1b-library-book-management-system

April 21, 2023

[1]:
```
import sqlite3
from tkinter import *
import tkinter as tk
import tkinter.ttk as ttk
import tkinter.messagebox as mb
import tkinter.simpledialog as sd

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

connector.execute('CREATE TABLE IF NOT EXISTS Books (BK_ID INTEGER PRIMARY KEY␣
 ↪NOT NULL,BK_NAME TEXT, AUTHOR_NAME TEXT, YEAR INTEGER)')
```

[1]:
```
<sqlite3.Cursor at 0x55e9c0cb20>
```

[2]:
```
def display_records():
    global connector, cursor
    global tree

    tree.delete(*tree.get_children())

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

    for records in data:
        tree.insert('', END, values=records)

def clear_fields():
    global  bk_id, bk_name, author_name, year

    year.set('Available')
    for i in ['bk_id', 'bk_name', 'author_name', 'year']:
        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()
```

[3]:
```
def view_record():
    global bk_name, bk_id, author_name, year
    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_id.set(selection[0]) ; bk_name.set(selection[1]) ;author_name.
↪set(selection[2]); year.set(selection[3])
```

[4]:
```
def add_record():
    global connector
    global bk_id,bk_name, author_name, year

    surety = mb.askyesno('Are you sure?','Are you sure this is the data you␣
↪want to enter?\n')
    if surety:
        try:
            connector.execute('INSERT INTO Books (BK_ID,BK_NAME,␣
↪AUTHOR_NAME,YEAR) VALUES (?, ?, ?, ?)',(bk_id.get(),bk_name.
↪get(),author_name.get(), year.get()))
            connector.commit()

            clear_and_display()

            mb.showinfo('Record added','The new record was successfully added␣
↪to your database')
        except sqlite3.IntegrityError:mb.showerror('error','Book ID already in␣
↪use!')
```

[5]:
```
def update_record():
    def update():
        global bk_id, bk_name, author_name, year
        global connector, tree
        cursor.execute('UPDATE Books SET BK_NAME=?, AUTHOR_NAME=?, YEAR=? WHERE␣
↪BK_ID=?', (bk_name.get(),author_name.get(), year.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,fg='White', width=20, command=update)
    edit.place(x=50, y=425)
```

[6]:
```
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 Books WHERE BK_ID=?', (selection[0], ))
    connector.commit()

    tree.delete(current_item)

    mb.showinfo('Done','The record deleted successfully.')

    clear_and_display()
```

[7]:
```
def Exit():
    result=mb.askquestion("Exit","Do you want to exit?(y/n)",icon='warning')
    if result=='yes':
        root.destroy()
        exit()
```

[8]:
```
# Variables
lf_bg = '#FFFF00' # Left Frame Background Color
rtf_bg = '#FFFF00' # Right Top Frame Background Color
```

```
rbf_bg = '#827B60' # Right Bottom Frame Background Color
btn_hlb_bg = '#000000' # 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('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_id = StringVar()
bk_name = StringVar()
author_name = StringVar()
year = 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,bg='#0C090A')
RB_frame.place(relx=0.3, rely=0.24, relheight=0.785, relwidth=0.7)

# Left Frame
Label(left_frame, text='Book ID', bg=lf_bg, font=lbl_font).place(x=100, y=20)
bk_id_entry = Entry(left_frame, width=25, font=entry_font, text=bk_id)
bk_id_entry.place(x=45, y=50)

Label(left_frame, text='Book Name', bg=lf_bg, font=lbl_font).place(x=90, y=90)
Entry(left_frame, width=25, font=entry_font, text=bk_name).place(x=45, y=125)

Label(left_frame, text='Author Name', bg=lf_bg, font=lbl_font).place(x=90,␣
 ↪y=165)
Entry(left_frame, width=25, font=entry_font, text=author_name).place(x=45,␣
 ↪y=205)

Label(left_frame, text='Year', bg=lf_bg, font=lbl_font).place(x=115, y=235)
Entry(left_frame, width=25, font=entry_font, text=year).place(x=45, y=270)
```

```
submit = Button(left_frame, text='Submit', font=btn_font, bg=btn_hlb_bg,␣
 ↪fg='White',width=20, command=add_record)
submit.place(x=50, y=315)

clear = Button(left_frame, text='Clear fields', font=btn_font,␣
 ↪bg=btn_hlb_bg,fg='White', width=10, command=clear_fields)
clear.place(x=10, y=375)

Exit = Button(left_frame, text='Exit', font=btn_font, bg=btn_hlb_bg,␣
 ↪fg='White',width=10, command=Exit)
Exit.place(x=190, y=375)


# Right Top Frame
Button(RT_frame, text='Update book details', font=btn_font,␣
 ↪bg=btn_hlb_bg,fg='White', width=20,command=update_record).place(x=118,y=30)
Button(RT_frame, text='Delete book record', font=btn_font,␣
 ↪bg=btn_hlb_bg,fg='White', width=20, command=remove_record).place(x=378, y=30)

# Right Bottom Frame
Label(RB_frame, text='Book Details', bg=btn_hlb_bg, fg='White',font=("Noto Sans␣
 ↪CJK TC", 20, 'bold')).pack(side=TOP, fill=X)

tree = ttk.Treeview(RB_frame, selectmode=BROWSE, columns=('Book ID','Book␣
 ↪Name', 'Author','Year'))

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 ID', text='Book ID', anchor=CENTER)
tree.heading('Book Name', text='Book Name', anchor=CENTER)
tree.heading('Author', text='Author', anchor=CENTER)
tree.heading('Year', text='Year', anchor=CENTER)

tree.column('#0', width=0, stretch=NO)
tree.column('#1', width=55, stretch=NO)
tree.column('#2', width=190, stretch=NO)
tree.column('#3', width=270, stretch=NO)
tree.place(y=30, x=0, relheight=0.9, relwidth=1)
Label(RB_frame,text='*Select a data from the list before Updating or deleteing␣
  ↪',font=("Noto Sans CJK TC", 8)).pack(side=TOP, fill=X)

clear_and_display()

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

[ ]:

```
```

[ ]:

```
```