

BÁO CÁO PYTHON NÂNG CAO_BAITAP 2

Tên: PHIMMACHAK Lattakone

Mssv: 227480201is001

1. Giải đoạn code

Bài tập này phát triển một ứng dụng quản lý thư viện sách sử dụng giao diện đồ họa với Tkinter và kết nối cơ sở dữ liệu PostgreSQL thông qua thư viện psycopg2. Ứng dụng cho phép thêm, cập nhật, xóa và tải lại danh sách sách từ cơ sở dữ liệu.

Cụ thể, các chức năng chính được chia thành những phần chính sau:

- **Kết nối cơ sở dữ liệu:** Hàm connect_db đảm bảo kết nối với cơ sở dữ liệu PostgreSQL.

```
def connect_db(dbname='postgres'):  
    try:  
        conn = psycopg2.connect(  
            dbname="Lattakone",  
            user="postgres",  
            password="1234",  
            host="localhost",  
            port='5432'  
        )
```

- **Tạo cơ sở dữ liệu:** Hàm create_database tạo cơ sở dữ liệu có tên là "books".
- **Tạo bảng sách:** Hàm create_table tạo bảng trong cơ sở dữ liệu để lưu trữ thông tin về sách.
- **Thêm sách:** Hàm add_book thêm thông tin sách mới vào cơ sở dữ liệu.

- **Cập nhật thông tin sách:** Hàm `update_book` cập nhật thông tin sách dựa trên lựa chọn từ danh sách hiển thị.
- **Xóa sách:** Hàm `delete_book` xóa sách đã chọn khỏi cơ sở dữ liệu.
- **Tải lại danh sách sách:** Hàm `reload_books` tải lại toàn bộ danh sách sách từ cơ sở dữ liệu và hiển thị trên giao diện.
- **Giao diện người dùng:** Sử dụng Tkinter để thiết kế giao diện với các thành phần như Label, Entry, Button, và Treeview để hiển thị danh sách sách.

2. Chức năng của code

Chức năng chính:

- **Kết nối cơ sở dữ liệu:** Kết nối đến cơ sở dữ liệu PostgreSQL sử dụng thông tin đăng nhập (username, password, host, port).
- **Tạo bảng:** Nếu chưa có bảng "books" trong cơ sở dữ liệu, ứng dụng sẽ tạo bảng với các cột bao gồm id, title, author, year, và genre.
- **Thêm sách:** Cho phép người dùng nhập thông tin về sách (title, author, year, genre) và lưu trữ chúng vào cơ sở dữ liệu.
- **Cập nhật sách:** Cập nhật thông tin sách dựa trên ID của sách đã chọn từ danh sách.
- **Xóa sách:** Xóa sách khỏi cơ sở dữ liệu dựa trên ID của sách đã chọn.
- **Hiển thị sách:** Tải toàn bộ dữ liệu từ cơ sở dữ liệu và hiển thị trên giao diện dưới dạng bảng.
- **Giao diện:** Giao diện người dùng sử dụng Tkinter, với các ô nhập liệu, các nút chức năng (Thêm, Sửa, Xóa, Tải lại) và bảng hiển thị danh sách sách.

