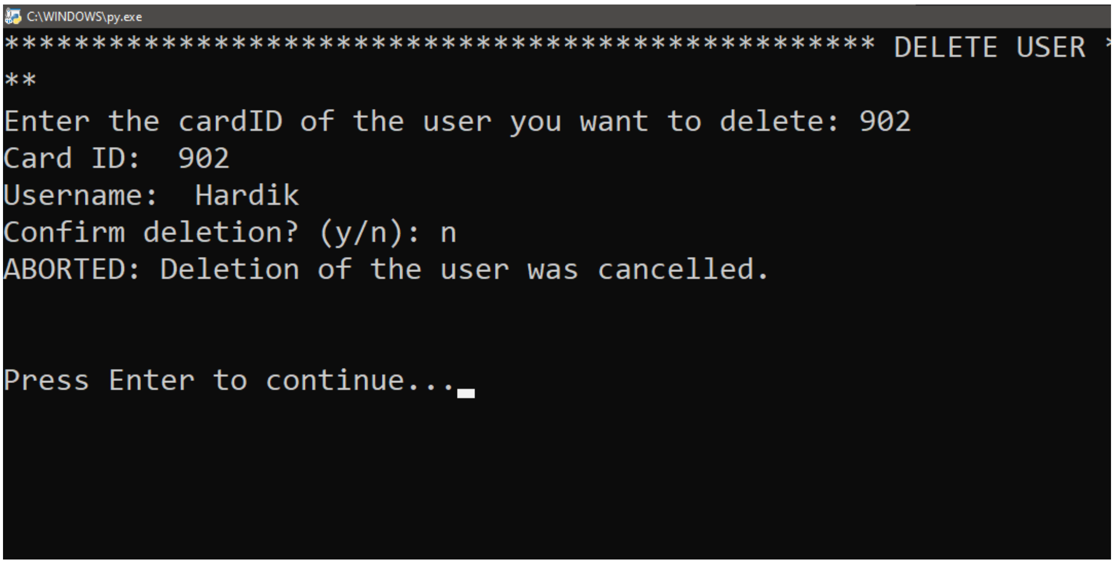


```
**************************************************
CardID   Username    Password

287    abc      hello
315    Chirag      chirag100
316    Kushal      kushal100
836    Nitya      nitya200
844    Shefali      shefali200
855    TestUser123      testpassword420
902    Hardik      hardik100
Press enter to continue...
```

e. Delete users
This functions in the same way as deleting books but instead administrator can delete users using this

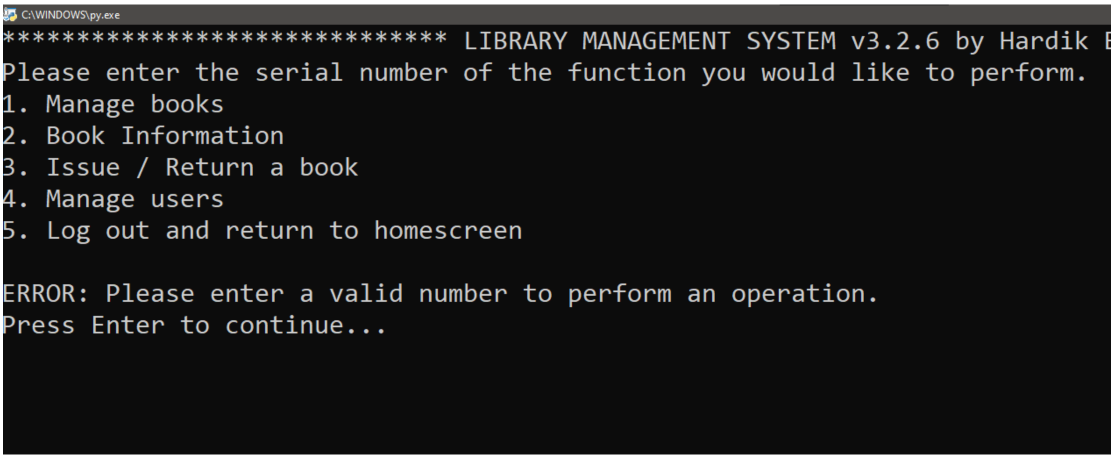


```
************************************************** DELETE USER ******************
**
Enter the cardID of the user you want to delete: 855
Card ID:  855
Username:  TestUser123
Confirm deletion? (y/n): y
SUCCESS: The user with ID:  855  has been deleted.


Press Enter to continue...
```



