

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN**

VAN LANG
UNIVERSITY



BÁO CÁO PROJECT 2

GIAO DIỆN QUẢN LÝ DATABASE ĐƠN GIẢN

Giảng viên hướng dẫn: Huỳnh Thái Học

Sinh viên thực hiện: Trương Việt Vũ

Mã số sinh viên: 2274802011045

Học phần : Lập trình Python nâng cao

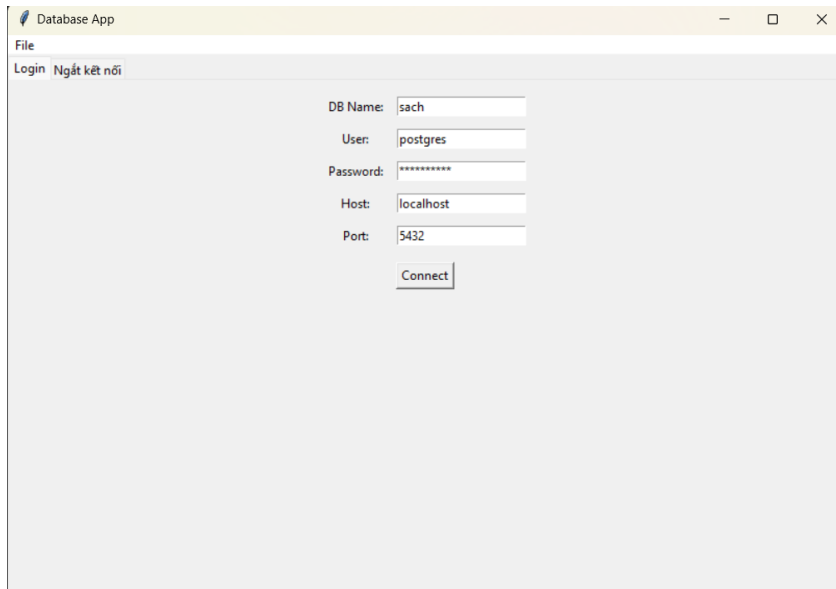
Tp Hồ Chí Minh ngày 20 tháng 10 năm 2024

NỘI DUNG BÁO CÁO

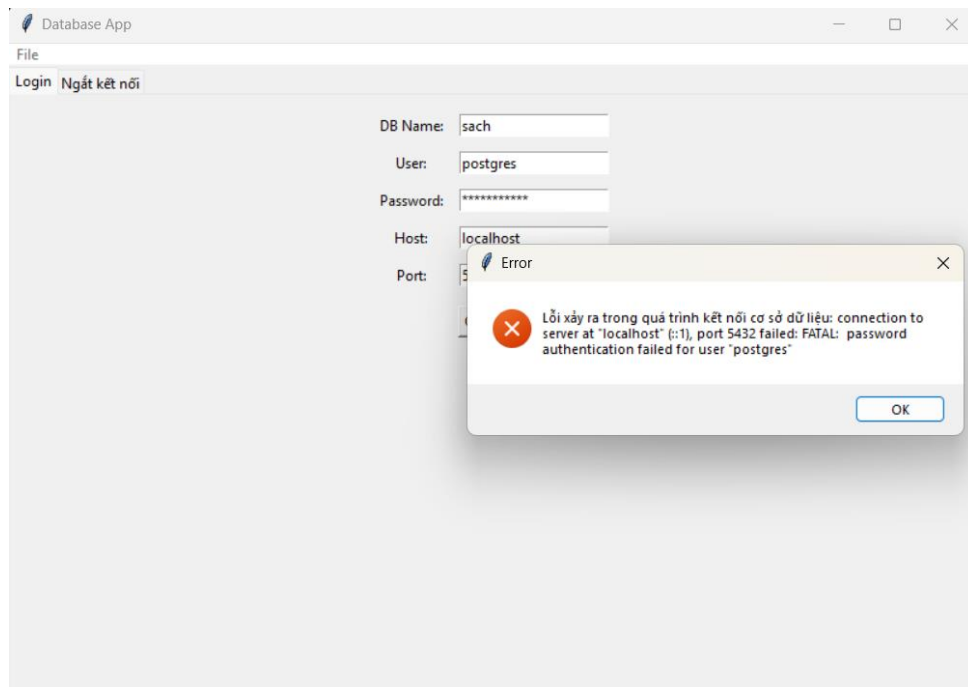
Giao diện GUI đơn giản viết bằng code python dùng để quản lí cơ sở dữ liệu liên quan đến sách

