Computer Science Project

ON:

Library Management

Submitted by
Adarsh Sarathy
(Roll no.)

INDEX

★ Certificate
★ Acknowledgement
★ Introduction
★ Files generated
★ Preface
★ Flowchart
★ Hardware and software requirements
★ Source Code (Front end and Back end)
★ Output
★ Future Scope
★ Bibliography

CERTIFICATE

This is to certify that ADARSH SARATHY (Roll No._____) of Hillwoods School has successfully completed his project in Computer Science on the topic "LIBRARY MANAGEMENT" as prescribed by CBSE in the academic session 2022-23.

SIGNATURE OF TEACHER

SIGNATURE OF HEAD OF SCHOOL

SIGNATURE OF EXAMINER

SCHOOL STAMP

Acknowledgement

I would like to express my special gratitude to our wonderful teacher Mrs. Khushboo Gupta for giving me this delightful opportunity to do this project on the topic of "Library Management" and also guiding me through the process of making it. I would also like to express my gratitude to my parents and project partner who made this possible. Also I would like to thank the internet for making the painstaking process of researching the topic a bit easier and fun. Also a special thanks to our librarian for her constant support.

INTRODUCTION

This project is a successor to the current manual library management system in our school. The main objective for this management system is to make the management of the library more automated, functional and effective. The librarian can keep a maintained record of the books present in the library by being able to ADD BOOKS, MODIFY BOOK DATA, DELETE BOOK DATA and SEARCH FOR PARTICULAR BOOKS on the basis of their BOOK NAME, AUTHOR NAME, AUTHOR CODE or ACCESSION NUMBER.

Under book information four functions are available:

1. Add Book Data
2. Search Book Data
3. Modify Book Data
4. Delete Book Data

It is designed in a very user-friendly manner, simplifying the tedious manual work of keeping the records of the thousands of books in our library.

The concepts of strings, lists, dictionaries, inbuilt functions, user defined functions, tkinter library, MySQL (Database) connectivity (mysql connector module), csv connectivity have been implemented in the making of the code.

FILES GENERATED

PROGRAM FILE

Library_Management_System.py

BookInfo.csv

BookInfo_sql.py

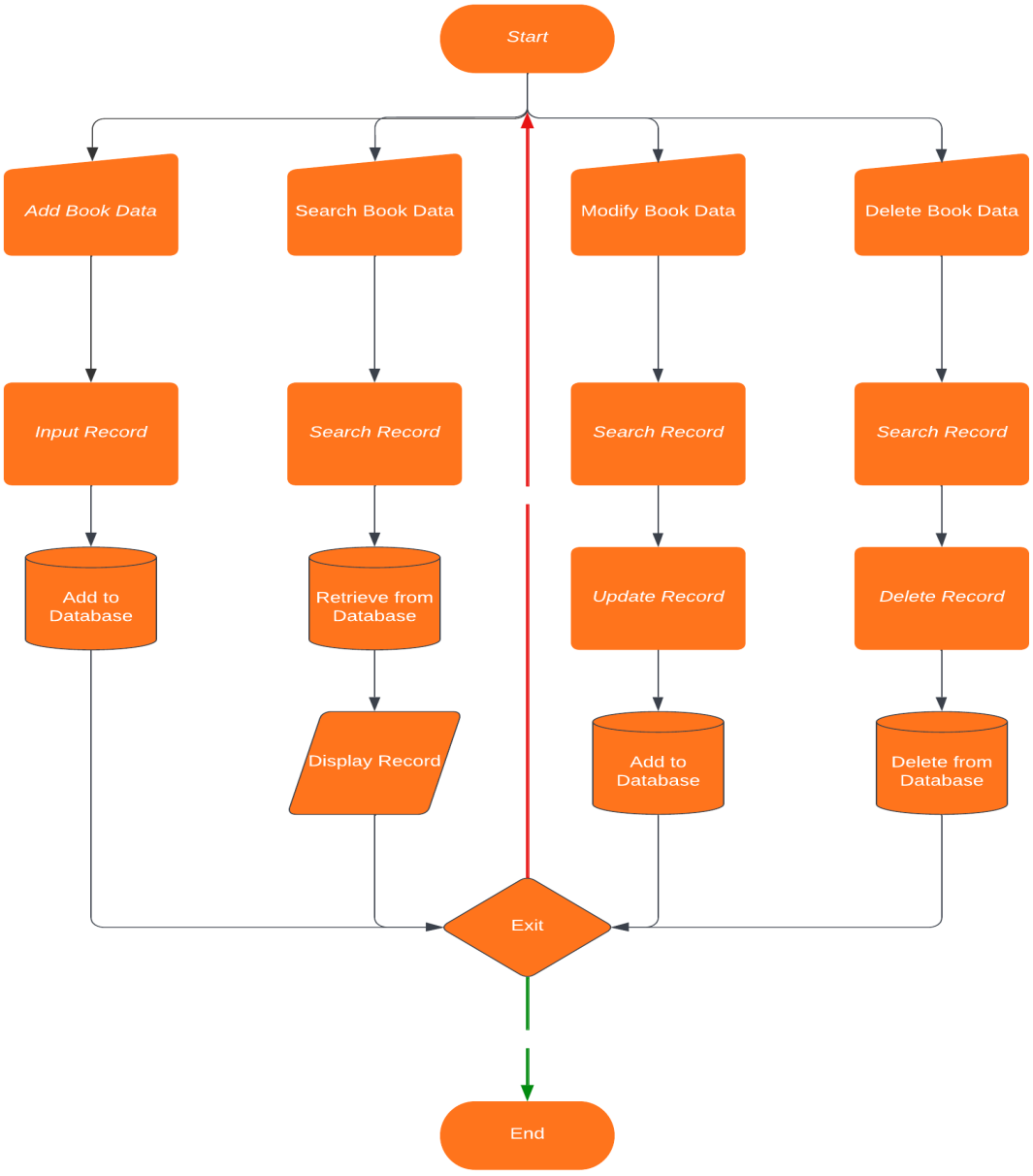
EXECUTION FILE

Library_Management_System.exe

PREFACE

This software is used to maintain book records and to maintain the Library in an efficient manner by recording the data of the books. The objective of this project is to apply the textbook knowledge of programming into a real world situation. Careful oversight was required to ensure that the project supports strategic objectives and resources be effectively implemented to have a smooth flow.

FLOW CHART



HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE:

I. OPERATING SYSTEM: WINDOWS 10 AND ABOVE
II. PROCESSOR: PENTIUM(ANY) OR AMD ATHALON(3800+-4200+ DUAL CORE)
III. MOTHERBOARD: 1.845 OR 915,995 FOR PENTIUM OR MSI K9MM-V VIA K8M800+8237R PLUS CHIPSET FOR AMD ATHALON
IV. RAM: 512MB+
V. Hard disk: SATA 40GB OR ABOVE
VI. MONITOR
VII. KEY BOARD
VIII. MOUSE

SOFTWARE:

I. Windows OS
II. Python (3.8.5)
III. MySQL

SOURCE CODE

FRONT END

```
import time
import tkinter
import mysql.connector as mc
from tkinter import *
from tkinter import ttk


mydb = mc.connect(host = 'localhost', user = 'libmansys', passwd = 'libmansys',
database = 'lib')
mycursor = mydb.cursor()


#Main Window
window = Tk()
tab_window = ttk.Notebook(window)
global accno
#########################################################################
#########################################################################
################################
#Add Data Tab
add_data_tab = Frame(tab_window)
def add_func():
    # add_data_tab.grid(row= 0, column= 0, padx=10, pady= (5,20))
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #LabelFrame to store accno and classno
   library_info_frame = LabelFrame(add_data_tab, text='Library Information',
padx= 20, pady= 15)
   library_info_frame.grid(row= 0, column= 0, columnspan= 2, padx= 20, pady=
(45,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store accno
   acc_no_frame = Frame(library_info_frame)
   acc_no_frame.grid(row=0,column=0,padx=(0,7.5))
   acc_no_label = Label(acc_no_frame, text= 'Accession
Number').grid(row=0,column=0)
   mycursor.execute('select accno from library')
   accno = [i for i in mycursor]
   acc_no_entry = Entry(acc_no_frame)
   acc_no_entry.insert(END,int(accno[-1][0])+1)
   acc_no_entry.config(state= DISABLED)
   acc_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store classno
   class_no_frame = Frame(library_info_frame)
   class_no_frame.grid(row=0,column=1,padx=(7.5,0))
   class_no_label = Label(class_no_frame, text= 'Class
Number').grid(row=0,column=0)
```

```
    class_no_entry = Entry(class_no_frame)
    class_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #LabelFrame to store book info
    book_info_frame = LabelFrame(add_data_tab, text='Book Information', padx=
20, pady= 15)
    book_info_frame.grid(row= 1, column= 0, columnspan= 2, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store title
    title_frame = Frame(book_info_frame)
    title_frame.grid(row= 0, column= 0, padx= (0,7.5))
    title_label = Label(title_frame, text= 'Title').grid(row= 0, column= 0)
    title_entry = Entry(title_frame)
    title_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store author name
    author_name_frame = Frame(book_info_frame)
    author_name_frame.grid(row= 1, column= 0, padx= (0,7.5), pady= (5,0))
    author_name_label = Label(author_name_frame, text= 'Author
Name').grid(row= 0, column= 0)
    author_name_entry = Entry(author_name_frame)
    author_name_entry.grid(row= 1, column= 0)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store author initials
   author_in_frame = Frame(book_info_frame)
   author_in_frame.grid(row= 1, column= 1, padx= (7.5,0), pady= (5,0))
   author_in_label = Label(author_in_frame, text= 'Author Initial').grid(row= 0,
column= 0)
   author_in_entry = Entry(author_in_frame)
   author_in_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store number of pages
   pages_frame = Frame(book_info_frame)
   pages_frame.grid(row= 0, column= 1, padx= (7.5,0))
   pages_label = Label(pages_frame, text= 'Pages').grid(row= 0, column= 0)
   pages_entry = Entry(pages_frame)
   pages_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #LabelFrame to store publisher info
   publisher_info_frame = LabelFrame(add_data_tab, text='Publisher
Information', padx= 20, pady= 15)
   publisher_info_frame.grid(row= 2, column=0, columnspan= 2, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
```

```
    #Frame to store publisher name
    publisher_name_frame = Frame(publisher_info_frame)
    publisher_name_frame.grid(row= 0, column= 0, padx= (0,7.5))
    publisher_name_label = Label(publisher_name_frame, text= 'Publisher
Name').grid(row= 0, column= 0)
    publisher_name_entry = Entry(publisher_name_frame)
    publisher_name_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #Frame to store publisher year
    publisher_year_frame = Frame(publisher_info_frame)
    publisher_year_frame.grid(row= 0, column= 1, padx= (7.5,0))
    publisher_year_label = Label(publisher_year_frame, text= 'Publishing
Year').grid(row= 0, column= 0)
    publisher_year_combobox = ttk.Combobox(publisher_year_frame, values=[i for
i in range(1750,time.localtime().tm_year+1)])
    publisher_year_combobox.current(time.localtime().tm_year-1750)
    publisher_year_combobox.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #LabelFrame to store cost & copies info
    cost_copies_info_frame = LabelFrame(add_data_tab, text='Cost & Copies',
padx= 20, pady= 15)
    cost_copies_info_frame.grid(row= 3, column= 0, columnspan= 2, padx= 20,
pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
```

```
    #Frame to store cost
    cost_frame = Frame(cost_copies_info_frame)
    cost_frame.grid(row= 0, column= 0, padx= (0,7.5))
    cost_label = Label(cost_frame, text= 'Cost').grid(row= 0, column= 0)
    cost_entry = Entry(cost_frame)
    cost_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store copies
    copies_frame = Frame(cost_copies_info_frame)
    copies_frame.grid(row= 0, column= 1, padx= (7.5,0))
    copies_label = Label(copies_frame, text= 'Copies').grid(row= 0, column= 0)
    var = IntVar(copies_frame)
    copies_spinbox = ttk.Spinbox(copies_frame, from_=1, to='infinity', textvariable=
var)
    copies_spinbox.grid(row= 1, column=0)
    var.set(1)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Button to Cancel
    def cancel():
        class_no_entry.delete(0,END)
        title_entry.delete(0,END)
        pages_entry.delete(0,END)
        author_name_entry.delete(0,END)
        author_in_entry.delete(0,END)
        publisher_name_entry.delete(0,END)
        publisher_year_combobox.current(time.localtime().tm_year-1750)
```

```
        cost_entry.delete(0,END)
        var.set(1)
    cancel_button = Button(add_data_tab, text= 'Cancel', command= cancel)
    cancel_button.grid(row= 4, column= 0, padx= (100,10), pady= (10,10))

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Button to Add
    global click
    click = 1
    def add():
        global click
        mycursor.execute(f"insert into library
value{(acc_no_entry.get(),class_no_entry.get(),title_entry.get(),pages_entry.get(),aut
hor_name_entry.get(),author_in_entry.get(),publisher_name_entry.get(),publisher_
year_combobox.get(),cost_entry.get(),copies_spinbox.get())}")
        mydb.commit()
        click+=1
        cancel()
        acc_no_entry.config(state= NORMAL)
        acc_no_entry.delete(0,END)
        acc_no_entry.insert(END,int(accno[-1][0])+click)
        acc_no_entry.config(state= DISABLED)
    add_button = Button(add_data_tab, text= 'Add Data', command= add)
    add_button.grid(row= 4, column= 1, padx= (10,100), pady= (10,10))
#########################################################################
#########################################################################
#################################
#Search Data Tab
search_data_tab = Frame(tab_window)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
#search by field frame
def search_func():
    global field,criteria
    search_field_frame = LabelFrame(search_data_tab, text= 'Search By', padx= 20,
pady= 15)
    search_field_frame.grid(row= 0, column= 0, columnspan= 3, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #search by field
    values = ['Accession Number','Class Number','Book Name','Author
Name','Author Initials','Publisher Name','Publishing Year','Cost','Copies']
    search_label = Label(search_field_frame, text= 'Search By:').grid(row= 0,
column= 0, padx= (0,15))
    search_combobox = ttk.Combobox(search_field_frame, values= values)
    search_combobox.grid(row= 0, column= 1, padx= (0,15))
    def go():
        global field
        global criteria
        criteria = ''
        field = ''
        if search_combobox.get() == values[0]:
            _acc_no_entry.config(state= NORMAL)
            field = 'accno'
        elif search_combobox.get() == values[1]:
            _class_no_entry.config(state= NORMAL)
            criteria = _class_no_entry.get()
```

```
    field = 'classno'
elif search_combobox.get() == values[2]:
    _title_entry.config(state= NORMAL)
    criteria = _title_entry.get()
    field = 'title'
elif search_combobox.get() == values[3]:
    _author_name_entry.config(state= NORMAL)
    criteria = _author_name_entry.get()
    field = 'author'
elif search_combobox.get() == values[4]:
    _author_in_entry.config(state= NORMAL)
    criteria = _author_in_entry.get()
    field = 'authorin'
elif search_combobox.get() == values[5]:
    _publisher_name_entry.config(state= NORMAL)
    criteria = _publisher_name_entry.get()
    field = 'publisher'
elif search_combobox.get() == values[6]:
    _publisher_year_combobox.config(state= NORMAL)
    criteria = _publisher_year_combobox.get()
    field = 'year'
elif search_combobox.get() == values[7]:
    _cost_entry.config(state= NORMAL)
    criteria = _cost_entry.get()
    field = 'netamount'
elif search_combobox.get() == values[8]:
    _copies_spinbox.config(state= NORMAL)
    criteria = _copies_spinbox.get()
    field = 'copies'
# return field, criteria
```

```
    go_button = Button(search_field_frame, text= 'Go', command= go, width= 6)
    go_button.grid(row= 0, column= 2)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Search Data
    #LabelFrame to store accno and classno
    _library_info_frame = LabelFrame(search_data_tab, text='Library Information',
padx= 20, pady= 15)
    _library_info_frame.grid(row= 1, column= 0, columnspan= 2, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store accno
    _acc_no_frame = Frame(_library_info_frame)
    _acc_no_frame.grid(row=0,column=0,padx=(0,7.5))
    _acc_no_label = Label(_acc_no_frame, text= 'Accession
Number').grid(row=0,column=0)
    _acc_no_entry = Entry(_acc_no_frame)
    _acc_no_entry.config(state= DISABLED)
    _acc_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store classno
    _class_no_frame = Frame(_library_info_frame)
    _class_no_frame.grid(row=0,column=1,padx=(7.5,0))
    _class_no_label = Label(_class_no_frame, text= 'Class
Number').grid(row=0,column=0)
```

```
    _class_no_entry = Entry(_class_no_frame, state= DISABLED)
    _class_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #LabelFrame to store book info
    _book_info_frame = LabelFrame(search_data_tab, text='Book Information',
padx= 20, pady= 15)
    _book_info_frame.grid(row= 2, column= 0, columnspan= 2, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store title
    _title_frame = Frame(_book_info_frame)