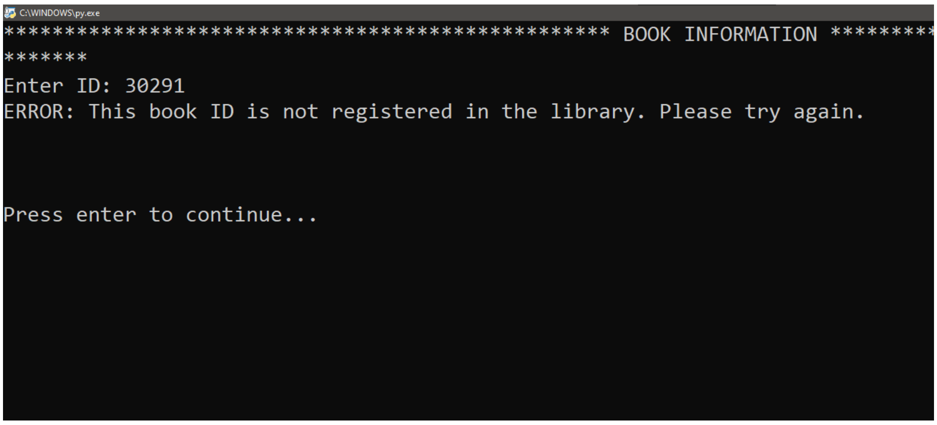
```
************************************************** DELETE USER *
**
Enter the cardID of the user you want to delete: 902
Card ID:  902
Username:  Hardik
Confirm deletion? (y/n): n
ABORTED: Deletion of the user was cancelled.


Press Enter to continue...
```

Part 4: Miscellaneous

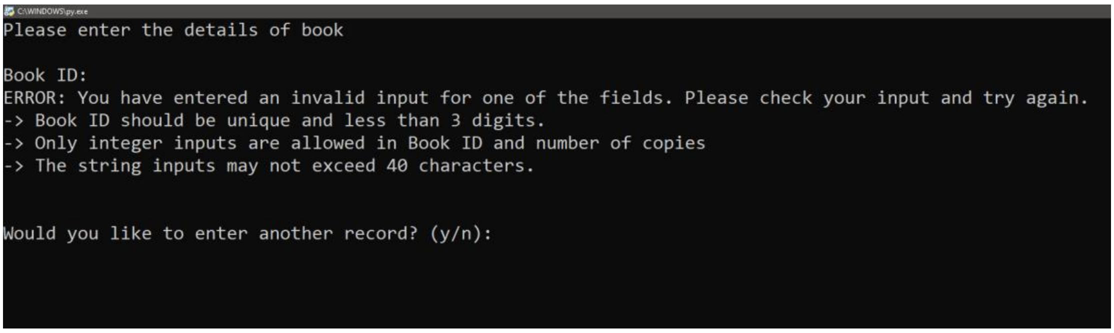


```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by Hardik E
Please enter the serial number of the function you would like to perform.
1. Manage books
2. Book Information
3. Issue / Return a book
4. Manage users
5. Log out and return to homescreen

ERROR: Please enter a valid number to perform an operation.
Press Enter to continue...
```

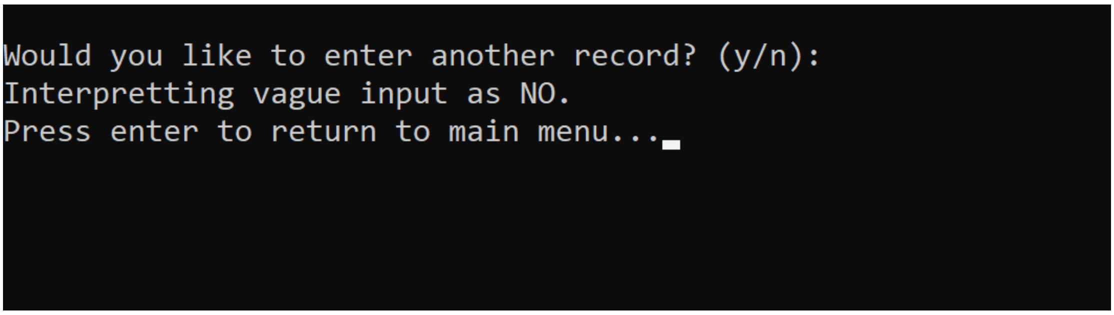
If you accidentally leave fields blank it will give you an error telling you what went wrong.



```
************************************************** BOOK INFORMATION ********
*******
Enter ID: 30291
ERROR: This book ID is not registered in the library. Please try again.



Press enter to continue...
```