1. Giao diện

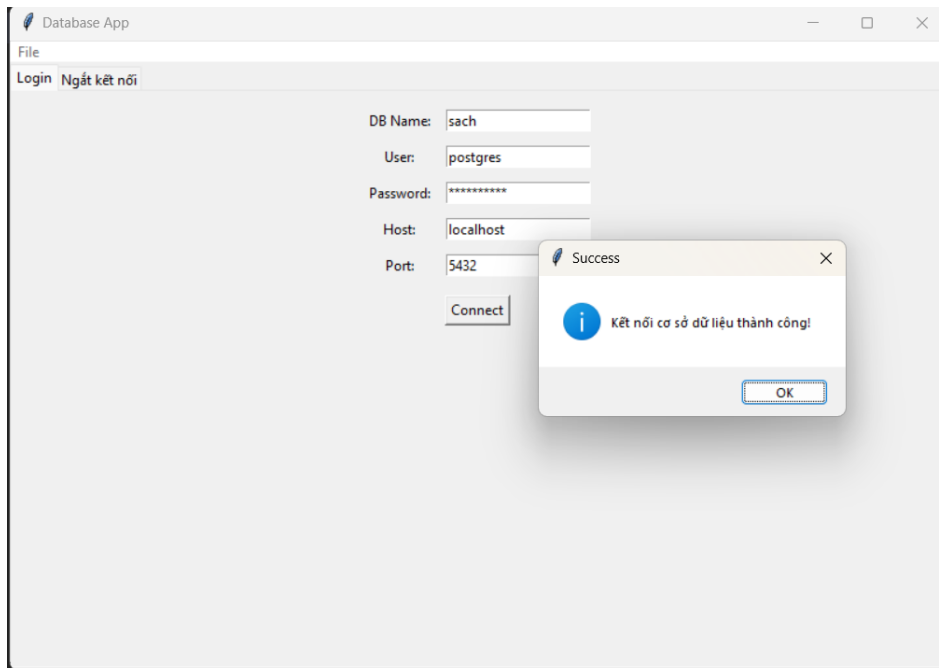
Giao diện đăng nhập



Nếu nhập sai sẽ báo lỗi

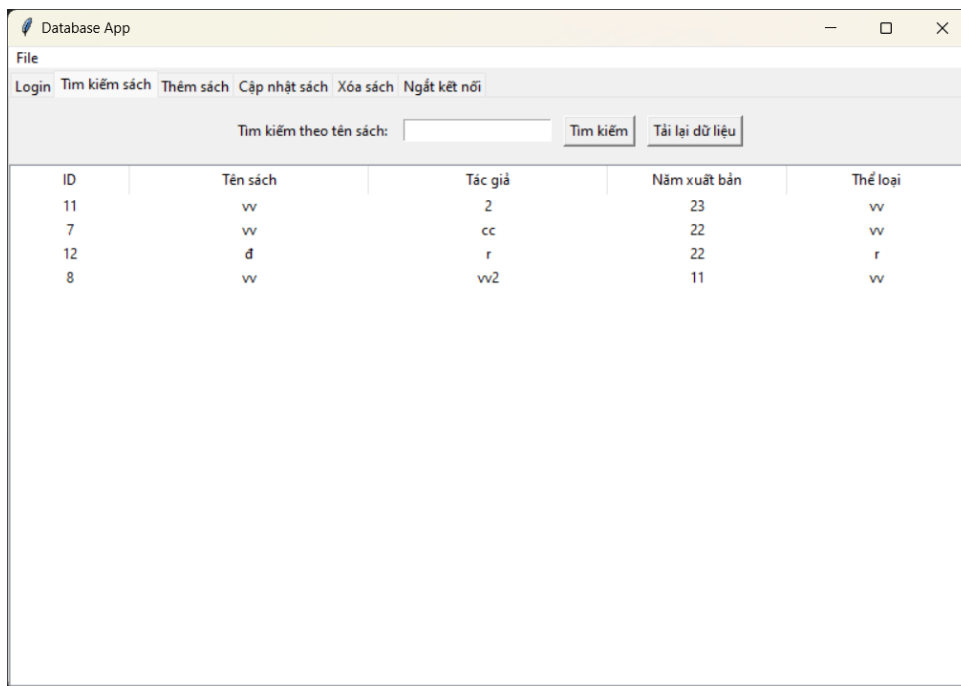


Và nhập đúng sẽ hiện thông báo kết nối thành công và chuyển sang giao diện quản lý

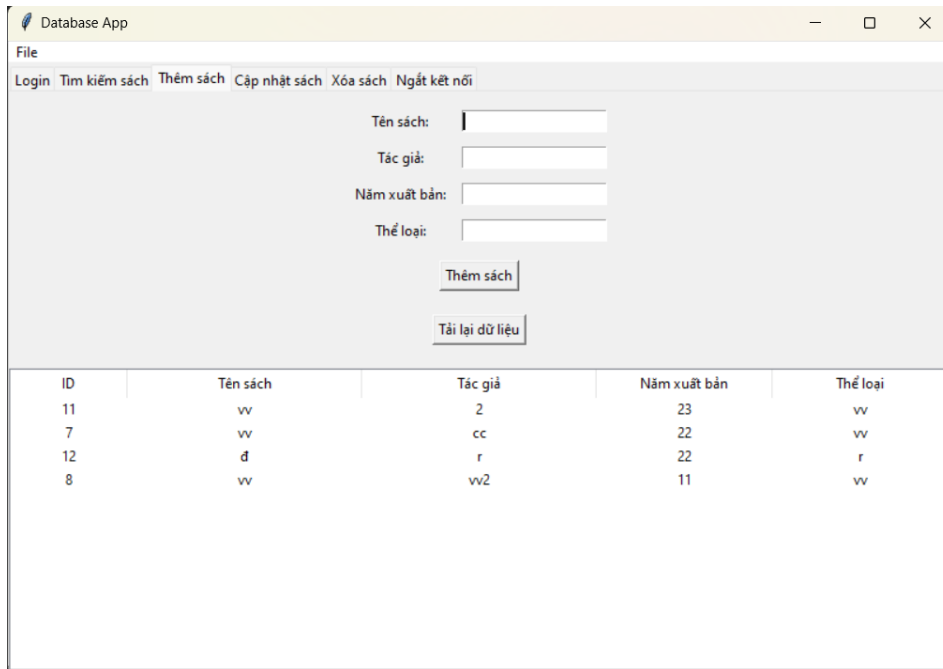


Lúc này việc quản lý cơ sở dữ liệu sẽ gồm các chức năng:

Chức năng tìm kiếm sách theo tên sách:



Chức năng thêm vào sách mới, không cần nhập ID khi thêm sách vì ID sẽ được tạo tự động, sau khi thêm thành công dữ liệu sẽ lập tức được đưa xuống phần hiển thị dữ liệu bên dưới



Database App

File

Login Tìm kiếm sách Thêm sách Cập nhật sách Xóa sách Ngắt kết nối

Tên sách:

Tác giả:

Năm xuất bản:

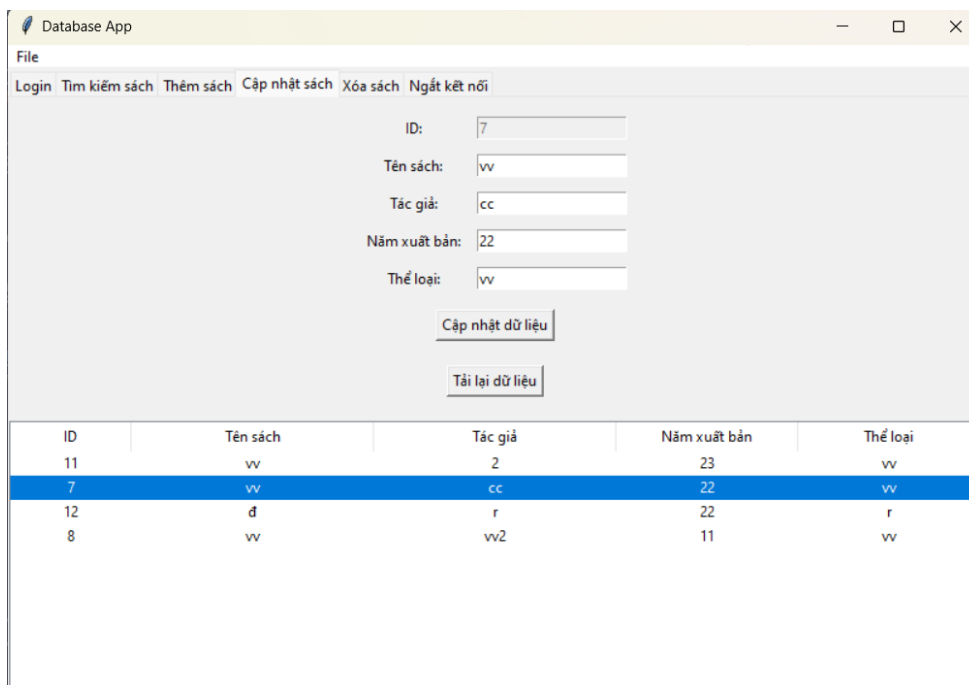
Thể loại:

Thêm sách

Tải lại dữ liệu

ID	Tên sách	Tác giả	Năm xuất bản	Thể loại
11	vv	2	23	vv
7	vv	cc	22	vv
12	đ	r	22	r
8	vv	vv2	11	vv

Chức năng cập nhật sách, người dung có thể nhập vào các entry hoặc chỉ cần nhập vào dữ liệu bên dưới lập tức dữ liệu sẽ được đưa vào entry để người dung có thể sửa đổi, id sẽ không thay đổi được



Database App

File

Login Tìm kiếm sách Thêm sách Cập nhật sách Xóa sách Ngắt kết nối

ID:

Tên sách:

Tác giả:

Năm xuất bản:

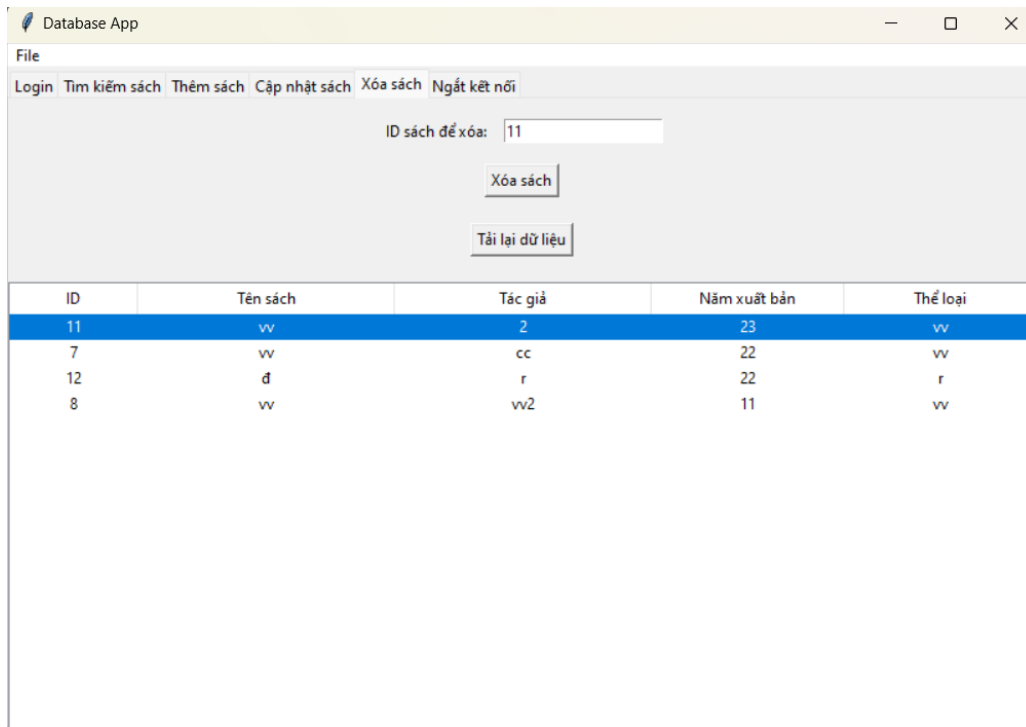
Thể loại:

Cập nhật dữ liệu

Tải lại dữ liệu

ID	Tên sách	Tác giả	Năm xuất bản	Thể loại
11	vv	2	23	vv
7	vv	cc	22	vv
12	đ	r	22	r
8	vv	vv2	11	vv

Chức năng xóa, người dùng nhập vào id của sách muốn xóa hoặc nhấp vào hàng dữ liệu bên dưới id sẽ lập tức được đưa lên entry phía trên để người dùng xóa dễ dàng hơn



