

**ENGINEERING EXPLORATION PROJECT REPORT**

**LIBRARY MANAGEMENT SYSTEM**

**Prepared for**

The Department of Engineering Exploration

Chaitanya Bharathi Institute of Technology, Hyderabad

**Prepared by**

Nikhitha Tubati ------------------------------------------------------160120733012

Ameen------------------------------------------------------------------160120733024

Philip -------------------------------------------------------------------160120733036

Sai Vishal --------------------------------------------------------------160120733048

Vishnu Sathwik Rebally---------------------------------------------160120733060

**ACKNOWLEDGEMENT**

We would like to express our deepest appreciation to all those who provided us with the possibility to complete this project. Special gratitude to our project mentor, Smt. Isha Padhy ma’am whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this report. Furthermore, we would also like to acknowledge with much appreciation the department of Engineering Exploration to give us this humble opportunity. A special thanks to our family members and friends for the support and encouragement they have given to us over the course.

**ABSTRACT**

**Library Management System – Python Project**

A College Library management is a project that manages and stores books information electronically according to student needs along with the student information. This helps both students and library managers to keep a constant track of all the books available in the library. It allows both the admin and the student to search for the desired book. It becomes necessary for colleges to continuously check the books issued and returned and even calculate fines. This task if carried out manually includes chances of mistakes. These errors are avoided by allowing the system to keep track of information such as issue date, last date to return the book and even fine information and thus there is no need to keep manual track of this information which avoids the chances of mistakes.

This project aims to create an interactive library management system using python as a base programming language, Tkinter library for graphical user interface and files for data management.

**INTRODUCTION**

The library management system is a project written in python programming language using Tkinter. This project is written to simulate the role of a librarian in a library. The features of this project are it can keep track of all the books available in the library. It has a very interactive and simple user interface GUI that both student and admin can use to receive or return books and admin to monitor the process and make sure everything goes correctly.

The code of this project is written in Python 3.9 using the Tkinter module for GUI and this project uses a text file(.txt) to store all the books available in the library and students can make changes to this file as it is regularly updated whenever a student needs a book or want to return a book. The admin can use this project to delete a particular book from the library and add any new books purchased to issue for the students.

This project contains 6 python files which are **mainwindow.py**, **add\_book.py**, **view\_books.py**, **remove\_book.py**, **issue\_book.py**, **return\_book.py** and all of the information regarding the books are stored in the **books.txt** file.

**Project aims:**

The project aims and objectives that will be achieved after completion of this project are:

* A database containing the list of all books in the library.
* Facility to mark book either as available or issued.
* Add various books to the library database.
* To store the information regarding the author’s name and book id.
* Facility to issue an available book to students.
* To get the student id while the student borrows a book.
* Facility to remove books from the system.
* Facility to allow students to return the book to the library.

**SYSTEM REQUIREMENTS:**

**Software Requirements**:

Python 3.9 IDLE or PyCharm

Windows (7 or above) or Mac OS (10.11 or higher)

Installed Tkinter library

**Hardware Requirements:**

Processor: Intel I3 or above

Disk space: 1 GB or more

**Existing system vs Proposed system:**

Early days Libraries are managed manually. It required a lot of time to record or retrieve the details. The employees who have to record the details must perform their job very carefully. Even a small mistake would create a lot of problems. Security of information is very less. Report generations of all the information are a very tough task.

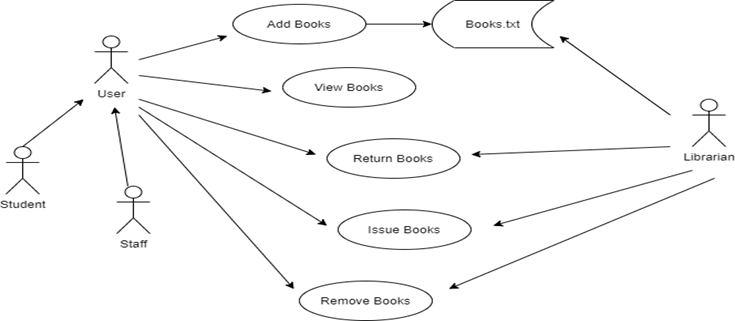
Maintenance of the Library catalogue and arrangement of the books to the catalogue is a very complex task. In addition to its maintenance of member details, issue dates and return dates etc. manually is a complex task.