    _title_frame.grid(row= 0, column= 0, padx= (0,7.5))
    _title_label = Label(_title_frame, text= 'Title').grid(row= 0, column= 0)
    _title_entry = Entry(_title_frame, state= DISABLED)
    _title_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store author name
    _author_name_frame = Frame(_book_info_frame)
    _author_name_frame.grid(row= 1, column= 0, padx= (0,7.5), pady= (5,0))
    _author_name_label = Label(_author_name_frame, text= 'Author
Name').grid(row= 0, column= 0)
    _author_name_entry = Entry(_author_name_frame, state= DISABLED)
    _author_name_entry.grid(row= 1, column= 0)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store author initials
   _author_in_frame = Frame(_book_info_frame)
   _author_in_frame.grid(row= 1, column= 1, padx= (7.5,0), pady= (5,0))
   _author_in_label = Label(_author_in_frame, text= 'Author Initial').grid(row= 0,
column= 0)
   _author_in_entry = Entry(_author_in_frame, state= DISABLED)
   _author_in_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store number of pages
   _pages_frame = Frame(_book_info_frame)
   _pages_frame.grid(row= 0, column= 1, padx= (7.5,0))
   _pages_label = Label(_pages_frame, text= 'Pages').grid(row= 0, column= 0)
   _pages_entry = Entry(_pages_frame, state= DISABLED)
   _pages_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #LabelFrame to store publisher info
   _publisher_info_frame = LabelFrame(search_data_tab, text='Publisher
Information', padx= 20, pady= 15)
   _publisher_info_frame.grid(row= 3, column=0, columnspan= 2, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
```

```
    #Frame to store publisher name
    _publisher_name_frame = Frame(_publisher_info_frame)
    _publisher_name_frame.grid(row= 0, column= 0, padx= (0,7.5))
    _publisher_name_label = Label(_publisher_name_frame, text= 'Publisher
Name').grid(row= 0, column= 0)
    _publisher_name_entry = Entry(_publisher_name_frame, state= DISABLED)
    _publisher_name_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store publisher year
    _publisher_year_frame = Frame(_publisher_info_frame)
    _publisher_year_frame.grid(row= 0, column= 1, padx= (7.5,0))
    _publisher_year_label = Label(_publisher_year_frame, text= 'Publishing
Year').grid(row= 0, column= 0)
    _publisher_year_combobox = ttk.Combobox(_publisher_year_frame, values=[i
for i in range(1750,time.localtime().tm_year+1)], state= DISABLED)
    _publisher_year_combobox.current(time.localtime().tm_year-1750)
    _publisher_year_combobox.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #LabelFrame to store cost & copies info
    _cost_copies_info_frame = LabelFrame(search_data_tab, text='Cost & Copies',
padx= 20, pady= 15)
    _cost_copies_info_frame.grid(row= 4, column= 0, columnspan= 2, padx= 20,
pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
```

```
    #Frame to store cost
    _cost_frame = Frame(_cost_copies_info_frame)
    _cost_frame.grid(row= 0, column= 0, padx= (0,7.5))
    _cost_label = Label(_cost_frame, text= 'Cost').grid(row= 0, column= 0)
    _cost_entry = Entry(_cost_frame, state= DISABLED)
    _cost_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store copies
    _copies_frame = Frame(_cost_copies_info_frame)
    _copies_frame.grid(row= 0, column= 1, padx= (7.5,0))
    _copies_label = Label(_copies_frame, text= 'Copies').grid(row= 0, column= 0)
    _var = IntVar(_copies_frame)
    _copies_spinbox = ttk.Spinbox(_copies_frame, from_=1, to='infinity',
textvariable= _var, state= DISABLED)
    _copies_spinbox.grid(row= 1, column=0)
    _var.set(1)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    def search():
        global dictionary,field
        dictionary = {'accno' : _acc_no_entry, 'classno' : _class_no_entry, 'title' :
_title_entry, 'author' : _author_name_entry, 'authorin' : _author_in_entry,
'publisher' : _publisher_name_entry, 'year' : _publisher_year_combobox,
'netamount' : _cost_entry, 'copies' : _var}
        mycursor.execute(f"select * from library where {field} =
'{dictionary[field].get()}'")
        dictionary[field].delete(0,END)
```

```
        data_all = [i for i in mycursor]
        data = data_all[0]
        _acc_no_entry.config(state= NORMAL)
        _class_no_entry.config(state= NORMAL)
        _title_entry.config(state= NORMAL)
        _pages_entry.config(state= NORMAL)
        _author_name_entry.config(state= NORMAL)
        _author_in_entry.config(state= NORMAL)
        _publisher_name_entry.config(state= NORMAL)
        _publisher_year_combobox.config(state= NORMAL)
        _cost_entry.config(state= NORMAL)
        _acc_no_entry.insert(0,data[0])
        _class_no_entry.insert(0,data[1])
        _title_entry.insert(0,data[2])
        _pages_entry.insert(0,data[3])
        _author_name_entry.insert(0,data[4])
        _author_in_entry.insert(0,data[5])
        _publisher_name_entry.insert(0,data[6])
        _publisher_year_combobox.current(int(data[7])-1750)
        _cost_entry.insert(0,data[8])
        try:
            _copies_spinbox.config(state= NORMAL)
            _var.set(int(data[9]))
        except:
            pass

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    def reset():
        global dictionary, field
```

```
        dictionary = {'accno' : _acc_no_entry, 'classno' : _class_no_entry, 'title' :
_title_entry, 'author' : _author_name_entry, 'authorin' : _author_in_entry,
'publisher' : _publisher_name_entry, 'year' : _publisher_year_combobox,
'netamount' : _cost_entry, 'copies' : _var}
        dictionary[field].delete(0,END)
        dictionary[field].config(state= DISABLED)
        search_combobox.delete(0,END)
        _acc_no_entry.delete(0,END)
        _class_no_entry.delete(0,END)
        _title_entry.delete(0,END)
        _pages_entry.delete(0,END)
        _author_name_entry.delete(0,END)
        _author_in_entry.delete(0,END)
        _publisher_name_entry.delete(0,END)
        _publisher_year_combobox.current(time.localtime().tm_year-1750)
        _cost_entry.delete(0,END)
        _var.set(1)
        _acc_no_entry.config(state= DISABLED)
        _class_no_entry.config(state= DISABLED)
        _title_entry.config(state= DISABLED)
        _pages_entry.config(state= DISABLED)
        _author_name_entry.config(state= DISABLED)
        _author_in_entry.config(state= DISABLED)
        _publisher_name_entry.config(state= DISABLED)
        _publisher_year_combobox.config(state= DISABLED)
        _cost_entry.config(state= DISABLED)
        _copies_spinbox.config(state= DISABLED)
    reset_button = Button(search_data_tab, text= 'Reset', command= reset)
    reset_button.grid(row= 5, column= 0, padx= (100,10), pady=(10,10))
```

```
    search_button = Button(search_data_tab, text= 'Search', command= search)
    search_button.grid(row= 5, column= 1, padx= (10,100), pady=(10,10))
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
#Search Data Tab
modify_data_tab = Frame(tab_window)
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
#__search by _field frame
def modify_func():
    global _field
    __search_field_frame = LabelFrame(modify_data_tab, text= 'Search By', padx=
20, pady= 15)
    __search_field_frame.grid(row= 0, column= 0, columnspan= 3, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #__search by _field
    __values = ['Accession Number','Class Number','Book Name','Author
Name','Author Initials','Publisher Name','Publishing Year','Cost','Copies']
    __search_label = Label(__search_field_frame, text= 'Search By:').grid(row= 0,
column= 0, padx= (0,15))
    __search_combobox = ttk.Combobox(__search_field_frame, values= __values)
    __search_combobox.grid(row= 0, column= 1, padx= (0,15))
    def __go():
        global _field
        _field = ''
        if __search_combobox.get() == __values[0]:
            __acc_no_entry.config(state= NORMAL)
```

```
            _field = 'accno'
        elif __search_combobox.get() == __values[1]:
            __class_no_entry.config(state= NORMAL)
            _field = 'classno'
        elif __search_combobox.get() == __values[2]:
            __title_entry.config(state= NORMAL)
            _field = 'title'
        elif __search_combobox.get() == __values[3]:
            __author_name_entry.config(state= NORMAL)
            _field = 'author'
        elif __search_combobox.get() == __values[4]:
            __author_in_entry.config(state= NORMAL)
            _field = 'authorin'
        elif __search_combobox.get() == __values[5]:
            __publisher_name_entry.config(state= NORMAL)
            _field = 'publisher'
        elif __search_combobox.get() == __values[6]:
            __publisher_year_combobox.config(state= NORMAL)
            _field = 'year'
        elif __search_combobox.get() == __values[7]:
            __cost_entry.config(state= NORMAL)
            _field = 'netamount'
        elif __search_combobox.get() == __values[8]:
            __copies_spinbox.config(state= NORMAL)
            _field = 'copies'
        # return _field, criteria
    __go_button = Button(__search_field_frame, text= 'Go', command= __go,
width= 6)
    __go_button.grid(row= 0, column= 2)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Search Data
   #LabelFrame to store accno and classno
   __library_info_frame = LabelFrame(modify_data_tab, text='Library
Information', padx= 20, pady= 15)
   __library_info_frame.grid(row= 1, column= 0, columnspan= 3, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store accno
   __acc_no_frame = Frame(__library_info_frame)
   __acc_no_frame.grid(row=0,column=0,padx=(0,7.5))
   __acc_no_label = Label(__acc_no_frame, text= 'Accession
Number').grid(row=0,column=0)
   __acc_no_entry = Entry(__acc_no_frame)
   __acc_no_entry.config(state= DISABLED)
   __acc_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store classno
   __class_no_frame = Frame(__library_info_frame)
   __class_no_frame.grid(row=0,column=1,padx=(7.5,0))
   __class_no_label = Label(__class_no_frame, text= 'Class
Number').grid(row=0,column=0)
   __class_no_entry = Entry(__class_no_frame, state= DISABLED)
   __class_no_entry.grid(row=1,column=0)
```

```
#-----------------------------------------------------------------------------------------------------------------------------#
   #LabelFrame to store book info
   __book_info_frame = LabelFrame(modify_data_tab, text='Book Information', padx= 20, pady= 15)
   __book_info_frame.grid(row= 2, column= 0, columnspan= 3, padx= 20, pady= (10,0), sticky= 'news')

#-----------------------------------------------------------------------------------------------------------------------------#
   #Frame to store title