Database App

File

Login Tìm kiếm sách Thêm sách Cập nhật sách Xóa sách Ngắt kết nối

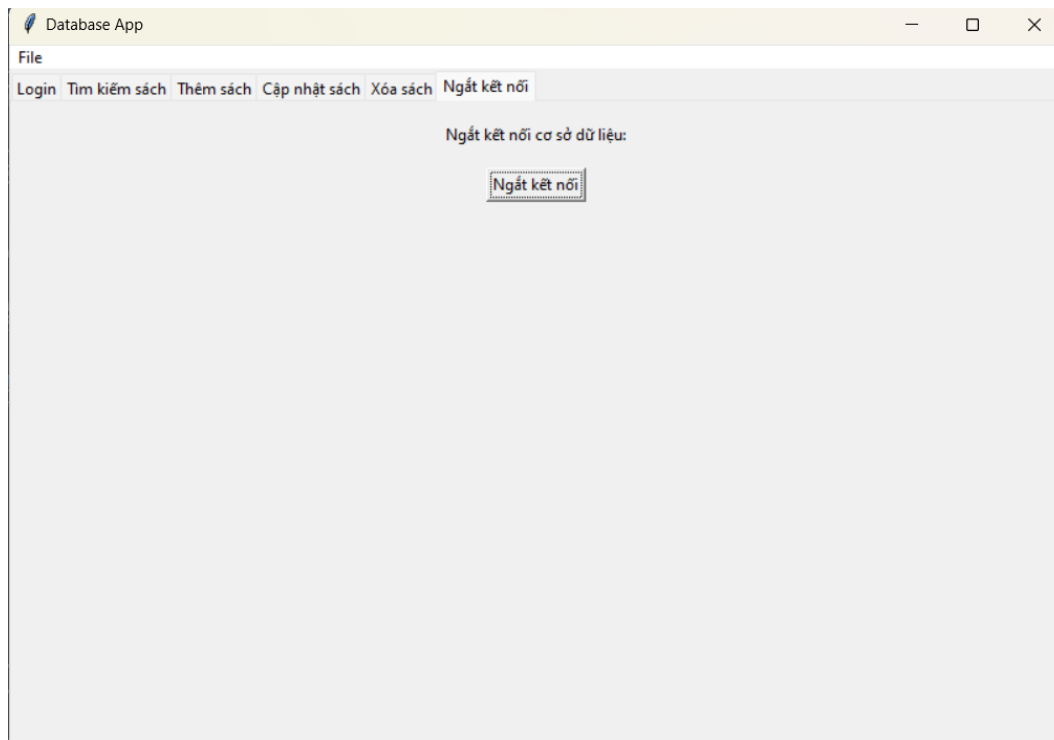
ID sách để xóa: 11

Xóa sách

Tải lại dữ liệu

ID	Tên sách	Tác giả	Năm xuất bản	Thể loại
11	vv	2	23	vv
7	vv	cc	22	vv
12	đ	r	22	r
8	vv	vv2	11	vv

Chức năng ngắt kết nối database, ấn vào button ngắt kết nối sẽ giúp bạn ngắt kết nối ra khỏi database và quay về trang login



Database App

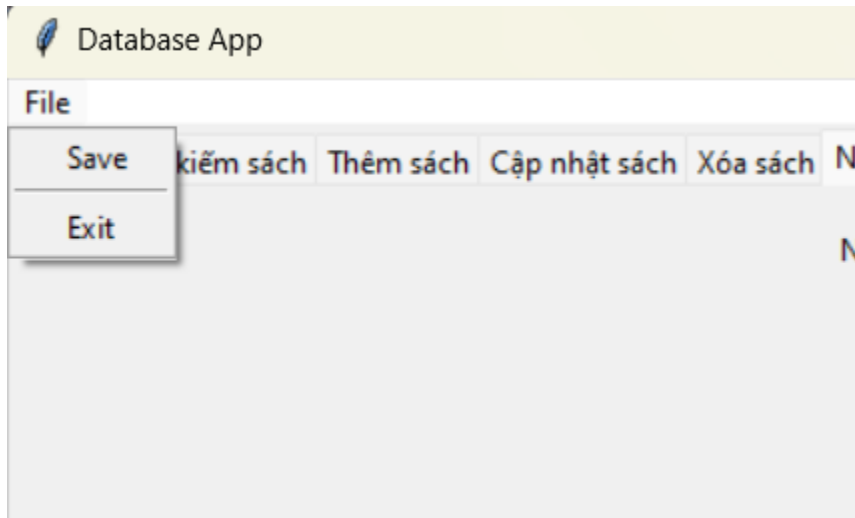
File

Login Tìm kiếm sách Thêm sách Cập nhật sách Xóa sách Ngắt kết nối

Ngắt kết nối cơ sở dữ liệu:

Ngắt kết nối

Ngoài ra giao diện còn có menu File bao gồm việc save file và exit ra khỏi giao diện



2. Chức năng

Đây là 1 giao diện quản lí database liên quan đến sách đơn giản với các chức năng như là:

- Đăng nhập vào database
- Tìm kiếm sách
- Thêm vào sách mới
- Cập nhật thông tin của sách
- Xóa sách
- Ngắt kết nối database
- Các thao tác trên cơ sở dữ liệu sau khi thành công đều sẽ hiện thông báo ra màn hình cũng như hiện thông báo khi có lỗi xảy ra
- Sử dụng sql.Identifier để tránh xảy ra các xung đột trong câu lệnh sql
- Đã xử lí các trường hợp bị block transaction