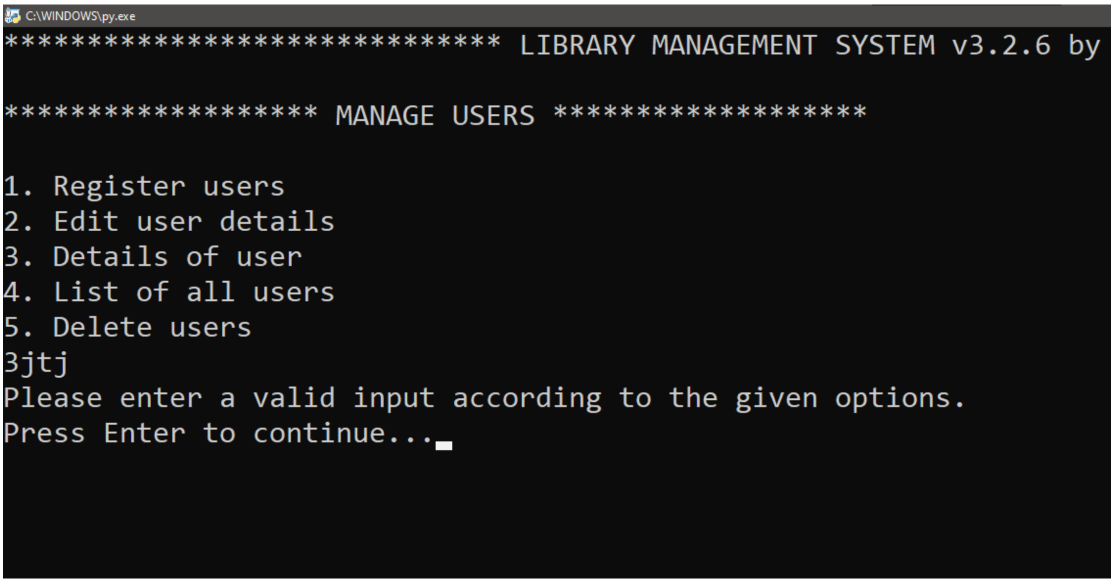


```
Please enter the details of book

Book ID:
ERROR: You have entered an invalid input for one of the fields. Please check your input and try again.
-> Book ID should be unique and less than 3 digits.
-> Only integer inputs are allowed in Book ID and number of copies
-> The string inputs may not exceed 40 characters.


Would you like to enter another record? (y/n):
```



```
Would you like to enter another record? (y/n):
Interpretting vague input as NO.
Press enter to return to main menu...
```

If you leave a field blank it will accept it as No/Yes depending on the function, for example if you leave a user deletion field blank, it will assume no unless it is given ‘y’ as input.



```
****************************** LIBRARY MANAGEMENT SYSTEM v3.2.6 by

******************* MANAGE USERS *******************

1. Register users
2. Edit user details
3. Details of user
4. List of all users
5. Delete users
3jtj
Please enter a valid input according to the given options.
Press Enter to continue...
```

REQUIREMENTS

Minimum hardware requirements:

1. COMPUTER- For coding and typing the required documents for this project
2. PROCESSOR – Core 2 Duo
3. RAM – 128 MB
4. DISK SPACE – 200 MB

Minimum software requirements:

1. OPERATING SYSTEM – Windows 7
2. PYTHON 3.7 – For running the code
3. MYSQL COMMAND LINE CLIENT 5.5 - For a database management
4. PYTHON mysql-connector – For connecting the 2 softwares
5. MS WORD – For presentation of output

BIBLIOGRAPHY

1. SUMITA ARORA COMPUTER SCIENCE WITH PYTHON CLASS 12
2. SUMITA ARORA COMPUTER SCIENCE WITH PYTHON CLASS 11
3. Stackoverflow.com
4. W3schools.com
5. Geeksforgeeks.com
6. Github.com