   __title_frame = Frame(__book_info_frame)
   __title_frame.grid(row= 0, column= 0, padx= (0,7.5))
   __title_label = Label(__title_frame, text= 'Title').grid(row= 0, column= 0)
   __title_entry = Entry(__title_frame, state= DISABLED)
   __title_entry.grid(row= 1, column= 0)

#-----------------------------------------------------------------------------------------------------------------------------#
   #Frame to store author name
   __author_name_frame = Frame(__book_info_frame)
   __author_name_frame.grid(row= 1, column= 0, padx= (0,7.5), pady= (5,0))
   __author_name_label = Label(__author_name_frame, text= 'Author Name').grid(row= 0, column= 0)
   __author_name_entry = Entry(__author_name_frame, state= DISABLED)
   __author_name_entry.grid(row= 1, column= 0)

#-----------------------------------------------------------------------------------------------------------------------------#
```

```
    #Frame to store author initials
    __author_in_frame = Frame(__book_info_frame)
    __author_in_frame.grid(row= 1, column= 1, padx= (7.5,0), pady= (5,0))
    __author_in_label = Label(__author_in_frame, text= 'Author Initial').grid(row=
0, column= 0)
    __author_in_entry = Entry(__author_in_frame, state= DISABLED)
    __author_in_entry.grid(row= 1, column= 0)

#-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------#
    #Frame to store number of pages
    __pages_frame = Frame(__book_info_frame)
    __pages_frame.grid(row= 0, column= 1, padx= (7.5,0))
    __pages_label = Label(__pages_frame, text= 'Pages').grid(row= 0, column= 0)
    __pages_entry = Entry(__pages_frame, state= DISABLED)
    __pages_entry.grid(row= 1, column=0)

#-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------#
    #LabelFrame to store publisher info
    __publisher_info_frame = LabelFrame(modify_data_tab, text='Publisher
Information', padx= 20, pady= 15)
    __publisher_info_frame.grid(row= 3, column=0, columnspan= 3, padx= 20,
pady= (10,0), sticky= 'news')

#-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------#
    #Frame to store publisher name
    __publisher_name_frame = Frame(__publisher_info_frame)
    __publisher_name_frame.grid(row= 0, column= 0, padx= (0,7.5))
```

```
    __publisher_name_label = Label(__publisher_name_frame, text= 'Publisher
Name').grid(row= 0, column= 0)
    __publisher_name_entry = Entry(__publisher_name_frame, state= DISABLED)
    __publisher_name_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store publisher year
    __publisher_year_frame = Frame(__publisher_info_frame)
    __publisher_year_frame.grid(row= 0, column= 1, padx= (7.5,0))
    __publisher_year_label = Label(__publisher_year_frame, text= 'Publishing
Year').grid(row= 0, column= 0)
    __publisher_year_combobox = ttk.Combobox(__publisher_year_frame,
values=[i for i in range(1750,time.localtime().tm_year+1)], state= DISABLED)
    __publisher_year_combobox.current(time.localtime().tm_year-1750)
    __publisher_year_combobox.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #LabelFrame to store cost & copies info
    __cost_copies_info_frame = LabelFrame(modify_data_tab, text='Cost &
Copies', padx= 20, pady= 15)
    __cost_copies_info_frame.grid(row= 4, column= 0, columnspan= 3, padx= 20,
pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store cost
    __cost_frame = Frame(__cost_copies_info_frame)
    __cost_frame.grid(row= 0, column= 0, padx= (0,7.5))
```

```
    __cost_label = Label(__cost_frame, text= 'Cost').grid(row= 0, column= 0)
    __cost_entry = Entry(__cost_frame, state= DISABLED)
    __cost_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store copies
    __copies_frame = Frame(__cost_copies_info_frame)
    __copies_frame.grid(row= 0, column= 1, padx= (7.5,0))
    __copies_label = Label(__copies_frame, text= 'Copies').grid(row= 0, column= 0)
    __var = IntVar(__copies_frame)
    __copies_spinbox = ttk.Spinbox(__copies_frame, from_=1, to='infinity',
textvariable= __var, state= DISABLED)
    __copies_spinbox.grid(row= 1, column=0)
    __var.set(1)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    def __search():
        global __dictionary,_field
        __dictionary = {'accno' : __acc_no_entry, 'classno' : __class_no_entry, 'title' :
__title_entry, 'author' : __author_name_entry, 'authorin' : __author_in_entry,
'publisher' : __publisher_name_entry, 'year' : __publisher_year_combobox,
'netamount' : __cost_entry, 'copies' : __var}
        mycursor.execute(f"select * from library where {_field} =
'{__dictionary[_field].get()}'")
        __dictionary[_field].delete(0,END)
        __data_all = [i for i in mycursor]
        __data = __data_all[0]
        __acc_no_entry.config(state= NORMAL)
```

```
        __class_no_entry.config(state= NORMAL)
        __title_entry.config(state= NORMAL)
        __pages_entry.config(state= NORMAL)
        __author_name_entry.config(state= NORMAL)
        __author_in_entry.config(state= NORMAL)
        __publisher_name_entry.config(state= NORMAL)
        __publisher_year_combobox.config(state= NORMAL)
        __cost_entry.config(state= NORMAL)
        __acc_no_entry.insert(0,__data[0])
        __class_no_entry.insert(0,__data[1])
        __title_entry.insert(0,__data[2])
        __pages_entry.insert(0,__data[3])
        __author_name_entry.insert(0,__data[4])
        __author_in_entry.insert(0,__data[5])
        __publisher_name_entry.insert(0,__data[6])
        __publisher_year_combobox.current(int(__data[7])-1750)
        __cost_entry.insert(0,__data[8])
        try:
            __copies_spinbox.config(state= NORMAL)
            __var.set(int(__data[9]))
        except:
            pass

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    def __reset():
        global __dictionary, _field
        __dictionary = {'accno' : __acc_no_entry, 'classno' : __class_no_entry, 'title' :
__title_entry, 'author' : __author_name_entry, 'authorin' : __author_in_entry,
```

```
'publisher' : __publisher_name_entry, 'year' : __publisher_year_combobox,
'netamount' : __cost_entry, 'copies' : __var}
        __dictionary[_field].delete(0,END)
        __dictionary[_field].config(state= DISABLED)
        __search_combobox.delete(0,END)
        __acc_no_entry.delete(0,END)
        __class_no_entry.delete(0,END)
        __title_entry.delete(0,END)
        __pages_entry.delete(0,END)
        __author_name_entry.delete(0,END)
        __author_in_entry.delete(0,END)
        __publisher_name_entry.delete(0,END)
        __publisher_year_combobox.current(time.localtime().tm_year-1750)
        __cost_entry.delete(0,END)
        __var.set(1)
        __acc_no_entry.config(state= DISABLED)
        __class_no_entry.config(state= DISABLED)
        __title_entry.config(state= DISABLED)
        __pages_entry.config(state= DISABLED)
        __author_name_entry.config(state= DISABLED)
        __author_in_entry.config(state= DISABLED)
        __publisher_name_entry.config(state= DISABLED)
        __publisher_year_combobox.config(state= DISABLED)
        __cost_entry.config(state= DISABLED)
        __copies_spinbox.config(state= DISABLED)

    global click
    click = 1
    def __add():
        global click
```

```
        mycursor.execute(f"insert into library
value{(__acc_no_entry.get(),__class_no_entry.get(),__title_entry.get(),__pages_entr
y.get(),__author_name_entry.get(),__author_in_entry.get(),__publisher_name_entr
y.get(),__publisher_year_combobox.get(),__cost_entry.get(),__copies_spinbox.get())
}")
        mydb.commit()
        click+=1
        __reset()
        __acc_no_entry.config(state= NORMAL)
        __acc_no_entry.delete(0,END)
        __acc_no_entry.insert(END,int(accno[-1][0])+click)
        __acc_no_entry.config(state= DISABLED)



    __reset_button = Button(modify_data_tab, text= 'Reset', command= __reset)
    __reset_button.grid(row= 5, column= 0, padx= (60,10), pady= (10,10))

    __search_button = Button(modify_data_tab, text= 'Search', command=
__search)
    __search_button.grid(row= 5, column= 1, padx= (10,10), pady= (10,10))

    __modify_button = Button(modify_data_tab, text= 'Modify', command= __add)
    __modify_button.grid(row= 5, column= 2, padx= (10,60), pady= (10,10))
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
#Search Data Tab
delete_data_tab = Frame(tab_window)
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
```

```
#___search by ___field frame
def delete_func():
    global ___field
    ___search_field_frame = LabelFrame(delete_data_tab, text= 'Search By', padx= 20, pady= 15)
    ___search_field_frame.grid(row= 0, column= 0, columnspan= 3, padx= 20, pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------------------------------------------#
    #___search by ___field
    ___values = ['Accession Number','Class Number','Book Name','Author Name','Author Initials','Publisher Name','Publishing Year','Cost','Copies']
    ___search_label = Label(___search_field_frame, text= 'Search By:').grid(row= 0, column= 0, padx= (0,15))
    ___search_combobox = ttk.Combobox(___search_field_frame, values= ___values)
    ___search_combobox.grid(row= 0, column= 1, padx= (0,15))
    def ___go():
        global ___field
        ___field = ''
        if ___search_combobox.get() == ___values[0]:
            ___acc_no_entry.config(state= NORMAL)
            ___field = 'accno'
        elif ___search_combobox.get() == ___values[1]:
            ___class_no_entry.config(state= NORMAL)
            ___field = 'classno'
        elif ___search_combobox.get() == ___values[2]:
            ___title_entry.config(state= NORMAL)
            ___field = 'title'
```

```
        elif ___search_combobox.get() == ___values[3]:
            ___author_name_entry.config(state= NORMAL)
            ___field = 'author'
        elif ___search_combobox.get() == ___values[4]:
            ___author_in_entry.config(state= NORMAL)
            ___field = 'authorin'
        elif ___search_combobox.get() == ___values[5]:
            ___publisher_name_entry.config(state= NORMAL)
            ___field = 'publisher'
        elif ___search_combobox.get() == ___values[6]:
            ___publisher_year_combobox.config(state= NORMAL)
            ___field = 'year'
        elif ___search_combobox.get() == ___values[7]:
            ___cost_entry.config(state= NORMAL)
            ___field = 'netamount'
        elif ___search_combobox.get() == ___values[8]:
            ___copies_spinbox.config(state= NORMAL)
            ___field = 'copies'
        # return ___field, criteria
    ___go_button = Button(___search_field_frame, text= 'Go', command= ___go,
width= 6)
    ___go_button.grid(row= 0, column= 2)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Search Data