3. Mã nguồn

File app.py : dung để kết nối vào database cùng với các giao diện chính

```
import tkinter as tk
```

```

from tkinter import filedialog, messagebox
from tkinter import ttk
import psycopg2
from functions import SearchBooks, AddBooks, UpdateBooks, DeleteBooks,
DisconnectDB
class DatabaseApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Database App")

        # Tabs
        self.tab_control = ttk.Notebook(self.root)
        self.login_tab = tk.Frame(self.tab_control)
        self.search_tab = tk.Frame(self.tab_control)
        self.add_tab = tk.Frame(self.tab_control)
        self.update_tab = tk.Frame(self.tab_control)
        self.delete_tab = tk.Frame(self.tab_control)
        self.file_tab = tk.Frame(self.tab_control)

        self.tab_control.add(self.login_tab, text="Login")
        self.tab_control.add(self.search_tab, text="Tìm kiếm sách")
        self.tab_control.add(self.add_tab, text="Thêm sách")
        self.tab_control.add(self.update_tab, text="Cập nhật sách")
        self.tab_control.add(self.delete_tab, text="Xóa sách")

        self.tab_control.pack(expand=1, fill="both")

        # Hide tabs initially
        self.tab_control.hide(self.search_tab)
        self.tab_control.hide(self.add_tab)
        self.tab_control.hide(self.update_tab)
        self.tab_control.hide(self.delete_tab)

        # Database connection fields
        self.db_name = tk.StringVar(value='sach')
        self.user = tk.StringVar(value='postgres')
        self.password = tk.StringVar(value='1234567890')
        self.host = tk.StringVar(value='localhost')
        self.port = tk.StringVar(value='5432')
        self.table_name = tk.StringVar(value='sach')

        # Create the login interface
        self.create_login_widgets()

        # Create the add, update, delete interfaces

```

```

self.search_books = SearchBooks(self.search_tab, self)
self.add_books = AddBooks(self.add_tab, self)
self.update_books = UpdateBooks(self.update_tab, self)
self.delete_books = DeleteBooks(self.delete_tab, self)

self.disconnect_tab = tk.Frame(self.tab_control)
self.tab_control.add(self.disconnect_tab, text="Ngắt kết nối")
self.disconnect_db = DisconnectDB(self.disconnect_tab, self)

self.create_file_menu()
def create_file_menu(self):
    # Create a menu bar
    menubar = tk.Menu(self.root)

    # Create the File menu
    file_menu = tk.Menu(menubar, tearoff=0)
    file_menu.add_command(label="Save", command=self.save_data)
    file_menu.add_separator() # Optional separator
    file_menu.add_command(label="Exit", command=self.exit_app)

    # Add the File menu to the menubar
    menubar.add_cascade(label="File", menu=file_menu)

    # Configure the root window to use the menubar
    self.root.config(menu=menubar)

def save_data(self):
    # Mở hộp thoại lưu file
    file_path = filedialog.asksaveasfilename(
        defaultextension=".txt", # Định dạng mặc định
        filetypes=[("Text files", "*.txt"), ("All files", "*.*")], # Các
loại file hỗ trợ
        title="Lưu file"
    )

    if file_path: # Nếu người dùng đã chọn file
        try:
            with open(file_path, 'w', encoding='utf-8') as file:
                # Ví dụ: Lưu dữ liệu từ một biến hoặc danh sách
                data_to_save = "Dữ liệu ví dụ để lưu vào file" # Thay thế
bằng dữ liệu thực tế
                file.write(data_to_save)
            messagebox.showinfo("Save", "Dữ liệu đã được lưu thành công!")
        except Exception as e:
            messagebox.showerror("Error", f"Không thể lưu file: {e}")

```



```

def exit_app(self):
    # Hàm thoát ứng dụng
    self.root.quit()

def create_login_widgets(self):
    # Connection section in login tab
    connection_frame = tk.Frame(self.login_tab)
    connection_frame.pack(pady=10)

    tk.Label(connection_frame, text="DB Name:").grid(row=0, column=0, padx=5,
pady=5)
    tk.Entry(connection_frame, textvariable=self.db_name).grid(row=0,
column=1, padx=5, pady=5)

    tk.Label(connection_frame, text="User:").grid(row=1, column=0, padx=5,
pady=5)
    tk.Entry(connection_frame, textvariable=self.user).grid(row=1, column=1,
padx=5, pady=5)

    tk.Label(connection_frame, text="Password:").grid(row=2, column=0,
padx=5, pady=5)
    tk.Entry(connection_frame, textvariable=self.password,
show="*").grid(row=2, column=1, padx=5, pady=5)

    tk.Label(connection_frame, text="Host:").grid(row=3, column=0, padx=5,
pady=5)
    tk.Entry(connection_frame, textvariable=self.host).grid(row=3, column=1,
padx=5, pady=5)

    tk.Label(connection_frame, text="Port:").grid(row=4, column=0, padx=5,
pady=5)
    tk.Entry(connection_frame, textvariable=self.port).grid(row=4, column=1,
padx=5, pady=5)

    tk.Button(connection_frame, text="Connect",
command=self.connect_db).grid(row=5, columnspan=2, pady=10)

def connect_db(self):
    try:
        self.conn = psycopg2.connect(
            dbname=self.db_name.get(),
            user=self.user.get(),
            password=self.password.get(),
            host=self.host.get(),
            port=self.port.get()

```

```

    )
    self.cur = self.conn.cursor()
    messagebox.showinfo("Success", "Kết nối cơ sở dữ liệu thành công!")

    # Show search, add, update, delete book tabs
    self.tab_control.add(self.search_tab, text="Tìm kiếm sách")
    self.tab_control.add(self.add_tab, text="Thêm sách")
    self.tab_control.add(self.update_tab, text="Cập nhật sách")
    self.tab_control.add(self.delete_tab, text="Xóa sách")
    self.tab_control.select(self.search_tab)

    # Load data in all Treeviews
    self.search_books.load_data()
    self.add_books.load_data()
    self.update_books.load_data()
    self.delete_books.load_data()

    # Bind tab change event to load data
    self.tab_control.bind("<<NotebookTabChanged>>", self.on_tab_change)

except Exception as e:
    messagebox.showerror("Error", f"Lỗi xảy ra trong quá trình kết nối cơ
sở dữ liệu: {e}")

def on_tab_change(self, event):
    current_tab = event.widget.tab(event.widget.select(), "text")
    if current_tab == "Tìm kiếm sách":
        self.search_books.load_data()
    elif current_tab == "Thêm sách":
        self.add_books.load_data()
    elif current_tab == "Cập nhật sách":
        self.update_books.load_data()
    elif current_tab == "Xóa sách":
        self.delete_books.load_data()