All the operations must be performed perfectly for the maintenance of the library without any degradation which may finally fail the entire system.

To solve the inconveniences as mentioned in the existing system, a Library Management System is proposed. The proposed system contains the following features:

* A single file-based management system that holds the record of the book id, book name, author and whether the book is available or has been issued.
* A function to add new books to the library database.
* A function to remove books from the database.
* A function that issues the required book to a student depending on its availability.
* A function that manages the return of borrowed books from the students.
* Pop up errors if there is any fallacy in book id or student id.

**METHODOLOGY:**

The above level diagram is the functioning of the Library Management System in a very lucid way. The user can either be a staff or a student. The system will provide view functionality to view different books available in the library. Further, the library staff person can add/update the resources. The users of the system can request to issue/return the book for which they have to follow certain criteria.

**Description of the Files:**

* **mainwindow.py** – which does function call to all other python files
* **add\_book.py** – To add the book
* **view\_books.py** – To view the list of books in the library
* **remove\_book.py** – To remove a book from library
* **issue\_book.py** – To issue a book from library
* **return\_book.py** – To return a book to the library
* **books.txt** – To store the data related to books in the library

**Algorithm:**

**mainwindow.py file:**

Step-1: Start

Step-2: import tkinter module

Step-3: Define a function view()

Step-3.1: from view\_books import viewbooks

Step-3.2: viewbooks()

Step-4: Define a function add()

Step-4.1: from add\_book import bookadd

Step-4.2: bookadd()

Step-5: Define a function remove()

Step-5.1: from remove\_book import bookremove

Step-5.2: bookremove()

Step-6: Define a function issue()

Step-6.1: from issue\_book import bookissue

Step-6.2: bookissue()

Step-7: Define a function return()

Step-7.1 from return\_book import bookreturn

Step-7.2 bookreturn()

Step-8: Create a window of geometry 700 x 600 with the title “Library Management System - Dashboard”

Step-9: Add a background image to the window

Step-10: Add a Label “Library Management system” of the desired style and pack it

Step-11: Add a Label “WELCOME TO LIBRARY” of the desired style and pack it.

Step-12: Add a Label “Pick an option to continue” of the desired style and pack it

Step-13: Add a Button “View Books” with the command view of the desired style and pack it

Step-14: Add a Button “Add Book” with command add of the desired style and pack it

Step-15: Add a Button “Remove Book” with the command remove of the desired style and pack it

Step-16: Add a Button “Issue Book to student” with command issue of the desired style and pack it

Step-17: Add a Button “Return Book” with the command return of the desired style and pack it

Step-18: End

**view\_books.py file:**

Step-1: Start

Step-2: import Tkinter module

Step-3: Define a function viewbooks()

Step-3.1: Create a window of geometry 700 x 780 with title “View Books”

Step-3.1: Add a Label “Books Details” of desired style and pack it

Step-3.2: Create a Frame of size 700 X 780 of desired style and pack it

Step-3.3: Create a Label “book id : {book details}” of desired style and pack it

Step-3.4: with open (“books.txt”, “r”) as f:

Step-3.4.1: Read the data present in the text

Step-3.4.2: data = data.split(‘\n’)

Step-3.4.3: print(data)

Step-3.4.4: for i in data:

Step-3.4.4.1: Create a Label i of the desired style and pack it

Step-4:End

**add\_book.py file:**

Step-1:Start

Step-2:import Tkinter module

Step-3:Define a function bookadd()

Step-3.1: Declare global variables bookinfo1, bookinfo2, bookinfo3, bookinfo4

Step-3.2: Define a function addbook()

Step-3.2.1: book\_id = bookinfo1.get()

Step-3.2.2: title = bookinfo2.get()

Step-3.2.3: author = bookinfo3.get()

Step-3.2.4: status = bookinfo4.get()

Step-3.2.5: with open(‘books.txt’, ‘a’) as f:

Step-3.2.5.1: Add book\_id, title, author, status to the text file

Step-3.2.6: Display a message box “Book added successfully”

Step-3.3: Create a window of geometry 560 x 490 with the title “Library Management System”

Step-3.4: Add a Label “ADD BOOK” of the desired style and pack it

Step-3.5: Add a Label “Enter the following details” of the desired style and pack it

Step-3.6: Create a Frame of size 800 X 600 of the desired style and pack it

Step-3.7:Add a Label “Book ID:” of the desired style and pack it

Step-3.8: bookinfo1 = Entry of labelframe of Book ID

Step-3.9: Add a Label “Title:” of the desired style and pack it

Step-3.10: bookinfo2 = Entry of labelframe of Title

Step-3.11:Add a Label “Author:” of the desired style and pack it

Step-3.12: bookinfo3 = Entry of labelframe of Author

Step-3.13:Add a Label “Status (avail/issued):” of the desired style and pack it

Step-3.14: bookinfo4 = Entry of labelframe of Status

Step-3.15: Add a Button “Submit” with command addbook of the desired style and pack it

Step-3.16: Add a Button “Quit” with command root5.destroy and pack it

Step-4: End

**remove\_book.py file:**

Step-1: Start

Step-2: import tkinter module

Step-3: Define a function bookremove()

Step-3.1:Declare global variables book\_id,root2

Step-3.2:Define a function removebook()

Step-3.2.1:bid = book\_id.get()

Step-3.2.2: if bid:

Step-1:with open("books.txt", "r") as f\_old:

Step-1.1: data = f\_old.read

Step-1.2: with open(“books.text”,”w”) as f\_new:

Step-1.2.1: data=data.split(‘\n’)

Step-1.2.2: for j in data:

Step-1.2.2.1: if bid in j:

Step-1.2.2.1.1: pass

Step-1.2.2.2: else

Step-1.2.2.2.1: f\_new.write(f”{j}\n”)

Step-2: Display a message book “Book removed from data”

Step-3.2.3: else:

Step-1: Display a message book “Invalid ID,Please enter correct ID”

Step-3.3: Create a window of geometry 455 x 300 with title “Remove Book”

Step-3.4: Add a Label “REMOVE BOOK” of desired style and pack it

Step-3.5: Create a Frame of size 10 X 700 of desired style and pack it

Step-3.6: Add a Label “Enter book id :” of desired style and pack it

Step-3.7: book\_id = Entry of labelframe of Enter book id

Step-3.8:Add a Button “Exit” with command root2.destroy

Step-3.9: Add a Button “Submit” with command removebook and pack it

Step-4: End

**issue\_book.py file:**

Step-1: Start

Step-2: import tkinter module

Step-3: Define a function bookissue()

Step-3.1: Declare global variables book\_id, student\_id, root3

Step-3.2: Define a function issuebook()

Step-3.2.1: p = book\_id.get()

Step-3.2.2: q = student\_id.get()

Step-3.2.3: if p and q :

Step-1: with open(“books.txt”, “r”) as f:

Step-1.1:Read data of the file

Step-1.2: if p in data:

Step-1.2.1:Display a message book “Book issued successfully”

Step-1.3: else:

Step-1.3.1: Display a message book “Please enter a valid id”

Step-3.2.4: else:

Step-3.2.4.1: Display a message book “Please enter a valid id”

Step-3.3: Create a window of geometry 440 x 400 with title “Issue Book”

Step-3.4: Add a Label “ISSUE BOOK” of desired style and pack it

Step-3.5: Create a Frame of size 600 X 680 of desired style and pack it

Step-3.6: Add a Label “Enter book id :” of desired style and pack it

Step-3.7: book\_id = Entry of labelframe of Enter book id

Step-3.8: Add a Label “Enter student id :” of desired style and pack it

Step-3.9: student\_id = Entry of labelframe of Enter student id

Step-3.10: Add a Button “Exit” with command root3.destroy

Step-3.11: Add a Button “Submit” with command issuebook and pack it

Step-4: End

**return\_book.py file:**

Step-1: Start

Step-2: import tkinter module

Step-3: Define a function bookreturn()