    #LabelFrame to store accno and classno
    ___library_info_frame = LabelFrame(delete_data_tab, text='Library
Information', padx= 20, pady= 15)
```

```
   ___library_info_frame.grid(row= 1, column= 0, columnspan= 3, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store accno
   ___acc_no_frame = Frame(___library_info_frame)
   ___acc_no_frame.grid(row=0,column=0,padx=(0,7.5))
   ___acc_no_label = Label(___acc_no_frame, text= 'Accession
Number').grid(row=0,column=0)
   ___acc_no_entry = Entry(___acc_no_frame)
   ___acc_no_entry.config(state= DISABLED)
   ___acc_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store classno
   ___class_no_frame = Frame(___library_info_frame)
   ___class_no_frame.grid(row=0,column=1,padx=(7.5,0))
   ___class_no_label = Label(___class_no_frame, text= 'Class
Number').grid(row=0,column=0)
   ___class_no_entry = Entry(___class_no_frame, state= DISABLED)
   ___class_no_entry.grid(row=1,column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #LabelFrame to store book info
   ___book_info_frame = LabelFrame(delete_data_tab, text='Book Information',
padx= 20, pady= 15)
```

```
    ___book_info_frame.grid(row= 2, column= 0, columnspan= 3, padx= 20, pady=
(10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #Frame to store title
    ___title_frame = Frame(___book_info_frame)
    ___title_frame.grid(row= 0, column= 0, padx= (0,7.5))
    ___title_label = Label(___title_frame, text= 'Title').grid(row= 0, column= 0)
    ___title_entry = Entry(___title_frame, state= DISABLED)
    ___title_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #Frame to store author name
    ___author_name_frame = Frame(___book_info_frame)
    ___author_name_frame.grid(row= 1, column= 0, padx= (0,7.5), pady= (5,0))
    ___author_name_label = Label(___author_name_frame, text= 'Author
Name').grid(row= 0, column= 0)
    ___author_name_entry = Entry(___author_name_frame, state= DISABLED)
    ___author_name_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------#
    #Frame to store author initials
    ___author_in_frame = Frame(___book_info_frame)
    ___author_in_frame.grid(row= 1, column= 1, padx= (7.5,0), pady= (5,0))
    ___author_in_label = Label(___author_in_frame, text= 'Author
Initial').grid(row= 0, column= 0)
    ___author_in_entry = Entry(___author_in_frame, state= DISABLED)
```

```
    ___author_in_entry.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store number of pages
    ___pages_frame = Frame(___book_info_frame)
    ___pages_frame.grid(row= 0, column= 1, padx= (7.5,0))
    ___pages_label = Label(___pages_frame, text= 'Pages').grid(row= 0, column= 0)
    ___pages_entry = Entry(___pages_frame, state= DISABLED)
    ___pages_entry.grid(row= 1, column=0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #LabelFrame to store publisher info
    ___publisher_info_frame = LabelFrame(delete_data_tab, text='Publisher
Information', padx= 20, pady= 15)
    ___publisher_info_frame.grid(row= 3, column=0, columnspan= 3, padx= 20,
pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    #Frame to store publisher name
    ___publisher_name_frame = Frame(___publisher_info_frame)
    ___publisher_name_frame.grid(row= 0, column= 0, padx= (0,7.5))
    ___publisher_name_label = Label(___publisher_name_frame, text= 'Publisher
Name').grid(row= 0, column= 0)
    ___publisher_name_entry = Entry(___publisher_name_frame, state=
DISABLED)
    ___publisher_name_entry.grid(row= 1, column=0)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store publisher year
   ___publisher_year_frame = Frame(___publisher_info_frame)
   ___publisher_year_frame.grid(row= 0, column= 1, padx= (7.5,0))
   ___publisher_year_label = Label(___publisher_year_frame, text= 'Publishing
Year').grid(row= 0, column= 0)
   ___publisher_year_combobox = ttk.Combobox(___publisher_year_frame,
values=[i for i in range(1750,time.localtime().tm_year+1)], state= DISABLED)
   ___publisher_year_combobox.current(time.localtime().tm_year-1750)
   ___publisher_year_combobox.grid(row= 1, column= 0)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #LabelFrame to store cost & copies info
   ___cost_copies_info_frame = LabelFrame(delete_data_tab, text='Cost &
Copies', padx= 20, pady= 15)
   ___cost_copies_info_frame.grid(row= 4, column= 0, columnspan= 3, padx= 20,
pady= (10,0), sticky= 'news')

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store cost
   ___cost_frame = Frame(___cost_copies_info_frame)
   ___cost_frame.grid(row= 0, column= 0, padx= (0,7.5))
   ___cost_label = Label(___cost_frame, text= 'Cost').grid(row= 0, column= 0)
   ___cost_entry = Entry(___cost_frame, state= DISABLED)
   ___cost_entry.grid(row= 1, column=0)
```

```
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   #Frame to store copies
   ___copies_frame = Frame(___cost_copies_info_frame)
   ___copies_frame.grid(row= 0, column= 1, padx= (7.5,0))
   ___copies_label = Label(___copies_frame, text= 'Copies').grid(row= 0, column=
0)
   ___var = IntVar(___copies_frame)
   ___copies_spinbox = ttk.Spinbox(___copies_frame, from_=1, to='infinity',
textvariable= ___var, state= DISABLED)
   ___copies_spinbox.grid(row= 1, column=0)
   ___var.set(1)

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
   def ___search():
      global ___dictionary,___field
      ___dictionary = {'accno' : ___acc_no_entry, 'classno' : ___class_no_entry,
'title' : ___title_entry, 'author' : ___author_name_entry, 'authorin' :
___author_in_entry, 'publisher' : ___publisher_name_entry, 'year' :
___publisher_year_combobox, 'netamount' : ___cost_entry, 'copies' : ___var}
      mycursor.execute(f"select * from library where {___field} =
'{___dictionary[___field].get()}'")
      ___dictionary[___field].delete(0,END)
      ____data_all = [i for i in mycursor]
      ___data = ____data_all[0]
      ___acc_no_entry.config(state= NORMAL)
      ___class_no_entry.config(state= NORMAL)
      ___title_entry.config(state= NORMAL)
```

```
        ___pages_entry.config(state= NORMAL)
        ___author_name_entry.config(state= NORMAL)
        ___author_in_entry.config(state= NORMAL)
        ___publisher_name_entry.config(state= NORMAL)
        ___publisher_year_combobox.config(state= NORMAL)
        ___cost_entry.config(state= NORMAL)
        ___acc_no_entry.insert(0,___data[0])
        ___class_no_entry.insert(0,___data[1])
        ___title_entry.insert(0,___data[2])
        ___pages_entry.insert(0,___data[3])
        ___author_name_entry.insert(0,___data[4])
        ___author_in_entry.insert(0,___data[5])
        ___publisher_name_entry.insert(0,___data[6])
        ___publisher_year_combobox.current(int(___data[7])-1750)
        ___cost_entry.insert(0,___data[8])
        try:
            ___copies_spinbox.config(state= NORMAL)
            ___var.set(int(___data[9]))
        except:
            pass

#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
    def ___reset():
        global ___dictionary, ___field
        ___dictionary = {'accno' : ___acc_no_entry, 'classno' : ___class_no_entry,
'title' : ___title_entry, 'author' : ___author_name_entry, 'authorin' :
___author_in_entry, 'publisher' : ___publisher_name_entry, 'year' :
___publisher_year_combobox, 'netamount' : ___cost_entry, 'copies' : ___var}
        ___dictionary[___field].delete(0,END)
```

```
        ___dictionary[___field].config(state= DISABLED)
        ___search_combobox.delete(0,END)
        ___acc_no_entry.delete(0,END)
        ___class_no_entry.delete(0,END)
        ___title_entry.delete(0,END)
        ___pages_entry.delete(0,END)
        ___author_name_entry.delete(0,END)
        ___author_in_entry.delete(0,END)
        ___publisher_name_entry.delete(0,END)
        ___publisher_year_combobox.current(time.localtime().tm_year-1750)
        ___cost_entry.delete(0,END)
        ___var.set(1)
        ___acc_no_entry.config(state= DISABLED)
        ___class_no_entry.config(state= DISABLED)
        ___title_entry.config(state= DISABLED)
        ___pages_entry.config(state= DISABLED)
        ___author_name_entry.config(state= DISABLED)
        ___author_in_entry.config(state= DISABLED)
        ___publisher_name_entry.config(state= DISABLED)
        ___publisher_year_combobox.config(state= DISABLED)
        ___cost_entry.config(state= DISABLED)
        ___copies_spinbox.config(state= DISABLED)


    def delete_data():
        global ___dictionary,___field
        ___dictionary = {'accno' : ___acc_no_entry, 'classno' : ___class_no_entry,
'title' : ___title_entry, 'author' : ___author_name_entry, 'authorin' :
___author_in_entry, 'publisher' : ___publisher_name_entry, 'year' :
___publisher_year_combobox, 'netamount' : ___cost_entry, 'copies' : ___var}
```

```
        mycursor.execute(f"delete from library where {___field} =
'{___dictionary[___field].get()}'")
        mydb.commit()
        ___reset()

    ___reset_button = Button(delete_data_tab, text= 'Reset', command= ___reset)
    ___reset_button.grid(row= 5, column= 0, padx= (60,10), pady= (10,10))
    ___search_button = Button(delete_data_tab, text= 'Search', command=
___search)
    ___search_button.grid(row= 5, column= 1, padx= (10,10), pady= (10,10))
    ___delete_button = Button(delete_data_tab, text= 'Delete', command=
delete_data)
    ___delete_button.grid(row= 5, column= 2, padx= (10,60), pady= (10,10))
#-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------#
homepage_tab = Frame(tab_window)
def homepage_func():
    Label(homepage_tab, text= 'LIBRARY', font= ('Arial',25)).grid(row= 0, column=
0, padx= (62,62), pady= (210,0))
    Label(homepage_tab, text= 'MANAGEMENT', font= ('Arial',25)).grid(row= 1,
column= 0, padx= (62,62), pady= (0,0))
    Label(homepage_tab, text= 'SYSTEM', font= ('Arial',25)).grid(row= 2, column= 0,
padx= (62,62), pady= (0,240))




def tab_change(event):
    if tab_window.select() == '.!notebook.!frame':
```

```
        add_func()
    elif tab_window.select() == '.!notebook.!frame2':
        search_func()
    elif tab_window.select() == '.!notebook.!frame3':
        modify_func()
    elif tab_window.select() == '.!notebook.!frame4':
        delete_func()
    elif tab_window.select() == '.!notebook.!frame5':
        homepage_func()
    else:
        pass

tab_window.bind("<<NotebookTabChanged>>", tab_change)

tab_window.add(homepage_tab, text= 'Home')
tab_window.add(add_data_tab, text= 'Add Data')
tab_window.add(search_data_tab, text= 'Search Data')
tab_window.add(modify_data_tab, text= 'Modify Data')
tab_window.add(delete_data_tab, text= 'Delete Data')
tab_window.grid(row= 0, column= 0)

print(tab_window.select())

window.mainloop()
```

