# E-library online public access catalog

### A MINI PROJECT REPORT

**18CSC207J - ADVANCED PROGRAMMING PRACTICE**

***Submitted by***

**GAUTAM SONI(RA21110510020)**

**BHARGAV SINGH(RA2111050010040)**

*Under the guidance of*

## Dr.Divya G

Assistant Professor, Department of DSBS

***in partial fulfillment for the award of the degree***

***of***

### BACHELOR OF TECHNOLOGY

in

## COMPUTER SCIENCE & ENGINEERING

of

### FACULTY OF ENGINEERING AND TECHNOLOGY



S.R.M. Nagar, Kattankulathur, Chengalpattu District

### MAY 2023

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

(Under Section 3 of UGC Act, 1956)

## BONAFIDE CERTIFICATE

Certified that Mini project report titled **E-library online public access catalog** is the bona fide work of **GAUTAM SONI (RA2111050010020)** and **BHARGAV SINGH (RA2111050010040)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

#### SIGNATURE SIGNATURE

Dr. Divya G

#### GUIDE

Assistant Professor

Department of DSBS

Dr. M. LAKSHMI

#### HEAD OF THE DEPARTMENT

Professor & Head

Department of DSBS

# ABSTRACT

An E-library online public access catalog (OPAC) is a digital system that provides access to various resources available in an online library. The system serves as an interface between users and the library's collection of books, articles, journals, multimedia resources, and other materials. OPACs are widely used by libraries worldwide to facilitate access to their collections and provide a better user experience for library patrons.

The E-library online public access catalog is an essential tool that enhances the management of library resources. It allows library staff to efficiently organize, classify, and catalog materials to ensure that they are easily searchable and retrievable. OPACs use various metadata standards to describe resources, including author, title, subject, keyword, and other information that enables users to find materials quickly and efficiently.

The OPAC provides a user-friendly search interface that allows users to search for resources based on various criteria. Users can search by author, title, subject, keyword, and even ISBN or ISSN numbers. OPACs also provide advanced search features that enable users to refine their search results by limiting the search scope to specific formats, date ranges, and other parameters.

In addition to facilitating access to digital resources, the E-library online public access catalog also provides information about the availability and location of physical copies. Users can check the status of items, place holds on materials, and request interlibrary loans through the OPAC. The system also provides users with information about library policies, procedures, and other services, enhancing their overall experience.

The E-library online public access catalog is a powerful tool that promotes knowledge sharing and collaboration. Libraries can use OPACs to share their collections with other institutions and enable users to discover resources beyond their local library. OPACs also provide opportunities for users to contribute to the library's collections by suggesting new materials or submitting reviews and ratings of existing resources.

In conclusion, the E-library online public access catalog is a vital component of modern library services. It enables libraries to manage their resources more efficiently, provides users with a user-friendly search interface, and promotes knowledge sharing and collaboration. As libraries continue to evolve and expand their collections, OPACs will remain an essential tool for providing access to information and fostering learning and discovery.

# TABLE OF CONTENTS

[ABSTRACT 1](#_TOC_250002)

[TABLE OF CONTENTS 2](#_TOC_250001)

1. INTRODUCTION 3
2. LITERATURE SURVEY 4
3. SYSTEM ARCHITECTURE AND DESIGN 5
4. METHODOLOGY 6
5. CODING AND TESTING 7
6. SCREENSHOTS AND RESULTS 10
7. CONCLUSION AND FUTURE ENHANCEMENT 12

[REFERENCES 12](#_TOC_250000)

### CHAPTER 1

**INTRODUCTION**

An Online Public Access Catalog (OPAC) is a database of a library's collection of materials, such as books, journals, and audiovisual materials. It allows library users to search for and locate resources available in the library's collection. With the increasing use of electronic resources and the growth of e-libraries, the traditional OPAC has evolved into an e-library OPAC.

An e-library OPAC is a web-based interface that enables users to search for and access digital resources such as eBooks, eJournals, audiovisual materials, and other online materials. It provides a user-friendly platform that is accessible from anywhere, at any time, using a computer or mobile device with an internet connection. E-library OPACs have become an essential tool for modern libraries, allowing users to access digital resources and enhancing their experience by providing a user-friendly and customizable interface.

E-library OPACs provide users with advanced search capabilities, allowing them to search for resources by author, title, subject, keyword, and more. They also allow users to access resources remotely, as long as they have a valid library membership and internet access. E-library OPACs have revolutionized the way libraries provide access to digital resources, making it easier for users to search for and access information and enhancing the overall library experience.

In recent years, there has been a shift towards user-centered design in the development of e-library OPACs, with a focus on providing a user-friendly interface that meets the needs of library users. Libraries are continuously improving and updating their e-library OPACs to provide users with the best possible experience and access to digital resources.

Overall, e-library OPACs have become an essential tool for modern libraries, providing users with increased access to digital resources and enhancing their overall library experience. They are an integral part of the digital transformation of libraries, providing a platform that is accessible from anywhere, at any time, and from any device with an internet connection.