3. Mã nguồn

```
import psycopg2
import tkinter as tk
from tkinter import ttk, messagebox
```

```
# Database connection
def connect_db(dbname='postgres'):
    try:
        conn = psycopg2.connect(
            dbname="Lattakone",
            user="postgres",
            password="1234",
            host="localhost",
            port='5432'
        )
        print(f"Connection to {dbname}
successful")
        return conn
    except Exception as e:
        print(f"Error while connecting to
database {dbname}:", e)
        raise

def create_database():
    try:
        conn = connect_db()
        conn.set_isolation_level(psycopg2.e
xtensions.ISOLATION_LEVEL_AUTOCOMMIT)
        cur = conn.cursor()
```

```
        cur.execute("CREATE DATABASE
books")
        cur.close()
        conn.close()
        print("Database 'books' created
successfully")
    except
psycopg2.errors.DuplicateDatabase:
        print("Database 'books' already
exists")
    except Exception as e:
        print("Error creating database:",
e)

        raise

def create_table(connection):
    if connection is None:
        messagebox.showerror("Database
Error", "Unable to connect to the
database")
        return

    try:
        cursor = connection.cursor()
```

```
        cursor.execute("""
            CREATE TABLE IF NOT EXISTS
books (
                id SERIAL PRIMARY KEY,
                title VARCHAR(255) NOT
NULL,
                author VARCHAR(255) NOT
NULL,
                year INTEGER NOT NULL,
                genre VARCHAR(100) NOT NULL
            );
        """)
        connection.commit()
        print("Table 'books' has been
created successfully.")
    except Exception as e:
        print("Error while creating
table:", e)
        messagebox.showerror("Database
Error", str(e))
    finally:
        if cursor:
            cursor.close()
```

```
# Add book to the database
def add_book():
    title = entry_title.get()
    author = entry_author.get()
    year = entry_year.get()
    genre = entry_genre.get()

    if not validate_input(title, author,
year, genre):
        return

    try:
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("INSERT INTO books
(title, author, year, genre) VALUES (%s,
%s, %s, %s)",
                    (title, author,
int(year), genre))
        conn.commit()
        cur.close()
        conn.close()
        reload_books()
    except Exception as e:
```

```
        messagebox.showerror("Database
Error", str(e))

# Update selected book
def update_book():
    try:
        selected_item = tree.focus()
        if not selected_item:
            messagebox.showwarning("Selecti
on Error", "Please select a book to
update.")
            return

        book_id =
tree.item(selected_item)['values'][0]
        title = entry_title.get()
        author = entry_author.get()
        year = entry_year.get()
        genre = entry_genre.get()

        if not validate_input(title,
author, year, genre):
            return
```

```

        conn = connect_db()
        cur = conn.cursor()
        cur.execute("UPDATE books SET
title=%s, author=%s, year=%s, genre=%s
WHERE id=%s",
                    (title, author,
int(year), genre, book_id))
        conn.commit()
        cur.close()
        conn.close()
        reload_books()
    except Exception as e:
        messagebox.showerror("Update
Error", str(e))

# Delete selected book
def delete_book():
    try:
        selected_item = tree.focus()
        if not selected_item:
            messagebox.showwarning("Selecti
on Error", "Please select a book to
delete.")
        return

```



```
        book_id =
tree.item(selected_item)['values'][0]

        conn = connect_db()
        cur = conn.cursor()
        cur.execute("DELETE FROM books
WHERE id=%s", (book_id,))
        conn.commit()
        cur.close()
        conn.close()
        reload_books()
    except Exception as e:
        messagebox.showerror("Delete
Error", str(e))

# Reload the book list from database
def reload_books():
    try:
        conn = connect_db('books')
        cur = conn.cursor()

        # Kiểm tra xem bảng có tồn tại
không
```

```

        cur.execute("""
            SELECT EXISTS (
                SELECT FROM
information_schema.tables
                WHERE table_name = 'books'
            );
        """)
        table_exists = cur.fetchone()[0]

        if not table_exists:
            create_table(conn)

        cur.execute("SELECT * FROM books")
        rows = cur.fetchall()

        # Clear existing data
        for row in tree.get_children():
            tree.delete(row)

        # Insert new data
        for row in rows:
            tree.insert("", "end",
values=row)
        except Exception as e:

```

```
        messagebox.showerror("Load Error",
str(e))
    finally:
        if cur:
            cur.close()
        if conn:
            conn.close()

# Validate input fields
def validate_input(title, author, year,
genre):
    if not title or not author or not year
or not genre:
        messagebox.showwarning("Input
Error", "All fields are required.")
        return False
    try:
        int(year)
    except ValueError:
        messagebox.showwarning("Input
Error", "Year must be a number.")
        return False
    return True
```

```
# Setup GUI
root = tk.Tk()
root.title("Library Management")

# Labels and entries
tk.Label(root, text="Book
Title:").grid(row=0, column=0, padx=10,
pady=5)
entry_title = tk.Entry(root)
entry_title.grid(row=0, column=1, padx=10,
pady=5)

tk.Label(root, text="Author:").grid(row=1,
column=0, padx=10, pady=5)
entry_author = tk.Entry(root)
entry_author.grid(row=1, column=1, padx=10,
pady=5)

tk.Label(root, text="Year XB:").grid(row=2,
column=0, padx=10, pady=5)
entry_year = tk.Entry(root)
entry_year.grid(row=2, column=1, padx=10,
pady=5)
```

```
tk.Label(root,  
text="Category:").grid(row=3, column=0,  
padx=10, pady=5)  
entry_genre = tk.Entry(root)  
entry_genre.grid(row=3, column=1, padx=10,  
pady=5)  
  
# Buttons  
tk.Button(root, text="Add books",  
command=add_book).grid(row=4, column=0,  
padx=10, pady=10)  
tk.Button(root, text="Update information",  
command=update_book).grid(row=4, column=1,  
padx=10, pady=10)  
tk.Button(root, text="Delete book",  
command=delete_book).grid(row=4, column=2,  
padx=10, pady=10)  
tk.Button(root, text="Reload list",  
command=reload_books).grid(row=4, column=3,  
padx=10, pady=10)  
  
# Treeview for book list  
columns = ("ID", "Book Title", "Author",  
"year XB", "Category")
```

```
tree = ttk.Treeview(root, columns=columns,
show="headings")
tree.heading("ID", text="ID")
tree.heading("Book Title", text="Book
Title")
tree.heading("Author", text="Author")
tree.heading("year XB", text="year XB")
tree.heading("Category", text="Category")
tree.grid(row=5, column=0, columnspan=4,
padx=10, pady=10)

try:

    connection = connect_db('books')
    if connection:
        create_table(connection)
        connection.close()
except Exception as e:
    messagebox.showerror("Database Error",
f"Unable to setup the database: {str(e)}")
    root.quit()

# Start with a load of books
reload_books()
```

```
# Run the application
root.mainloop()
```