BACK END

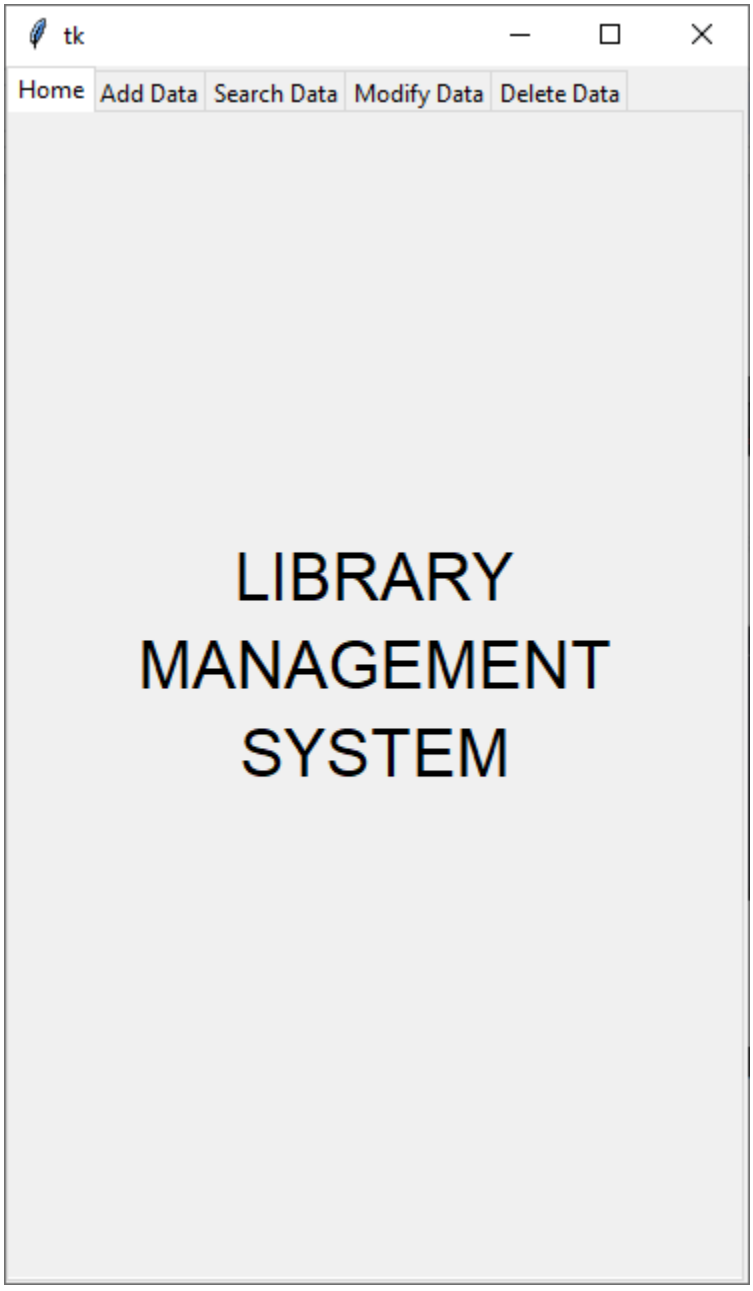
```
import  csv
with open(''BookInfo.csv'','r') as file:
   csvreader=csv.reader(file)
   for row in csvreader:
      print(row)

import mysql.connector
mydb =
mysql.connector.connect(host='127.0.0.1',user='root',passwd='',db='trial')
cursor = mydb.cursor()
file=open(''Hillwoods Acc. Register.xls - Sheet1.csv'','r')
csvreader = csv.reader(file)
for row in csvreader:
   if row[0] == 'Acc. No.':
      pass
   else:
      for i in row:
         if i == "":
            row[row.index(i)] = 0
         if i == "#":
            row[row.index(i)] = 0
      sql = f''INSERT INTO bookinfo (Acc_No, Class_No, Title, Author,
Publisher, Year, Pages, Author_initial, Volume, Net_Amount, Copies, CD)
VALUES {tuple(row)}''
      print(sql)
      try:
```