### CHAPTER 2 LITERATURE SURVEY

Online Public Access Catalogs (OPACs) have been an integral part of library services since the 1980s. They have transformed the way people access information, making it easier for library users to search for and locate resources. In recent years, there has been a shift towards electronic resources, and e-libraries have emerged as an important tool for providing access to digital content. In this literature survey, we will explore the development and use of e-library OPACs.

1. Introduction to e-library OPACs: A literature review by H. Chauhan and S. Kaushik (2016) provides an overview of e-library OPACs, including their history, features, and functionality. The authors also discuss the benefits and challenges of using e-library OPACs, such as increased access to digital resources and the need for staff training.
2. User Satisfaction with Library Electronic Resources and Services: A Literature Review by H. C. Chen (2017) examines the factors that influence user satisfaction with e-library OPACs. The author identifies key features such as search functionality, ease of use, and accessibility, and highlights the importance of user-centered design in creating effective e-library OPACs.
3. Improving User Satisfaction with Online Public Access Catalogs (OPACs) through Collaborative Design by C. L. Chuang and T. T. Lai (2018) focuses on the design of e-library OPACs. The authors argue that involving users in the design process can improve user satisfaction, and propose a collaborative design approach that includes feedback from library users.
4. A Study of User Perception of Online Public Access Catalog (OPAC) Systems by M. M. S. Kabir and S. Hossain (2018) examines user perceptions of e-library OPACs in Bangladesh. The authors find that users are generally satisfied with the features and functionality of the OPACs, but note that improvements are needed in areas such as search accuracy and navigation.
5. Enhancing User Experience of Online Public Access Catalog (OPAC) Systems: A Review of Literature by A. D. Musa, A. B. Abubakar, and F. M. Abdullahi (2020) provides an overview of the strategies that can be used to enhance the user experience of e-library OPACs. The authors highlight the importance of user-centered design, personalization, and customization in creating effective OPACs.

### CHAPTER 3

**SYSTEM ARCHITECTURE AND DESIGN**

The system architecture and design for an E-library online public access catalog can be broken down into several key components, including hardware, software, and database.

* **Hardware:** The system may require servers, workstations, network equipment, hard drives, solid-state drives, or network-attached storage (NAS) devices, printers, backup equipment including external hard drives, tapes, and cloud-based backup services.
* **Software:** The software component of the system architecture includes the E-library online public access catalog application, which is responsible for processing and storing library data. This software should be designed to be scalable and adaptable to the specific needs of the organization. The software should also include a user-friendly interface to enable easy access and management of library data.
* **Database:** The database component of the system architecture is responsible for storing and managing the library data. This database should be designed to be highly reliable, scalable, and secure. The database should also be able to handle large volumes of data and provide fast access to library records for reporting and analysis.
* **Reporting:** The reporting module generates reports based on the library data stored in the database. The reporting module should be customizable and should provide various types of reports such as daily library reports, monthly issue reports, and fine reports.
* **Integration:** The integration module allows the library management system to integrate with other systems such as students or staff systems. This module should be designed to be flexible and adaptable to various integration requirements.
* **Security:** To ensure data security, the system should implement various security measures such as data encryption, access controls, and user authentication. The E-library online public access catalog system should only allow authorized users to access the system and the data, and the system should be designed to protect against various types of attacks such as malware, phishing, and hacking.

### CHAPTER 4 METHODOLOGY

The methodology for developing an E-library online public access catalog System typically involves several steps, each of which plays an important role in ensuring the successful development and deployment of the system. The following are the typical steps involved in the development of an E-library online public access catalog:

* **Requirements Gathering:** This is a crucial step in the development process, as it involves identifying the specific needs and requirements of the organization. This can involve conducting interviews with managers, supervisors, and employees, as well as reviewing existing library data to identify areas for improvement. The goal is to gather as much information as possible about the organization's library tracking needs so that the system can be designed to meet those needs.
* **System Design:** Once the requirements have been gathered, the system architecture and design can be developed. This involves creating a detailed plan for the system that outlines the hardware, software, and database components, as well as the user interface, authentication mechanisms, and reporting modules. The system design should be based on the specific needs and requirements of the organization, as identified in the requirements gathering phase.
* **Implementation:** With the system design in place, the system can be developed using programming languages and software tools. This involves writing code to implement the various components of the system, integrating the hardware components such as biometric scanners or RFID card readers, and testing the system as it is being developed. It's important to follow best practices for software development during this phase to ensure that the system is reliable, efficient, and easy to maintain.
* **Testing:** Once the system has been developed, it must be thoroughly tested to ensure that it functions correctly and produces accurate library data. This involves various types of testing, including unit testing, integration testing, and system testing. The goal is to identify any bugs or errors in the system and fix them before the system is deployed.
* **Deployment:** After the system is tested and any bugs have been fixed, it can be deployed in the organization. This involves installing the hardware and software components, configuring the system to meet the specific needs of the organization, and training users on how to use the system. It's important to ensure that the system is fully operational before deploying it in the organization to minimize any disruptions to the organization's operations.
* **Maintenance and Support:** After the system has been deployed, it requires ongoing maintenance and support to ensure that it continues to function correctly. This involves regular updates and bug fixes, as well as user support and training. It's important to have a plan in place for maintaining and supporting the system to ensure that it continues to meet the needs of the organization over time.