Kết quả code:

1.Add book

Library Management

Book Title:

Author:

Year XB:

Category:

ID	Book Title	Author	year XB	Category
6	Lattakone	PHIMMACHAK	2024	4
7	Vansanah	Somsavanh	2024	4
8	Phongsavanh	Banmanivong	2024	4
9	Anousith	Keosisai	2024	4
10	Thongsavanh	Thoummavong	2024	4
11	Soukphansa	KYC	2024	4
12	Khounmy	KYC	2024	4
13	Temsak	Thatsanavong	2024	4
14	Ananh	mithong	2024	4

2. Update information

Library Management

Book Title:

Author:

Year XB:

Category:

ID	Book Title	Author	year XB	Category
6	Lattakone	PHIMMACHAK	2024	4
7	Vansanah	Somsavanh	2024	4
8	Phongsavanh	Banmanivong	2024	4
9	Anousith	Keosisai	2024	4
10	Thongsavanh	Thoummavong	2024	4
11	Soukphansa	KYC	2024	4
12	Khounmy	KYC	2024	4
13	Temsak	Thatsanavong	2024	4
14	Ananh	mithong	2025	4444444

3. Delete book

Click vào list tên mà mình muốn Xóa, sau đó Bấm nút delete book

Library Management

Book Title:

Author:

Year XB:

Category:

ID	Book Title	Author	year XB	Category
6	Lattakone	PHIMMACHAK	2024	4
7	Vansanah	Somsavanh	2024	4
8	Phongsavanh	Banmanivong	2024	4
9	Anousith	Keosisai	2024	4
10	Thongsavanh	Thoummavong	2024	4
11	Soukphansa	KYC	2024	4
12	Khounmy	KYC	2024	4
13	Temsak	Thatsanavong	2024	4

4. MySQL Data base

Object Explorer

Servers (1)

- PostgreSQL 17
 - Databases (5)
 - Lattakone
 - Casts
 - Catalogs (2)
 - ANSI (information_schema)
 - PostgreSQL Catalog (pg_catalog)
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables

Dashboard x Properties x SQL x Statistics x Dependencies x Dependents x Processes x Lattakone/postgres@PostgreSQL 17* x

Lattakone/postgres@PostgreSQL 17

No limit

Data Output Messages Notifications

	id [PK] integer	title character varying (255)	author character varying (255)	year integer	genre character varying (100)
1	6	Lattakone	PHIMMACHAK	2024	4
2	7	Vansanah	Somsavanh	2024	4
3	8	Phongsavanh	Banmanivong	2024	4
4	9	Anousith	Keosisai	2024	4
5	10	Thongsavanh	Thoummavong	2024	4
6	11	Soukphansa	KYC	2024	4
7	12	Khounmy	KYC	2024	4
8	13	Temsak	Thatsanavong	2024	4

4. Đưa lên GitHub

[Python_NangCao/Project_Bc_Python_nang_cao/Code_Project2.py](https://github.com/Python_NangCao/Project_Bc_Python_nang_cao/Code_Project2.py) at main · Lattakone1/Python_NangCao (github.com)