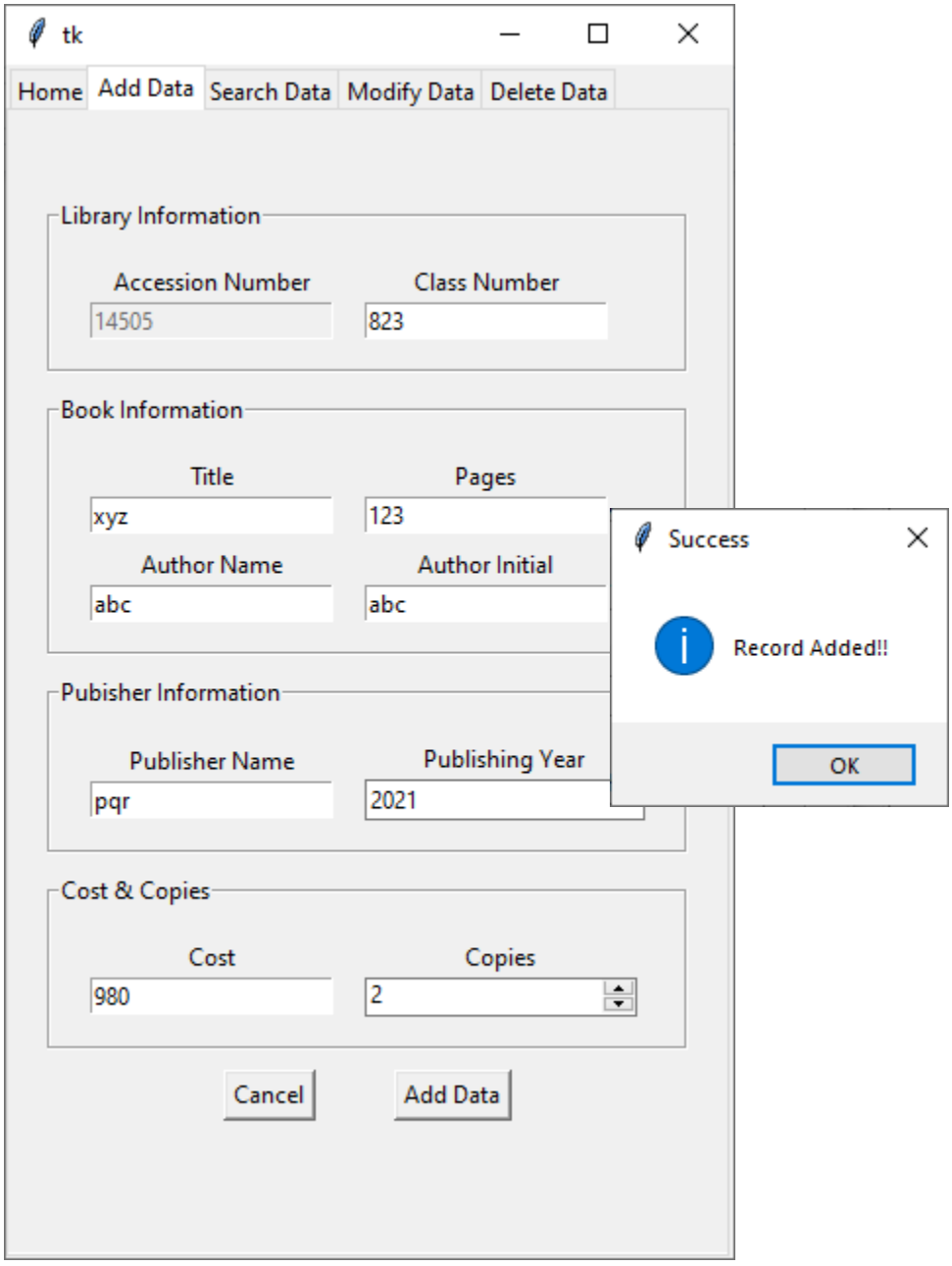
```
            cursor.execute(sql)
        except:
            mydb.commit()
mydb.commit()
mydb.close()
```