Step-3.1: Define global variables book\_id, student\_id, root1

Step-3.2: Define a function returnbook()

Step-3.2.1: p = book\_id.get()

Step-3.2.2: q = student\_id.get()

Step-3.2.3: if p and q:

Step-3.2.3.1: Display a message book “Book returned successfully”

Step-3.2.4: else:

Step-3.2.4.1: Display a message book “Please enter a valid id”

Step-3.3: Create window of geometry 455 x 400 with title “Return Book”

Step-3.4: Add a Label “RETURN BOOK” of desired style and pack it

Step-3.5: Create a Frame of size 600 X 700 of desired style and pack it

Step-3.6: Add a Label “Enter book id :” of desired style and pack it

Step-3.7: book\_id = Entry of labelframe of Enter book id

Step-3.8: Add a Label “Enter student id :” of desired style and pack it

Step-3.9: student\_id = Entry of labelframe of Enter student id

Step-3.10: Add a Button “Exit” with command root1.destroy

Step-3.11: Add a Button “Submit” with command returnbook and pack it

Step-4: End

**Code And Implementation:**

**Source Code:**

[Library Management System (E.E Project)](https://drive.google.com/drive/folders/1GXF8wnNP5fdUfvKAbP_KW7P_5Q8m20f1)

**view\_books.py file:**  
from tkinter import \*

def viewbooks():

global root4

root4=Tk()

root4.geometry("700x780")

root4.minsize(700,780)

root4.title("View Books")

heading=Label(root4,text="BOOKS DETAILS",bg='black',fg="yellow",font="gotham 20 bold",borderwidth=10,relief=SUNKEN,padx=10,pady=4)

heading.pack(pady=5,fill=X)

main\_frame=Frame(root4,height=700,width=780,borderwidth=8,relief=RIDGE,bg="grey",padx=8)

main\_frame.pack(side=LEFT,anchor="nw",fill=X and Y)

Label(main\_frame,text="book id : {book details}", font="cambria 13 bold",pady=5,bg="grey").pack(anchor="nw")

with open("books.txt", "r") as f:

data=f.read()

data=data.split('\n')

print(data)

for i in data:

Label(main\_frame,text=f"{i}",font="times 12",bg="grey").pack(anchor="nw")

root4.mainloop()

**add\_book.py file:**

from tkinter import \*

from tkinter import messagebox

def bookadd():

global bookinfo1, bookinfo2, bookinfo3, bookinfo4, root5

def addbook():

book\_id = bookinfo1.get()

title = bookinfo2.get()

author = bookinfo3.get()

status = bookinfo4.get()

with open('books.txt', 'a') as f:

f.write(f"\n{book\_id}: {{TITLE: {title}, AUTHOR: {author}, STATUS: {status} }}")

messagebox.showinfo('Success',"Book added successfully")

root5.destroy()

return

root5=Tk()

root5.geometry("560x490")

root5.minsize(560,490)

root5.title("Library Management System")

Label(root5,text="ADD BOOK",font=("times",28,"bold"),fg="yellow",bg="black",borderwidth=12,relief=SUNKEN).pack(side=TOP,fill=X)

Label(root5,text=" Enter the following details ",fg="black").pack(pady=5)

labelframe = Frame(root5,height=800,width=600,bg='black')

labelframe.place(relx=0.1,rely=0.3,relwidth=0.8,relheight=0.45)

lb1 = Label(labelframe,text='Book ID : ',bg='black',fg='white')

lb1.place(relx=0.05,rely=0.2,relheight=0.08)

bookinfo1 = Entry(labelframe)

bookinfo1.place(relx=0.3,rely=0.2,relwidth=0.62,relheight=0.08)

lb2 = Label(labelframe,text='Title : ',bg='black',fg='white')

lb2.place(relx=0.05,rely=0.35,relheight=0.08)

bookinfo2 = Entry(labelframe)

bookinfo2.place(relx=0.3,rely=0.35,relwidth=0.62,relheight=0.08)

lb3 = Label(labelframe,text='Author : ',bg='black',fg='white')

lb3.place(relx=0.05,rely=0.50,relheight=0.08)

bookinfo3 = Entry(labelframe)

bookinfo3.place(relx=0.3,rely=0.50,relwidth=0.62,relheight=0.08)

lb4 = Label(labelframe,text='Status(avail/issued) : ',bg='black',fg='white')

lb4.place(relx=0.05,rely=0.65,relheight=0.08)

bookinfo4 = Entry(labelframe)

bookinfo4.place(relx=0.3,rely=0.65,relwidth=0.62,relheight=0.08)