### CHAPTER 5

**CODING AND TESTING**

import sqlite3

from tkinter import \* import tkinter.ttk as ttk

import tkinter.messagebox as mb import tkinter.simpledialog as sd

# Connecting to Database

connector = sqlite3.connect('library.db') cursor = connector.cursor()

connector.execute(

'CREATE TABLE IF NOT EXISTS Library (BK\_NAME TEXT, BK\_ID TEXT PRIMARY KEY NOT NULL, AUTHOR\_NAME TEXT, BK\_STATUS TEXT, CARD\_ID TEXT)'

)

# Functions

def issuer\_card():

Cid = sd.askstring('Issuer Card ID', 'What is the Issuer\'s Card ID?\t\t\t') if not Cid:

mb.showerror('Issuer ID cannot be zero!', 'Can\'t keep Issuer ID empty, it must have a value') else:

return Cid

def display\_records():

global connector, cursor global tre

tree.insert('', END, values=records) tree.delete(\*tree.get\_children())

curr = connector.execute('SELECT \* FROM Library') data = curr.fetchall()

for records in data:

def clear\_fields():

global bk\_status, bk\_id, bk\_name, author\_name, card\_id

bk\_status.set('Available')

for i in ['bk\_id', 'bk\_name', 'author\_name', 'card\_id']: exec(f"{i}.set('')") bk\_id\_entry.config(state='normal')

try:

tree.selection\_remove(tree.selection()[0]) except:

pass

def clear\_and\_display(): clear\_fields() display\_records()

def add\_record():

global connector

global bk\_name, bk\_id, author\_name, bk\_status

if bk\_status.get() == 'Issued': card\_id.set(issuer\_card())

else:

card\_id.set('N/A')

surety = mb.askyesno('Are you sure?',

'Are you sure this is the data you want to enter?\nPlease note that Book ID cannot be changed in the future')

if surety:

try:

connector.execute(

'INSERT INTO Library (BK\_NAME, BK\_ID, AUTHOR\_NAME, BK\_STATUS, CARD\_ID) VALUES (?, ?, ?, ?,

?)',

(bk\_name.get(), bk\_id.get(), author\_name.get(), bk\_status.get(), card\_id.get())) connector.commit()

clear\_and\_display()

mb.showinfo('Record added', 'The new record was successfully added to your database') except sqlite3.IntegrityError:

mb.showerror('Book ID already in use!',

'The Book ID you are trying to enter is already in the database, please alter that book\'s record or check any

discrepancies on your side') def view\_record():

global bk\_name, bk\_id, bk\_status, author\_name, card\_id global tree

if not tree.focus():

mb.showerror('Select a row!', 'To view a record, you must select it in the table. Please do so before continuing.') return

current\_item\_selected = tree.focus() values\_in\_selected\_item = tree.item(current\_item\_selected) selection = values\_in\_selected\_item['values']

bk\_name.set(selection[0]) ; bk\_id.set(selection[1]) ; bk\_status.set(selection[3]) author\_name.set(selection[2])

try:

card\_id.set(selection[4]) except:

card\_id.set('') def update\_record():

def update():

global bk\_status, bk\_name, bk\_id, author\_name, card\_id global connector, tree

if bk\_status.get() == 'Issued':

card\_id.set(issuer\_card()) else:

card\_id.set('N/A')

cursor.execute('UPDATE Library SET BK\_NAME=?, BK\_STATUS=?, AUTHOR\_NAME=?, CARD\_ID=? WHERE BK\_ID=?',

(bk\_name.get(), bk\_status.get(), author\_name.get(), card\_id.get(), bk\_id.get())) connector.commit()

clear\_and\_display() edit.destroy()

bk\_id\_entry.config(state='normal')

clear.config(state='normal')

view\_record()

bk\_id\_entry.config(state='disable') clear.config(state='disable')

edit = Button(left\_frame, text='Update Record', font=btn\_font, bg=btn\_hlb\_bg, width=20, command=update) edit.place(x=50, y=375)

def remove\_record():

if not tree.selection():

mb.showerror('Error!', 'Please select an item from the database') return

current\_item = tree.focus() values = tree.item(current\_item) selection = values["values"]

cursor.execute('DELETE FROM Library WHERE BK\_ID=?', (selection[1], ))

connector.commit() tree.delete(current\_item)

mb.showinfo('Done', 'The record you wanted deleted was successfully deleted.') clear\_and\_display()

def delete\_inventory():

if mb.askyesno('Are you sure?', 'Are you sure you want to delete the entire inventory?\n\nThis command cannot be reversed'):

tree.delete(\*tree.get\_children())

cursor.execute('DELETE FROM Library') connector.commit()

else:

return

def change\_availability():

global card\_id, tree, connector

if not tree.selection():