```

File functions.py : dung để chứa các chức năng tìm kiếm sách, thêm sách, cập nhật sách, xóa sách và ngắt kết nối database

```

import tkinter as tk
from tkinter import messagebox
from tkinter import ttk
from psycopg2 import sql

```

```

class SearchBooks:
    def __init__(self, tab, app):
        self.tab = tab
        self.app = app

        search_frame = tk.Frame(self.tab)
        search_frame.pack(pady=10)

        self.search_var = tk.StringVar() # Tìm kiếm theo tên sách

        tk.Label(search_frame, text="Tìm kiếm theo tên sách:").grid(row=0,
column=0, padx=5, pady=5)
        tk.Entry(search_frame, textvariable=self.search_var).grid(row=0,
column=1, padx=5, pady=5)
        tk.Button(search_frame, text="Tìm kiếm",
command=self.search_data).grid(row=0, column=2, padx=5, pady=5)
        tk.Button(search_frame, text="Tải lại dữ liệu",
command=self.load_data).grid(row=0, column=3, padx=5, pady=5)

        # Treeview for displaying search results
        self.data_tree = ttk.Treeview(self.tab, columns=("ID", "Tên sách", "Tác
giả", "Năm xuất bản", "Thể loại"), show="headings", height=10)
        self.data_tree.column("ID", width=100, anchor="center")
        self.data_tree.column("Tên sách", width=200, anchor="center")
        self.data_tree.column("Tác giả", width=200, anchor="center")
        self.data_tree.column("Năm xuất bản", width=150, anchor="center")
        self.data_tree.column("Thể loại", width=150, anchor="center")
        self.data_tree.heading("ID", text="ID")
        self.data_tree.heading("Tên sách", text="Tên sách")
        self.data_tree.heading("Tác giả", text="Tác giả")
        self.data_tree.heading("Năm xuất bản", text="Năm xuất bản")
        self.data_tree.heading("Thể loại", text="Thể loại")
        self.data_tree.pack(side="bottom", fill="both", expand=True)

    def load_data(self):
        try:
            query = sql.SQL("SELECT * FROM
{}").format(sql.Identifier(self.app.table_name.get()))
            self.app.cur.execute(query)
            rows = self.app.cur.fetchall()

            # Clear previous data
            for item in self.data_tree.get_children():
                self.data_tree.delete(item)

```

```

        # Insert new data into Treeview
        for row in rows:
            self.data_tree.insert("", "end", values=row)
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")

    def search_data(self):
        try:
            search_query = sql.SQL("SELECT * FROM {} WHERE ten_sach ILIKE
%s").format(sql.Identifier(self.app.table_name.get()))
            self.app.cur.execute(search_query, ('%' + self.search_var.get() +
'%',))
            rows = self.app.cur.fetchall()

            # Clear previous data
            for item in self.data_tree.get_children():
                self.data_tree.delete(item)

            # Insert search results into Treeview
            for row in rows:
                self.data_tree.insert("", "end", values=row)
        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi tìm kiếm: {e}")

class AddBooks:
    def __init__(self, tab, app):
        self.tab = tab
        self.app = app

        # Insert section
        insert_frame = tk.Frame(self.tab)
        insert_frame.pack(pady=10)

        self.column1 = tk.StringVar() # Tên sách
        self.column2 = tk.StringVar() # Tác giả
        self.column3 = tk.StringVar() # NXB
        self.column4 = tk.StringVar() # Thể loại

        tk.Label(insert_frame, text="Tên sách:").grid(row=0, column=0, padx=5,
pady=5)
        tk.Entry(insert_frame, textvariable=self.column1).grid(row=0, column=1,
padx=5, pady=5)

```

```

        tk.Label(insert_frame, text="Tác giả:").grid(row=1, column=0, padx=5,
pady=5)
        tk.Entry(insert_frame, textvariable=self.column2).grid(row=1, column=1,
padx=5, pady=5)

        tk.Label(insert_frame, text="Năm xuất bản:").grid(row=2, column=0,
padx=5, pady=5)
        tk.Entry(insert_frame, textvariable=self.column3).grid(row=2, column=1,
padx=5, pady=5)

        tk.Label(insert_frame, text="Thể loại:").grid(row=3, column=0, padx=5,
pady=5)
        tk.Entry(insert_frame, textvariable=self.column4).grid(row=3, column=1,
padx=5, pady=5)

        tk.Button(insert_frame, text="Thêm sách",
command=self.insert_data).grid(row=4, columnspan=2, pady=10)
        tk.Button(insert_frame, text="Tải lại dữ liệu",
command=self.load_data).grid(row=5, columnspan=2, pady=10)

        # Treeview for displaying data
        self.data_tree = ttk.Treeview(self.tab, columns=("ID", "Tên sách", "Tác
giả", "Năm xuất bản", "Thể loại"), show="headings", height=10)
        self.data_tree.column("ID", width=100, anchor="center")
        self.data_tree.column("Tên sách", width=200, anchor="center")
        self.data_tree.column("Tác giả", width=200, anchor="center")
        self.data_tree.column("Năm xuất bản", width=150, anchor="center")
        self.data_tree.column("Thể loại", width=150, anchor="center")
        self.data_tree.heading("ID", text="ID")
        self.data_tree.heading("Tên sách", text="Tên sách")
        self.data_tree.heading("Tác giả", text="Tác giả")
        self.data_tree.heading("Năm xuất bản", text="Năm xuất bản")
        self.data_tree.heading("Thể loại", text="Thể loại")
        self.data_tree.pack(side="bottom", fill="both", expand=True)

    def load_data(self):
        try:
            query = sql.SQL("SELECT * FROM
{}").format(sql.Identifier(self.app.table_name.get()))
            self.app.cur.execute(query)
            rows = self.app.cur.fetchall()

            # Clear previous data
            for item in self.data_tree.get_children():
                self.data_tree.delete(item)