Submit = Button(labelframe,text='Submit',font=("times 11 bold"),command=addbook)

Submit.place(relx=0.3,rely=0.83,relwidth=0.18,relheight=0.15)

Quit = Button(labelframe,text='Quit',font=("times 11 bold"),command=root5.destroy)

Quit.place(relx=0.53,rely=0.83,relwidth=0.18,relheight=0.15)

root5.mainloop()

#bookadd()

**remove\_book.py file:**  
from tkinter import \*

from tkinter import messagebox

def bookremove():

global book\_id, root2

def removebook():

bid=book\_id.get()

if bid:

with open("books.txt", "r") as f\_old:

data=f\_old.read()

with open("books.txt", "w") as f\_new:

data=data.split('\n')

for j in data:

if bid in j:

pass

else:

f\_new.write(f"{j}\n")

messagebox.showinfo("Book Removed","Book removed from data.")

else:

messagebox.showerror("Invalid ID","Please enter correct book id.")

root2.destroy()

return

root2=Tk()

root2.geometry("455x300")

root2.maxsize(455,300)

root2.minsize(455,300)

root2.title("Remove Book")

heading=Label(root2,text="REMOVE BOOK",bg='black',fg="yellow",font="gotham 28 bold",borderwidth=10,relief=SUNKEN,padx=75,pady=8)

heading.grid(row=0,column=0)

f1=Frame(root2,height=10,width=700,bg="black",pady=30,padx=80)

f1.grid(row=3,column=0,pady=5)

Label(f1,text="Enter book id:",bg="black",fg="white",font="comicsanms 11",padx=10,pady=10).grid(row=3,column=1)

book\_id=Entry(f1)

book\_id.grid(row=3,column=2)

Button(f1,text="Exit",font=("times 11 bold"),command=root2.destroy,padx=8).grid(row=6,column=2,padx=10,pady=10)

Button(f1,text="Submit",font=("times 11 bold"),command=removebook).grid(row=5,column=2,padx=10,pady=10)

root2.mainloop()

#bookremove()

**issue\_book.py file:**

from tkinter import \*

from tkinter import messagebox

def bookissue():

global book\_id, student\_id, root3

def issuebook():

p=book\_id.get()

q=student\_id.get()

if p and q:

with open("books.txt", "r") as f:

data=f.read()

if p in data:

messagebox.showinfo("Issues","Book successfully issued to student.")

else:

messagebox.showerror("Invalid ID","Please enter a valid book ID")

else:

messagebox.showerror("Invalid ID","Please enter a valid ID")

root3.destroy()

return

root3=Tk()

root3.geometry("440x400")

root3.maxsize(440,400)

root3.minsize(440,400)

root3.title("Issue Book")

heading=Label(root3,text=" ISSUE BOOK ",bg='black',fg='yellow',font="gotham 28 bold",borderwidth=10,relief=SUNKEN,padx=80,pady=8)

heading.grid(row=0,column=0)

f1=Frame(root3,height=600,width=680,bg="black",pady=50,padx=80)

f1.grid(row=3,column=0,pady=5)

Label(f1,text="Enter book id:",bg="black",fg="white",font="comicsanms 11",padx=10,pady=10).grid(row=3,column=1)

book\_id=Entry(f1)

book\_id.grid(row=3,column=2)

Label(f1,text="Enter student id:",bg="black",fg="white",font="comicsanms 11",padx=10,pady=10).grid(row=4,column=1)

student\_id=Entry(f1)

student\_id.grid(row=4,column=2)

Button(f1,text="Exit",font=("times 11 bold"),command=root3.destroy,padx=8).grid(row=6,column=2,padx=10,pady=10)