mb.showerror('Error!', 'Please select a book from the database') return

current\_item = tree.focus()

values = tree.item(current\_item) BK\_id = values['values'][1] BK\_status = values["values"][3]

if BK\_status == 'Issued':

surety = mb.askyesno('Is return confirmed?', 'Has the book been returned to you?') if surety:

cursor.execute('UPDATE Library SET bk\_status=?, card\_id=? WHERE bk\_id=?', ('Available', 'N/A', BK\_id)) connector.commit()

else: mb.showinfo(

'Cannot be returned', 'The book status cannot be set to Available unless it has been returned')

else:

cursor.execute('UPDATE Library SET bk\_status=?, card\_id=? where bk\_id=?', ('Issued', issuer\_card(), BK\_id)) connector.commit()

clear\_and\_display() # Variables

lf\_bg = 'LightSkyBlue' # Left Frame Background Color rtf\_bg = 'DeepSkyBlue' # Right Top Frame Background Color

rbf\_bg = 'DodgerBlue' # Right Bottom Frame Background Color btn\_hlb\_bg = 'SteelBlue' # Background color for Head Labels and Buttons

lbl\_font = ('Georgia', 13) # Font for all labels

entry\_font = ('Times New Roman', 12) # Font for all Entry widgets btn\_font = ('Gill Sans MT', 13)

# Initializing the main GUI window root = Tk()

root.title('PythonGeeks Library Management System') root.geometry('1010x530')

root.resizable(0, 0)

Label(root, text='LIBRARY MANAGEMENT SYSTEM', font=("Noto Sans CJK TC", 15, 'bold'), bg=btn\_hlb\_bg, fg='White').pack(side=TOP, fill=X)

# StringVars

bk\_status = StringVar() bk\_name = StringVar() bk\_id = StringVar() author\_name = StringVar() card\_id = StringVar()

# Frames

left\_frame = Frame(root, bg=lf\_bg)

left\_frame.place(x=0, y=30, relwidth=0.3, relheight=0.96) RT\_frame = Frame(root, bg=rtf\_bg) RT\_frame.place(relx=0.3, y=30, relheight=0.2, relwidth=0.7) RB\_frame = Frame(root)

RB\_frame.place(relx=0.3, rely=0.24, relheight=0.785, relwidth=0.7) # Left Frame

Label(left\_frame, text='Book Name', bg=lf\_bg, font=lbl\_font).place(x=98, y=25)

Entry(left\_frame, width=25, font=entry\_font, text=bk\_name).place(x=45, y=55) Label(left\_frame, text='Book ID', bg=lf\_bg, font=lbl\_font).place(x=110, y=105) bk\_id\_entry = Entry(left\_frame, width=25, font=entry\_font, text=bk\_id) bk\_id\_entry.place(x=45, y=135)

Label(left\_frame, text='Author Name', bg=lf\_bg, font=lbl\_font).place(x=90, y=185) Entry(left\_frame, width=25, font=entry\_font, text=author\_name).place(x=45, y=215) Label(left\_frame, text='Status of the Book', bg=lf\_bg, font=lbl\_font).place(x=75, y=265) dd = OptionMenu(left\_frame, bk\_status, \*['Available', 'Issued']) dd.configure(font=entry\_font, width=12)

dd.place(x=75, y=300)

submit = Button(left\_frame, text='Add new record', font=btn\_font, bg=btn\_hlb\_bg, width=20, command=add\_record) submit.place(x=50, y=375)

clear = Button(left\_frame, text='Clear fields', font=btn\_font, bg=btn\_hlb\_bg, width=20, command=clear\_fields) clear.place(x=50, y=435)

# Right Top Frame

Button(RT\_frame, text='Delete book record', font=btn\_font, bg=btn\_hlb\_bg, width=17, command=remove\_record).place(x=8, y=30)

Button(RT\_frame, text='Delete full inventory', font=btn\_font, bg=btn\_hlb\_bg, width=17, command=delete\_inventory).place(x=178, y=30)

Button(RT\_frame, text='Update book details', font=btn\_font, bg=btn\_hlb\_bg, width=17, command=update\_record).place(x=348, y=30)

Button(RT\_frame, text='Change Book Availability', font=btn\_font, bg=btn\_hlb\_bg, width=19, command=change\_availability).place(x=518, y=30)

# Right Bottom Frame

Label(RB\_frame, text='BOOK INVENTORY', bg=rbf\_bg, font=("Noto Sans CJK TC", 15, 'bold')).pack(side=TOP, fill=X) tree = ttk.Treeview(RB\_frame, selectmode=BROWSE, columns=('Book Name', 'Book ID', 'Author', 'Status', 'Issuer Card ID'))

XScrollbar = Scrollbar(tree, orient=HORIZONTAL, command=tree.xview) YScrollbar = Scrollbar(tree, orient=VERTICAL, command=tree.yview) XScrollbar.pack(side=BOTTOM, fill=X)