```

```

        # Insert new data into Treeview
        for row in rows:
            self.data_tree.insert("", "end", values=row)
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")

    def insert_data(self):
        try:
            if not self.column1.get() or not self.column2.get():
                messagebox.showwarning("Cảnh báo", "Vui lòng điền đầy đủ thông tin sách!")
            return

            add_query = sql.SQL("INSERT INTO {} (ten_sach, tac_gia, nxb, the_loai) VALUES (%s, %s, %s, %s)".format(sql.Identifier(self.app.table_name.get())))
            self.app.cur.execute(add_query, (self.column1.get(), self.column2.get(), self.column3.get(), self.column4.get()))
            self.app.conn.commit()

            messagebox.showinfo("Thành công", "Sách đã được thêm thành công!")
            self.load_data() # Refresh data after addition

        except Exception as e:
            self.app.conn.rollback() # Rollback the transaction on error
            messagebox.showerror("Lỗi", f"Lỗi thêm sách: {e}")

class UpdateBooks:
    def __init__(self, tab, app):
        self.tab = tab
        self.app = app

        # Insert section
        update_frame = tk.Frame(self.tab)
        update_frame.pack(pady=10)

        self.column1 = tk.StringVar() # ID (disabled)
        self.column2 = tk.StringVar() # Tên sách
        self.column3 = tk.StringVar() # Tác giả
        self.column4 = tk.StringVar() # NXB
        self.column5 = tk.StringVar() # Thể loại

```

```

        tk.Label(update_frame, text="ID:").grid(row=0, column=0, padx=5, pady=5)
        self.id_entry = tk.Entry(update_frame, textvariable=self.column1,
state='disabled') # ID disabled
        self.id_entry.grid(row=0, column=1, padx=5, pady=5)

        tk.Label(update_frame, text="Tên sách:").grid(row=1, column=0, padx=5,
pady=5)
        self.name_entry = tk.Entry(update_frame, textvariable=self.column2)
        self.name_entry.grid(row=1, column=1, padx=5, pady=5)

        tk.Label(update_frame, text="Tác giả:").grid(row=2, column=0, padx=5,
pady=5)
        self.author_entry = tk.Entry(update_frame, textvariable=self.column3)
        self.author_entry.grid(row=2, column=1, padx=5, pady=5)

        tk.Label(update_frame, text="Năm xuất bản:").grid(row=3, column=0,
padx=5, pady=5)
        self.nxb_entry = tk.Entry(update_frame, textvariable=self.column4)
        self.nxb_entry.grid(row=3, column=1, padx=5, pady=5)

        tk.Label(update_frame, text="Thể loại:").grid(row=4, column=0, padx=5,
pady=5)
        self.the_loai_entry = tk.Entry(update_frame, textvariable=self.column5)
        self.the_loai_entry.grid(row=4, column=1, padx=5, pady=5)

        tk.Button(update_frame, text="Cập nhật dữ liệu",
command=self.update_data).grid(row=5, columnspan=2, pady=10)
        tk.Button(update_frame, text="Tải lại dữ liệu",
command=self.load_data).grid(row=6, columnspan=2, pady=10)

        # Treeview for displaying data
        self.data_tree = ttk.Treeview(self.tab, columns=("ID", "Tên sách", "Tác
giả", "Năm xuất bản", "Thể loại"), show="headings", height=10)
        self.data_tree.column("ID", width=100, anchor="center")
        self.data_tree.column("Tên sách", width=200, anchor="center")
        self.data_tree.column("Tác giả", width=200, anchor="center")
        self.data_tree.column("Năm xuất bản", width=150, anchor="center")
        self.data_tree.column("Thể loại", width=150, anchor="center")
        self.data_tree.heading("ID", text="ID")
        self.data_tree.heading("Tên sách", text="Tên sách")
        self.data_tree.heading("Tác giả", text="Tác giả")
        self.data_tree.heading("Năm xuất bản", text="Năm xuất bản")
        self.data_tree.heading("Thể loại", text="Thể loại")
        self.data_tree.pack(side="bottom", fill="both", expand=True)

```

```

# Bind Treeview click event
self.data_tree.bind("<ButtonRelease-1>", self.select_item)

def load_data(self):
    try:
        query = sql.SQL("SELECT * FROM
{}".format(sql.Identifier(self.app.table_name.get())))
        self.app.cur.execute(query)
        rows = self.app.cur.fetchall()

        # Clear previous data
        for item in self.data_tree.get_children():
            self.data_tree.delete(item)

        # Insert new data into Treeview
        for row in rows:
            self.data_tree.insert("", "end", values=row)
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")

def select_item(self, event):
    # Get selected item from Treeview
    selected_item = self.data_tree.selection()[0]
    item_values = self.data_tree.item(selected_item, 'values')

    # Set the selected values into the Entry widgets
    self.column1.set(item_values[0]) # ID
    self.column2.set(item_values[1]) # Tên sách
    self.column3.set(item_values[2]) # Tác giả
    self.column4.set(item_values[3]) # NXB
    self.column5.set(item_values[4]) # Thể loại

def update_data(self):
    try:
        if not self.column1.get():
            messagebox.showwarning("Cảnh báo", "Vui lòng nhập ID của sách cần
sửa!")

        return

        update_query = sql.SQL("UPDATE {} SET ten_sach=%s, tac_gia=%s,
nxb=%s, the_loai=%s WHERE
id=%s").format(sql.Identifier(self.app.table_name.get()))
        self.app.cur.execute(update_query, (self.column2.get(),
self.column3.get(), self.column4.get(), self.column5.get(), self.column1.get()))