OUTPUT

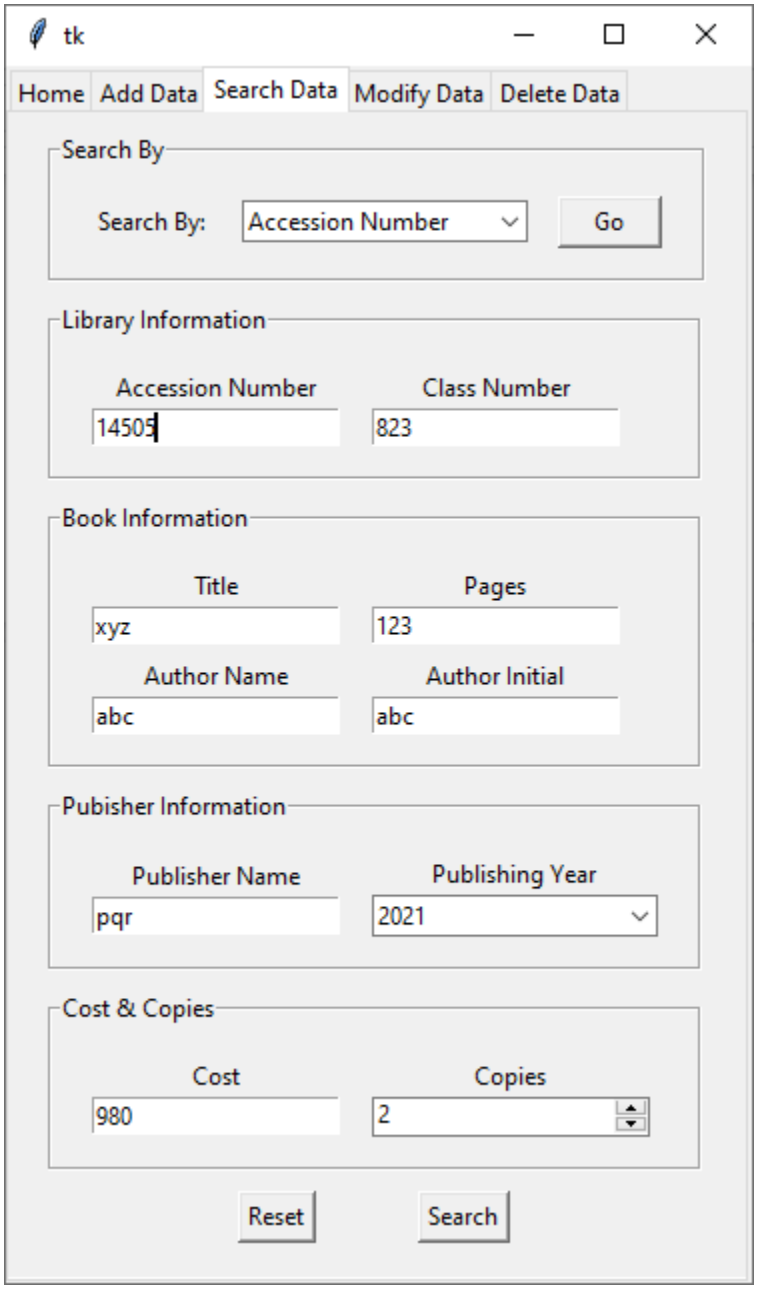
Home Screen



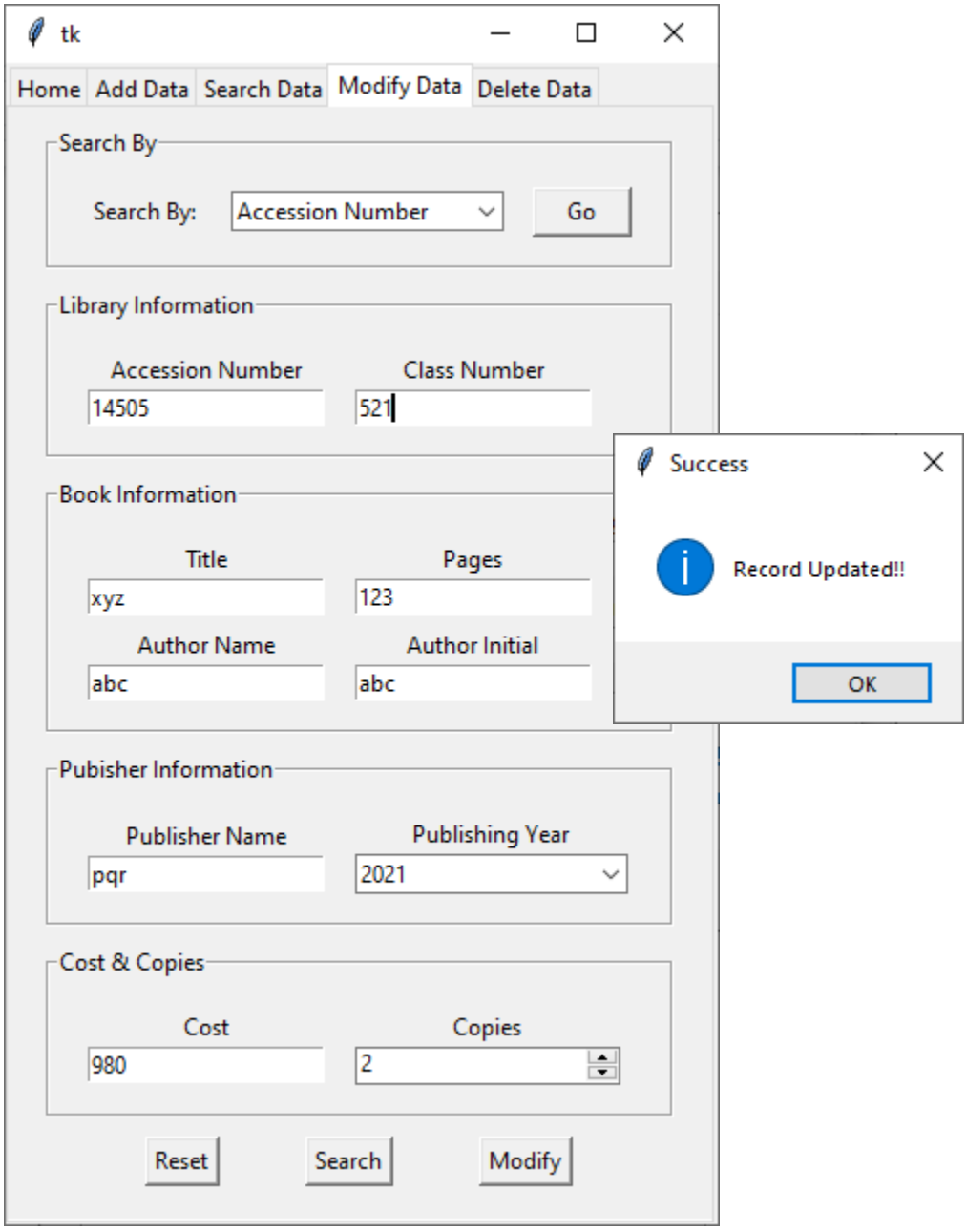
Add Data Tab


tk
Home
Add Data
Search Data
Modify Data
Delete Data
Library Information
Accession Number
14505
Class Number
823
Book Information
Title
xyz
Pages
123
Author Name
abc
Author Initial
abc
Pubisher Information
Publisher Name
pqr
Publishing Year
2021
Cost & Copies
Cost
980
Copies
2
Cancel
Add Data
Success
Record Added!!
OK

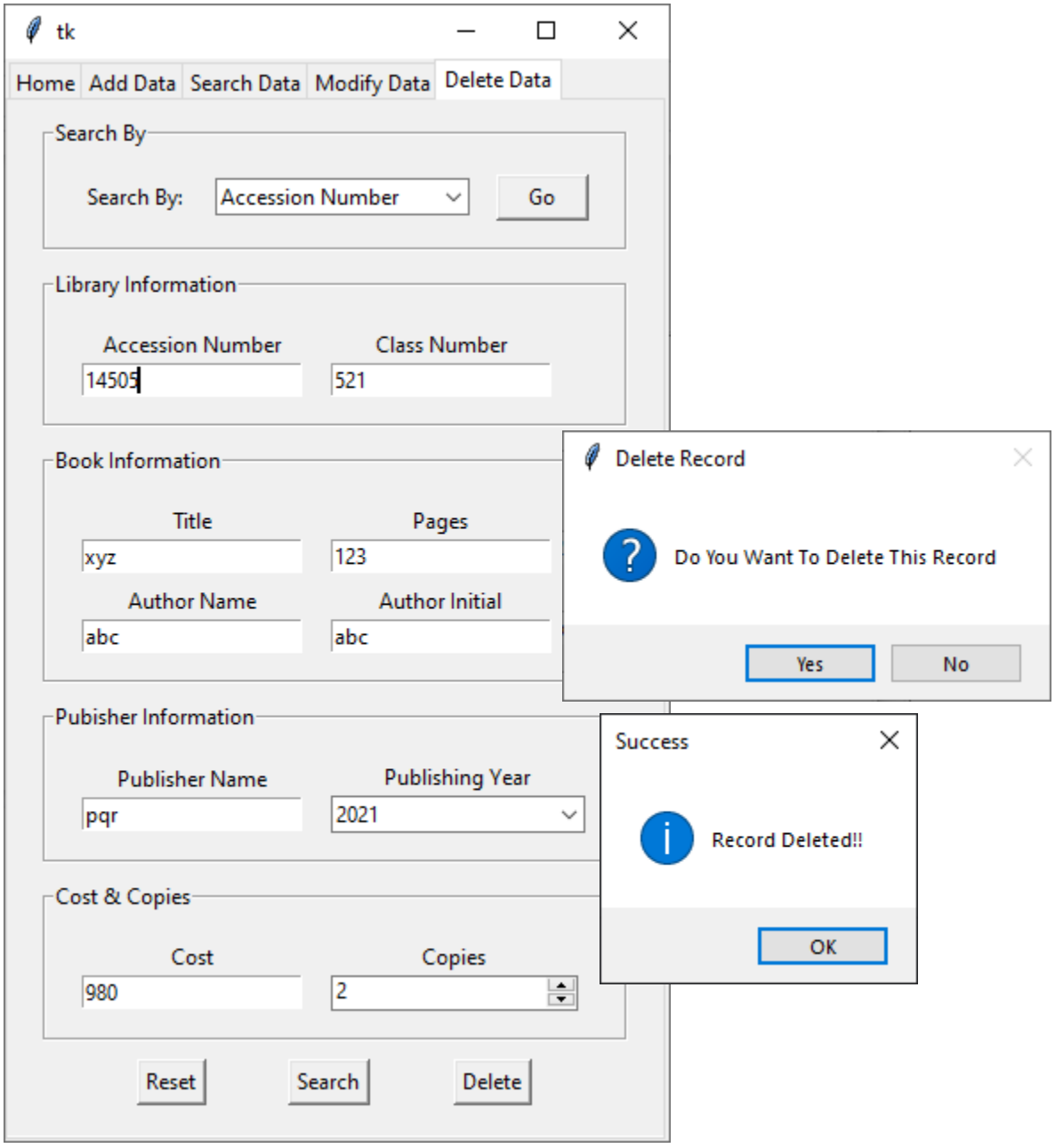
Search Data Tab



Modify Data Tab



Delete Book Data



FUTURE SCOPE

The program can be made more functional by adding the option to be able to access student data. This access to the student data can be used to maintain a record of the books issued from the library in addition to maintaining a record of defaulters and the date of issue and return.
A bar code scanner and bar code tags can also be applied on books to maintain a more precise data of total number of books and books issued by students. This also makes the process more automated and reduces the manual work and redundancy.

BIBLIOGRAPHY

★ Computer Science with Python (Textbook for class XII) by SUMITA ARORA

★ www.cbse.nic.in