YScrollbar.pack(side=RIGHT, fill=Y) tree.config(xscrollcommand=XScrollbar.set, yscrollcommand=YScrollbar.set)

tree.heading('Book Name', text='Book Name', anchor=CENTER) tree.heading('Book ID', text='Book ID', anchor=CENTER) tree.heading('Author', text='Author', anchor=CENTER) tree.heading('Status', text='Status of the Book', anchor=CENTER)

tree.heading('Issuer Card ID', text='Card ID of the Issuer', anchor=CENTER)

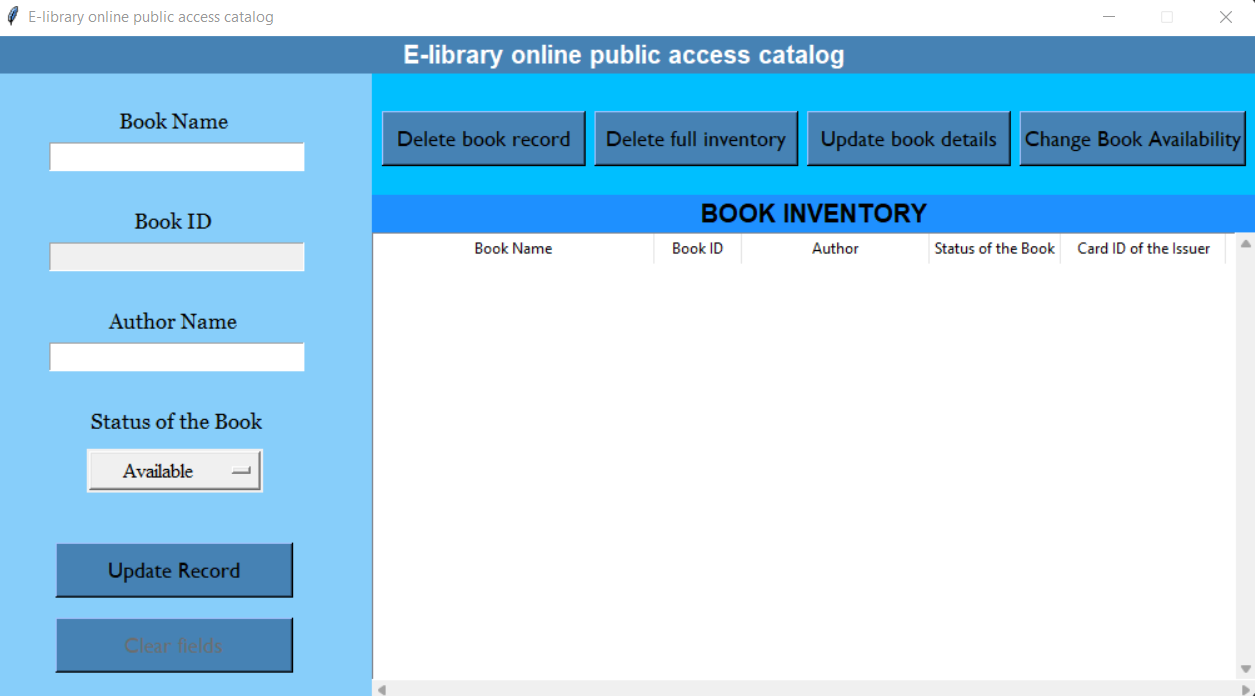
tree.column('#0', width=0, stretch=NO) tree.column('#1', width=225, stretch=NO) tree.column('#2', width=70, stretch=NO) tree.column('#3', width=150, stretch=NO) tree.column('#4', width=105, stretch=NO) tree.column('#5', width=132, stretch=NO)