Button(f1,text="Submit",font=("times 11 bold"),command=issuebook).grid(row=5,column=2,padx=10,pady=10)

root3.mainloop()

#bookissue()

**return\_book.py file:**  
from tkinter import \*

from tkinter import messagebox

def bookreturn():

global book\_id, student\_id, root1

def returnbook():

p=book\_id.get()

q=student\_id.get()

if p and q:

messagebox.showinfo("Returns","Book returned successfully. Thank you!")

else:

messagebox.showerror("Invalid ID","Please enter a valid ID")

root1.destroy()

return

root1=Tk()

root1.geometry("455x400")

root1.maxsize(455,400)

root1.minsize(455,400)

root1.title("Return Book")

heading=Label(root1,text="RETURN BOOK",bg='black',fg='yellow',font="gotham 28 bold",borderwidth=10,relief=SUNKEN,padx=75,pady=8)

heading.grid(row=0,column=0)

f1=Frame(root1,height=600,width=700,bg="black",pady=50,padx=80)

f1.grid(row=3,column=0,pady=5)

Label(f1,text="Enter book id:",bg="black",fg="white",font="comicsanms 11",padx=10,pady=10).grid(row=3,column=1)

book\_id=Entry(f1)

book\_id.grid(row=3,column=2)

Label(f1,text="Enter student id:",bg="black",fg="white",font="comicsanms 11",padx=10,pady=10).grid(row=4,column=1)

student\_id=Entry(f1)

student\_id.grid(row=4,column=2)

Button(f1,text="Exit",font=("times 11 bold"),command=root1.destroy,padx=8).grid(row=6,column=2,padx=10,pady=10)

Button(f1,text="Submit",font=("times 11 bold"),command=returnbook).grid(row=5,column=2,padx=10,pady=10)

root1.mainloop()

#bookreturn()

**mainwindow.py file:**

from tkinter import \*

def view():

from view\_books import viewbooks

viewbooks()

def add():

from add\_book import bookadd

bookadd()

def remove():

from remove\_book import bookremove

bookremove()

def issue():

from issue\_book import bookissue

bookissue()

def returns():

from return\_book import bookreturn

bookreturn()

root=Tk()

root.geometry("700x600")

root.minsize(655,500)

root.title("Library Management System-Dashboard")

bg=PhotoImage(file="bird-library-livestream.png")

Label(root,image=bg).place(x=0,y=0)

Label(root,text="LIBRARY MANAGEMENT SYSTEM",font=("times",28,"bold"),bg="black",fg="yellow",relief=RIDGE,borderwidth=8).pack(side=TOP,fill=X)

Label(root,text=" WELCOME TO THE LIBRARY ",font=("avenir",18,"bold"),fg="blue",bg="lightblue").pack(side=TOP,padx=10,pady=15)

Label(root,text=" Pick an option to continue ",bg="grey",fg="black").pack(pady=5)

Button(root,text="View Books",padx=90,pady=10,command=view,font=("gotham",14),bg="black",fg="lightblue").pack(padx=2,pady=4)

Button(root,text="Add Book",padx=100,pady=10,command=add,font=("gotham",14),bg="black",fg="lightblue").pack(padx=2,pady=4)

Button(root,text="Remove Book",padx=80,pady=10,command=remove,font=("gotham",14),bg="black",fg="lightblue").pack(padx=2,pady=4)

Button(root,text="Issue Book to student",padx=50,pady=10,command=issue,font=("gotham",14),bg="black",fg="lightblue").pack(padx=2,pady=4)

Button(root,text="Return Book",padx=90,pady=10,command=returns,font=("gotham",14),bg="black",fg="lightblue").pack(padx=2,pady=4)

