# Importing all necessary modules

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 (BKNAME TEXT, BKID TEXT PRIMARY KEY NOT NULL, AUTHORNAME TEXT, BKSTATUS TEXT, CARD_ID TEXT)'
)

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 emp
else:
  return Cid
```

def display_records():
global connector, cursor
global tree

```
tree.delete(*tree.get_children())

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

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

def clearfields():

global bkstatus, bkid, bkname, authorname, cardid

```
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 clearanddisplay():

clearfields()

displayrecords()

def viewrecord():

global bkname, bkid, bkstatus, authorname, cardid

global tree

```
      if not tree.focus():
        mb.showerror('Select a row!', 'To view a record, you must sele
        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.
      author_name.set(selection[2])
```

def addrecord():

global connector

global bkname, bkid, authorname, 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?\nPle

if surety:
      try:
            connector.execute(
             'INSERT INTO Library (BK_NAME, BK_ID, AUTHOR_NAME, BK_ST
                    (bk_name.get(), bk_id.get(), author_name.get(), b
            connector.commit()

            clear_and_display()

            mb.showinfo('Record added', 'The new record was successfu
  except sqlite3.IntegrityError:
            mb.showerror('Book ID already in use!',
                         'The Book ID you are trying to enter is alr
```

def updaterecord():

def update():

global bkstatus, bkname, bkid, authorname, cardid

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_
     connector.commit()

     clear_and_display()

     edit.destroy()
     bk_id_entry.config(state='normal')
     clear.config(state='normal')
```

```
view_record()

bkidentry.config(state='disable')
clear.config(state='disable')

edit = Button(leftframe, text='Update Record', font=btnfont, bg=btnhlbbg, 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

currentitem = tree.focus()
values = tree.item(currentitem)
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.')

clearanddisplay()

def deleteinventory():
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.getchildren())

cursor.execute('DELETE FROM Library')
connector.commit()
else:
return
```

```
def changeavailability():
global cardid, tree, connector

if not tree.selection():
mb.showerror('Error!', 'Please select a book from the database')
return

currentitem = tree.focus()
values = tree.item(currentitem)
BKid = values['values'][1]
BKstatus = values["values"][3]

if BKstatus == 'Issued':
surety = mb.askyesno('Is return confirmed?', 'Has the book been returned to you?')
if surety:
cursor.execute('UPDATE Library SET bkstatus=?, cardid=? WHERE bkid=?',
('Available', 'N/A', BKid))
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 bkstatus=?, cardid=? where bkid=?', ('Issued',
issuercard(), BKid))
connector.commit()

clearanddisplay()
```

# Variables

```
lfbg = 'LightSkyBlue' # Left Frame Background Color
rtfbg = 'DeepSkyBlue' # Right Top Frame Background Color
rbfbg = 'DodgerBlue' # Right Bottom Frame Background Color
btnhlb_bg = 'SteelBlue' # Background color for Head Labels and Buttons
```

```
lblfont = ('Georgia', 13) # Font for all labels
entryfont = ('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=btnhlbbg, fg='White').pack(side=TOP, fill=X)
```

# StringVars

```
bkstatus = StringVar()
bkname = StringVar()
bkid = StringVar()
authorname = StringVar()
card_id = StringVar()
```

# Frames

```
leftframe = Frame(root, bg=lfbg)
left_frame.place(x=0, y=30, relwidth=0.3, relheight=0.96)

RTframe = Frame(root, bg=rtfbg)
RT_frame.place(relx=0.3, y=30, relheight=0.2, relwidth=0.7)

RBframe = Frame(root)
RBframe.place(relx=0.3, rely=0.24, relheight=0.785, relwidth=0.7)
```

# Left Frame

```
Label(leftframe, text='Book Name', bg=lfbg, font=lblfont).place(x=98, y=25)
Entry(leftframe, width=25, font=entryfont, text=bkname).place(x=45, y=55)

Label(leftframe, text='Book ID', bg=lfbg, font=lblfont).place(x=110, y=105)
bkidentry = Entry(leftframe, width=25, font=entryfont, text=bkid)
bkidentry.place(x=45, y=135)

Label(leftframe, text='Author Name', bg=lfbg, font=lblfont).place(x=90, y=185)
Entry(leftframe, width=25, font=entryfont, text=authorname).place(x=45, y=215)

Label(leftframe, text='Status of the Book', bg=lfbg, font=lblfont).place(x=75, y=265)
dd = OptionMenu(leftframe, bkstatus, *['Available', 'Issued'])
dd.configure(font=entryfont, width=12)
dd.place(x=75, y=300)

submit = Button(leftframe, text='Add new record', font=btnfont, bg=btnhlbbg, width=20,
command=add_record)
submit.place(x=50, y=375)

clear = Button(leftframe, text='Clear fields', font=btnfont, bg=btnhlbbg, width=20,
command=clear_fields)
clear.place(x=50, y=435)
```

# Right Top Frame

```
Button(RTframe, text='Delete book record', font=btnfont, bg=btnhlbbg, width=17,
command=removerecord).place(x=8, y=30)
Button(RTframe, text='Delete full inventory', font=btnfont, bg=btnhlbbg, width=17,
command=deleteinventory).place(x=178, y=30)
Button(RTframe, text='Update book details', font=btnfont, bg=btnhlbbg, width=17,
command=updaterecord).place(x=348, y=30)
Button(RTframe, text='Change Book Availability', font=btnfont, bg=btnhlbbg, width=19,
command=changeavailability).place(x=518, y=30)
```

# Right Bottom Frame

```
Label(RBframe, text='BOOK INVENTORY', bg=rbfbg, 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)

clearanddisplay()
```

# Finalizing the window

```
root.update()
root.mainloop()
```