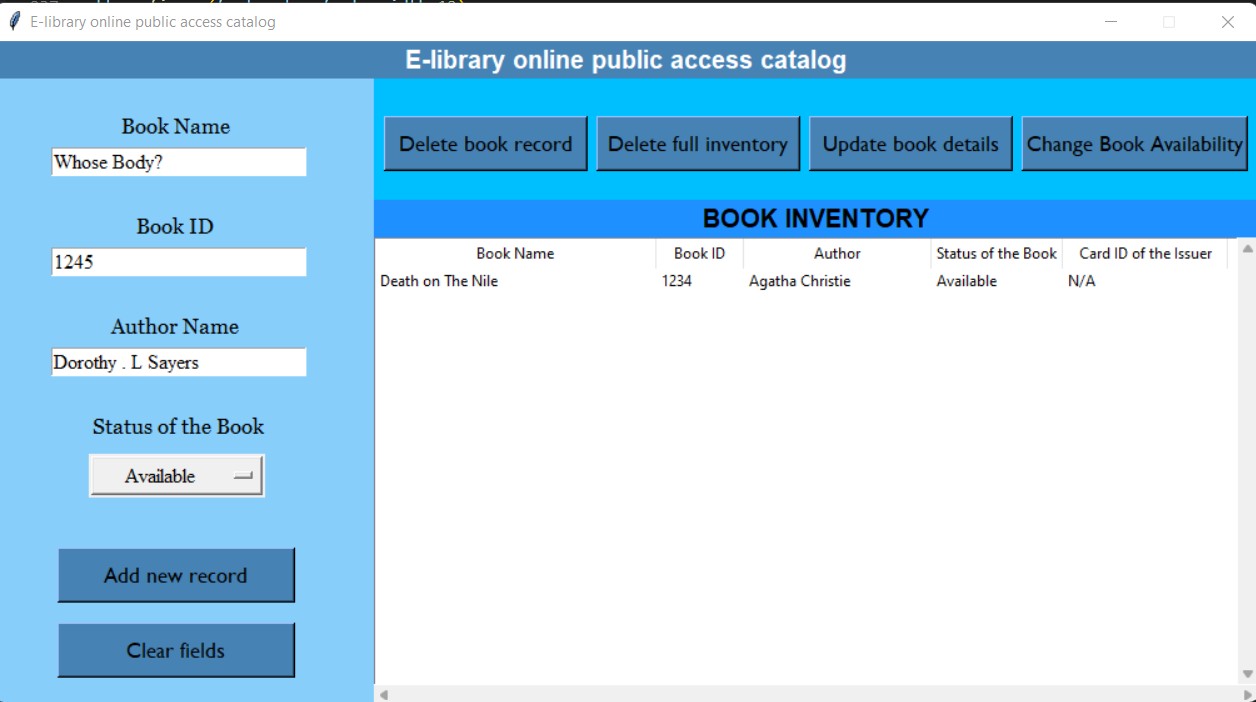
tree.place(y=30, x=0, relheight=0.9, relwidth=1) clear\_and\_display()

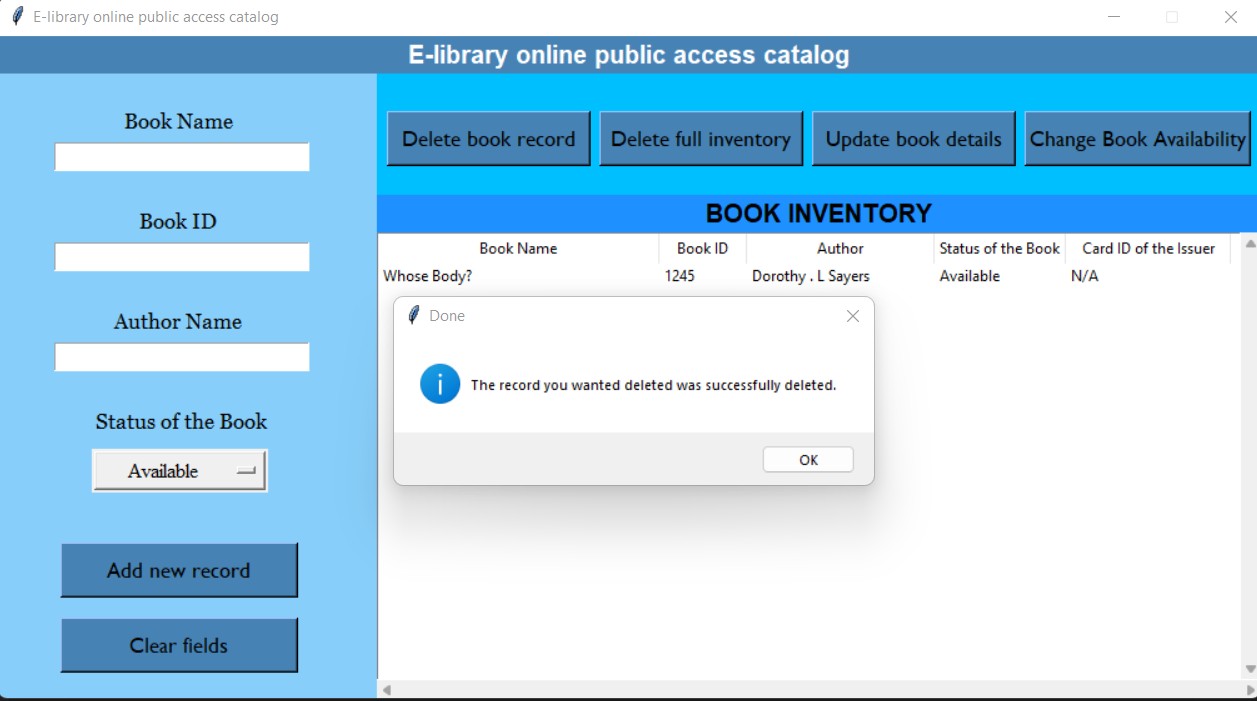
# Finalizing the window root.update() root.mainloop()