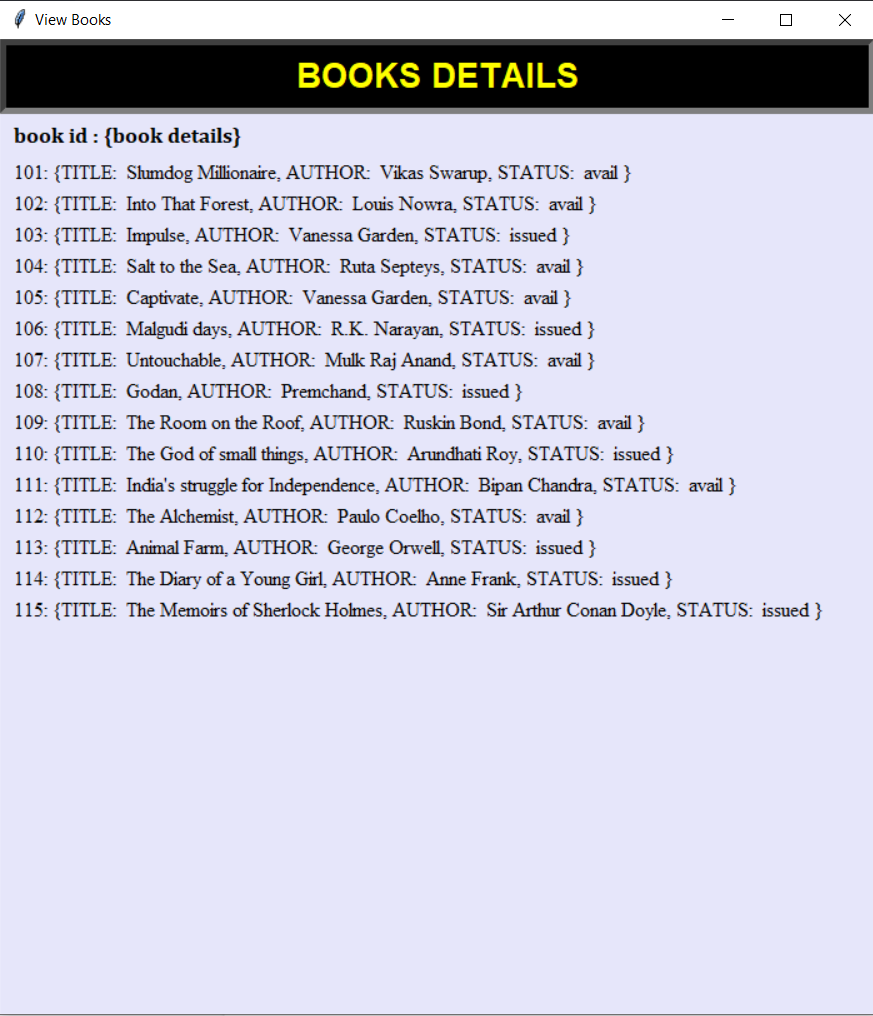
root.mainloop()