```



```

        if self.app.cur.rowcount == 0: # Kiểm tra số bản ghi bị ảnh hưởng
            messagebox.showwarning("Cảnh báo", "Không có ID nào trùng khớp để
sửa!")
        else:
            self.app.conn.commit()
            messagebox.showinfo("Thành công", "Sách đã được sửa thành công!")
            self.load_data() # Refresh data after update

    except Exception as e:
        self.app.conn.rollback() # Rollback the transaction on error
        messagebox.showerror("Lỗi", f"Lỗi sửa sách: {e}")

class DeleteBooks:
    def __init__(self, tab, app):
        self.tab = tab
        self.app = app

        delete_frame = tk.Frame(self.tab)
        delete_frame.pack(pady=10)

        self.id_var = tk.StringVar() # ID

        tk.Label(delete_frame, text="ID sách để xóa:").grid(row=0, column=0,
padx=5, pady=5)
        self.id_entry = tk.Entry(delete_frame, textvariable=self.id_var)
        self.id_entry.grid(row=0, column=1, padx=5, pady=5)

        tk.Button(delete_frame, text="Xóa sách",
command=self.delete_data).grid(row=1, columnspan=2, pady=10)
        tk.Button(delete_frame, text="Tải lại dữ liệu",
command=self.load_data).grid(row=2, columnspan=2, pady=10)

        # Treeview for displaying data
        self.data_tree = ttk.Treeview(self.tab, columns=("ID", "Tên sách", "Tác
giả", "Năm xuất bản", "Thể loại"), show="headings", height=10)
        self.data_tree.column("ID", width=100, anchor="center")
        self.data_tree.column("Tên sách", width=200, anchor="center")
        self.data_tree.column("Tác giả", width=200, anchor="center")
        self.data_tree.column("Năm xuất bản", width=150, anchor="center")
        self.data_tree.column("Thể loại", width=150, anchor="center")
        self.data_tree.heading("ID", text="ID")
        self.data_tree.heading("Tên sách", text="Tên sách")
        self.data_tree.heading("Tác giả", text="Tác giả")
        self.data_tree.heading("Năm xuất bản", text="Năm xuất bản")

```

```

self.data_tree.heading("Thẻ loại", text="Thẻ loại")
self.data_tree.pack(side="bottom", fill="both", expand=True)

# Bind double-click event
self.data_tree.bind("<ButtonRelease-1>", self.on_treeview_select)

def on_treeview_select(self, event):
    # Get selected item
    selected_item = self.data_tree.selection()
    if selected_item:
        item_values = self.data_tree.item(selected_item, "values")
        if item_values:
            self.id_var.set(item_values[0]) # Set ID vào entry

def load_data(self):
    try:
        query = sql.SQL("SELECT * FROM
{}".format(sql.Identifier(self.app.table_name.get())))
        self.app.cur.execute(query)
        rows = self.app.cur.fetchall()

        # Clear previous data
        for item in self.data_tree.get_children():
            self.data_tree.delete(item)

        # Insert new data into Treeview
        for row in rows:
            self.data_tree.insert("", "end", values=row)
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")

def delete_data(self):
    try:
        delete_query = sql.SQL("DELETE FROM {} WHERE
id=%s").format(sql.Identifier(self.app.table_name.get()))
        self.app.cur.execute(delete_query, (self.id_var.get(),))

        if self.app.cur.rowcount == 0: # Kiểm tra số bản ghi bị ảnh hưởng
            messagebox.showwarning("Cảnh báo", "Không có ID nào trùng khớp!")
        else:
            self.app.conn.commit()
            messagebox.showinfo("Thành công", "Sách đã được xóa thành công!")
            self.load_data() # Refresh data after deletion

    except Exception as e:

```

```

        self.app.conn.rollback() # Rollback the transaction on error
        messagebox.showerror("Lỗi", f"Lỗi xóa sách: {e}")

class DisconnectDB:
    def __init__(self, tab, app):
        self.tab = tab
        self.app = app

        disconnect_frame = tk.Frame(self.tab)
        disconnect_frame.pack(pady=10)

        tk.Label(disconnect_frame, text="Ngắt kết nối cơ sở dữ  
liệu:").grid(row=0, column=0, padx=5, pady=5)
        tk.Button(disconnect_frame, text="Ngắt kết nối",  
command=self.disconnect_data).grid(row=1, columnspan=2, pady=10)

    def disconnect_data(self):
        try:
            if hasattr(self.app, 'conn') and self.app.conn is not None:
                self.app.cur.close()
                self.app.conn.close()
                messagebox.showinfo("Thành công", "Đã ngắt kết nối cơ sở dữ  
liệu!")

            # Quay về tab đăng nhập
            self.app.tab_control.select(self.app.login_tab)

            # Ẩn các tab khác sau khi ngắt kết nối
            self.app.tab_control.hide(self.app.search_tab)
            self.app.tab_control.hide(self.app.add_tab)
            self.app.tab_control.hide(self.app.update_tab)
            self.app.tab_control.hide(self.app.delete_tab)

        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi ngắt kết nối: {e}")

```

File main.py : dung để xử lí toàn bộ code và chạy giao diện

```

import tkinter as tk
from tkinter import ttk
from app import DatabaseApp

def main():
    root = tk.Tk()
    app = DatabaseApp(root)

```

```
root.mainloop()

if __name__ == "__main__":
    main()
```

4. Git-hub

Link: [vuviet1207/LapTrinhPythonNangCao_BaiTap2](https://github.com/vuviet1207/LapTrinhPythonNangCao_BaiTap2)