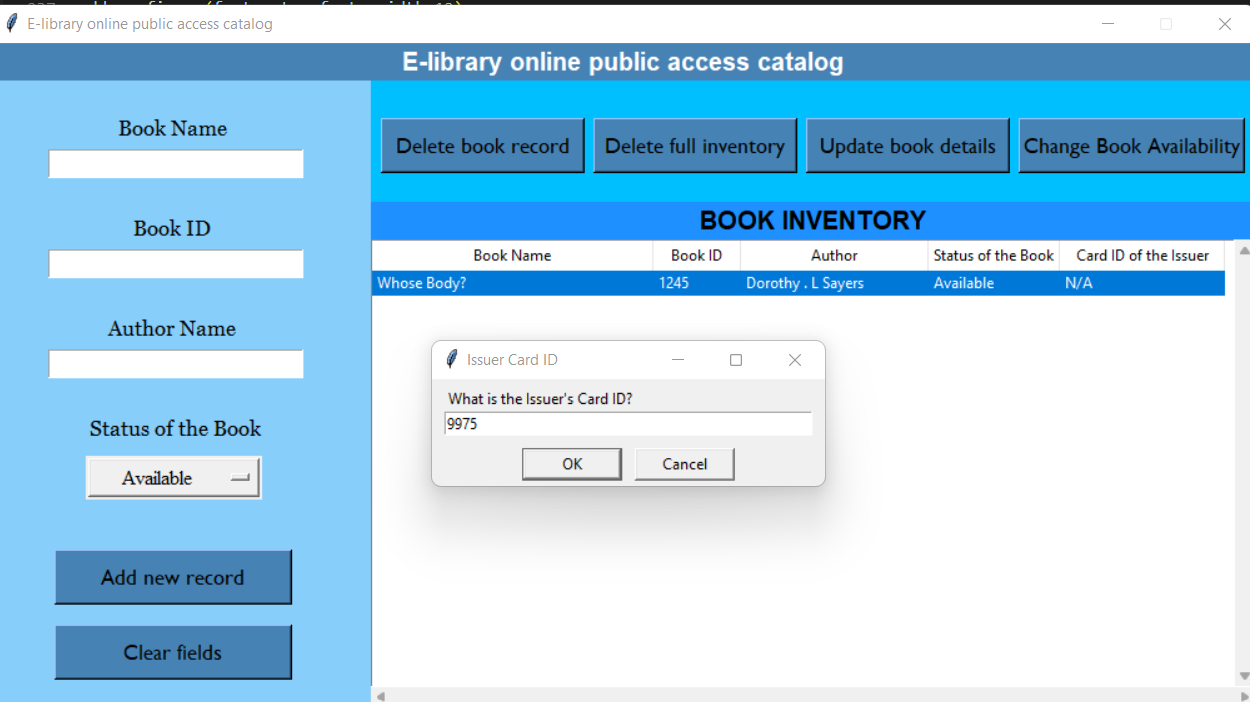
### CHAPTER 6

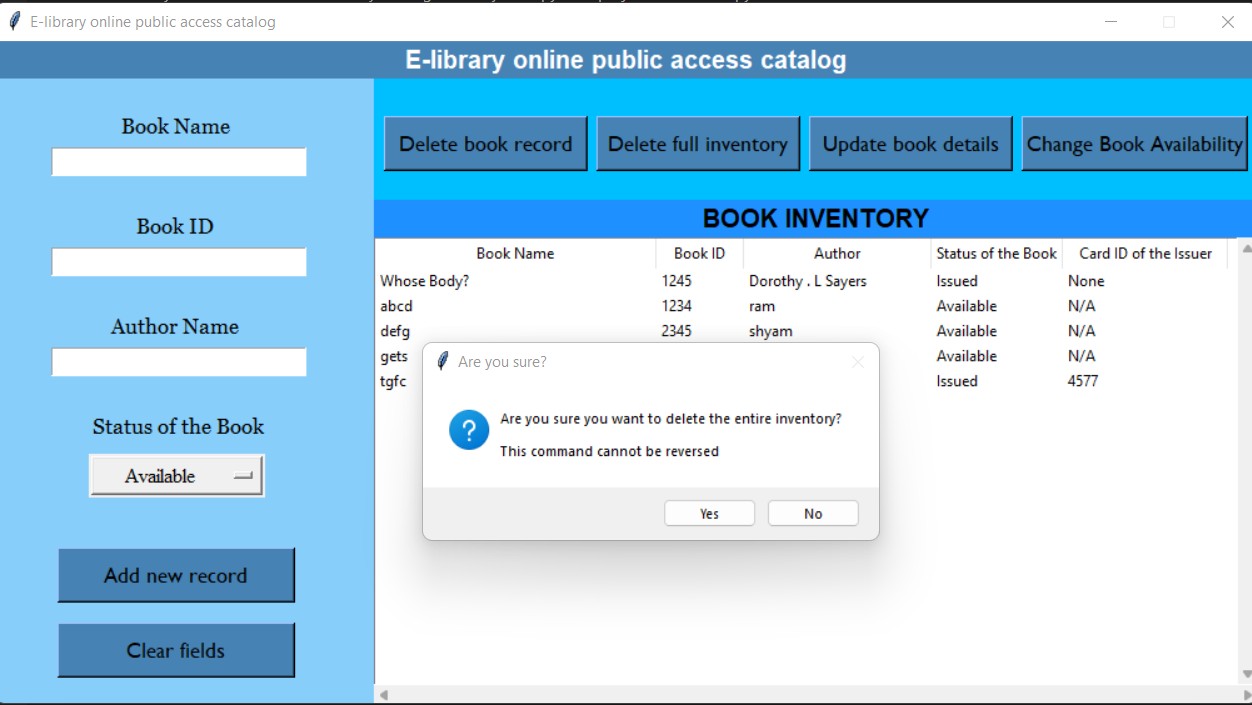
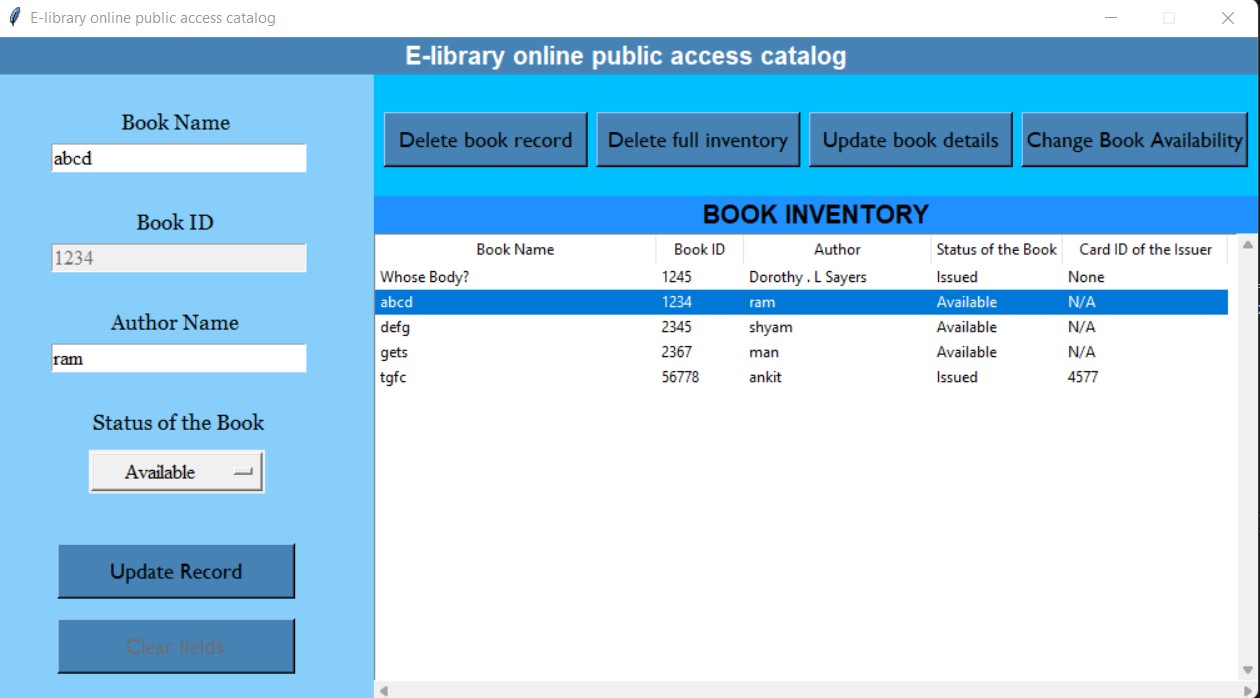
**SCREENSHOTS AND RESULTS**











### CHAPTER 7

**CONCLUSION AND FUTURE ENHANCEMENTS**

In conclusion, this project aimed to demonstrate the implementation of a simple E-library online public access catalog using Python. The project achieved its objective by utilizing Python's built-in data structures such as dictionaries and functions to manage library system.

E-library online public access catalog with the features of adding, updating, deleting records, changing record status from available to issued, and deleting all records is an essential tool for modern libraries. These features allow librarians to manage their collections efficiently, making it easier for users to find and borrow books. The ability to add, update, and delete records ensures that the catalog is up-to-date and accurate, while the ability to change the status of records helps librarians keep track of book availability. However, it's important to remember that these features must be used with care, and proper training and safeguards must be in place to prevent accidental or unauthorized changes to the catalog.

Overall, this project serves as an excellent example of how Python can be used to develop solutions that automate routine tasks, making them faster, more efficient, and less prone to errors. With a few modifications, this project can be adapted to suit different contexts.

### REFERENCES

* 1. "KOHA: An Open-Source Integrated Library System" by Chris Cormack and Nicole C. Engard (ISBN: 978-1-4419-7140-6)
  2. "The Open Source Library System: A Resource Guide for Libraries" by Emily Lynema and Joy Nelson (ISBN: 978-1-61069-529-7)
  3. "Library Management System using PHP and MySQL" by Prashant Kukreja and Vikas Gupta (ISBN: 978-3-8473-0455-1)
  4. "Integrated Library Systems: Planning, Selecting, and Implementing" by John W. Head and Gerard B. McCabe (ISBN: 978-0-8389-1204-4)
  5. "Library Automation: Core Concepts and Practical Systems Analysis" by Dania Bilal (ISBN: 978-1-931666-40-9)