**SCREENSHOTS OF THE PROJECT:**

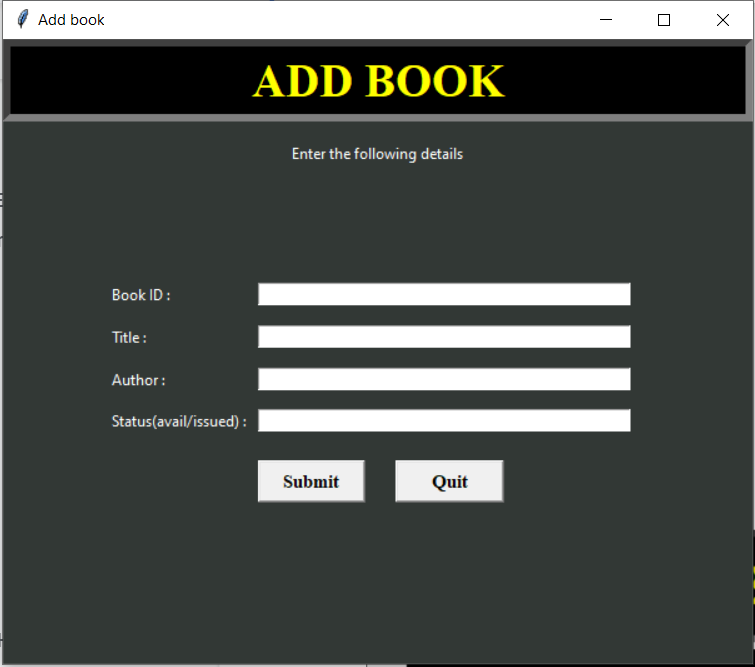
**Main window:**

****

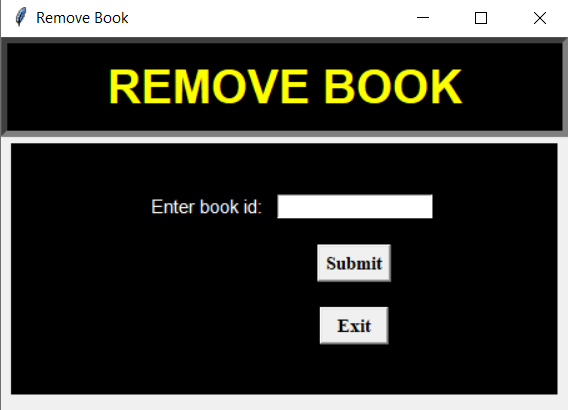
**View books window:**



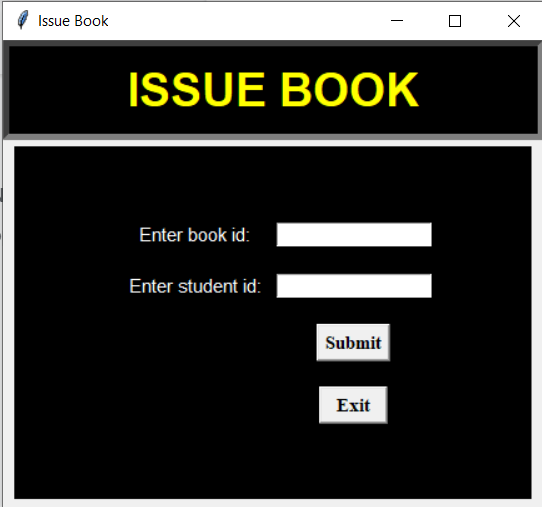
**Add book window:**



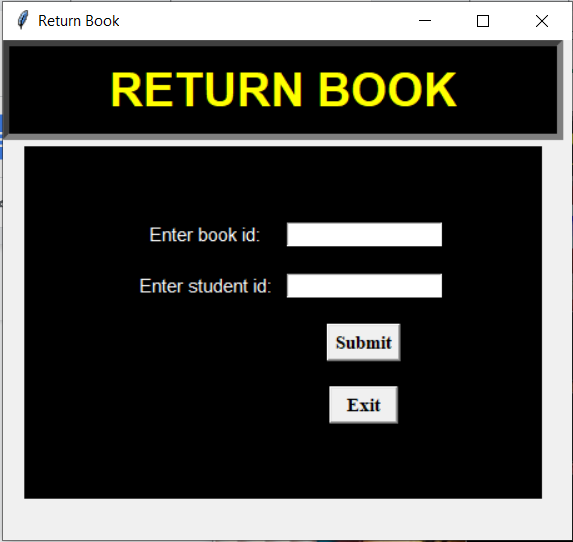
**Remove book window:**



**Issue book window:**



**Return book window:**



**REPORT AND ANALYSIS:**

The above model of the Library Management System makes use of various modules to function as a system with integrity. In such programs, the extent of integration of various modules is a parameter defining the susceptibility of the whole system. However, the above system was observed to be versatile and the modules were integrating without any concerns, making the program run smoothly and perform various executions without any glitches. The handover of command from one module to the other when a desired function was called happened to be errorless and smoothly executed. The add books button which opens a window prompt to add the books to the database stores the data in a file in a dictionary format with the book id as the key. The issue book and return book functions throw an error when an incorrect book or student id is entered. The issue book button opens a prompt that asks the book and student id and issues the book only if it is marked as available in the database. The subtle use of message boxes makes the system more user friendly and easy to use. Although major improvements can be implemented to this system there is no hesitancy that the system is operable, user friendly and fulfils the objectives of this project.

Hereby we conclude that the Library Management System was able to accomplish all of its prerequisites successfully and also has the potential to make the process of library management computerised, relieving the workload from a librarian.