

TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN



BÀI TẬP 3

MÔN HỌC LẬP TRÌNH PYTHON NÂNG CAO

Đề bài:

**LẬP TRÌNH ỨNG DỤNG QUẢN LÝ CÓ KẾT
NỐI DATABASE CHẠY TRÊN WEB FLASK**

SINH VIÊN THỰC HIỆN:

NGUYỄN TIỀN PHÚC 2274802010685

LỚP: 241_71ITSE31003_02

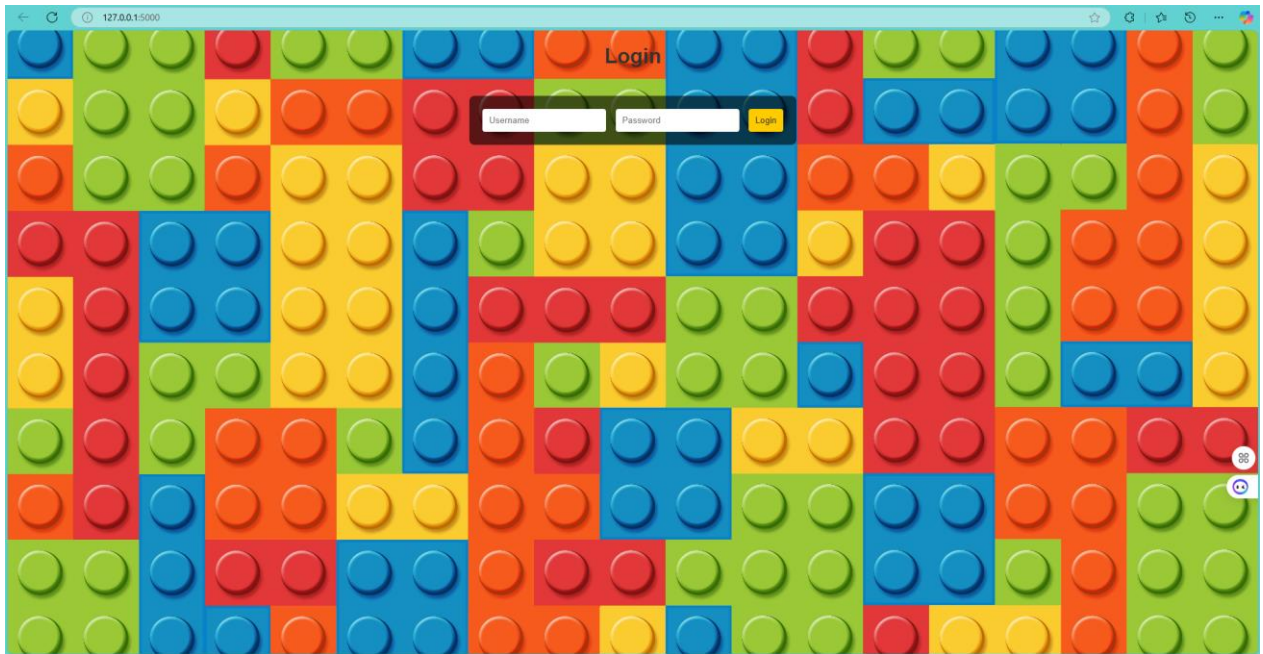
GVHD: HUỲNH THÁI HỌC

TP. Hồ Chí Minh – 18/10/2024

MỤC LỤC

I.	GIAO DIỆN	3
II.	CHỨC NĂNG.....	5
1.	Login	5
2.	Loading.....	6
3.	Kết nối database và thực thi lệnh	6
III.	MÃ CHƯƠNG TRÌNH.....	9
IV.	GITHUB.....	16

I. GIAO DIỆN



Themes

Theme	Actions
Architecture	Edit Delete
City	Edit Delete
Classic	Edit Delete
Creator 3in1	Edit Delete
Creator Expert	Edit Delete
DC	Edit Delete
Disney	Edit Delete
Friends	Edit Delete
Ideas	Edit Delete
Jurassic World	Edit Delete
Marvel	Edit Delete
Minecraft	Edit Delete
Technic	Edit Delete

Add New Theme [Add Theme](#)

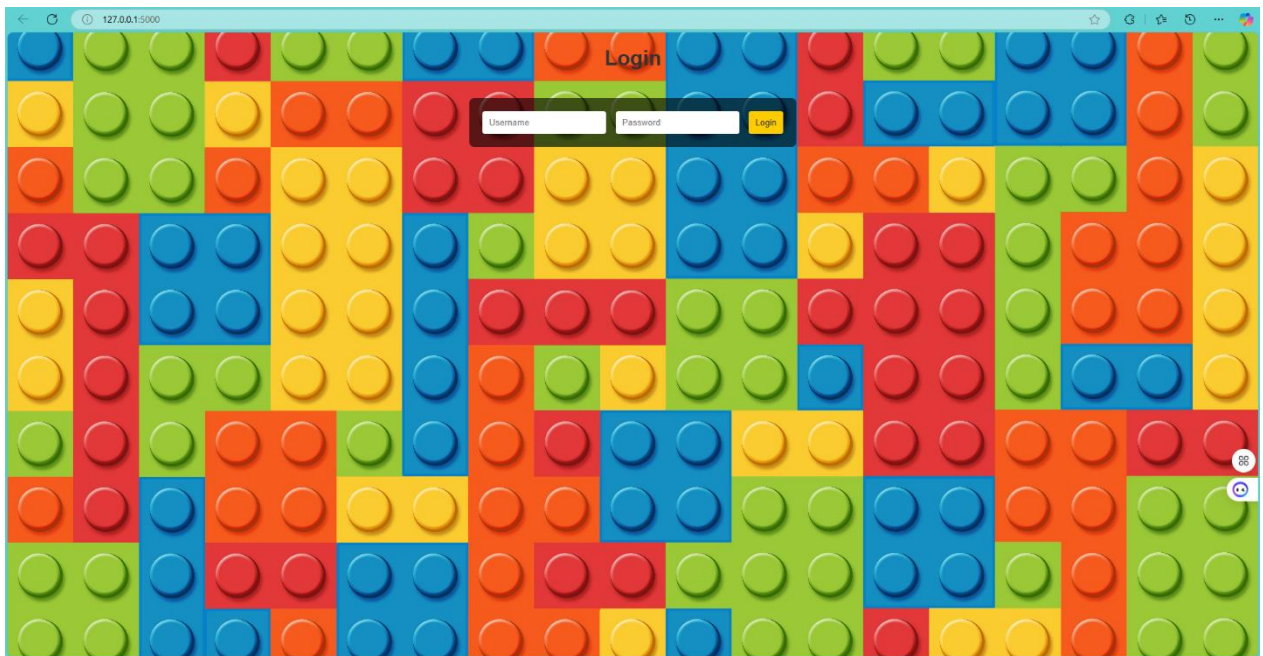
Lego Sets

ID	Theme	Name	Actions
10297	Creator Expert	Boutique Hotel	Edit Delete
10302	Creator Expert	Optimus Prime	Edit Delete
21042	Architecture	Statue of Liberty	Edit Delete
21060	Architecture	Himeji Castle	Edit Delete
21061	Architecture	Notre-Dame de Paris	Edit Delete
42154	Technic	2022 Ford GT	Edit Delete
42159	Technic	Yamaha MT-10 SP	Edit Delete
42160	Technic	Audi RS Q e-tron	Edit Delete
60368	City	Arctic Explorer Ship	Edit Delete
60440	City	Yellow Delivery Truck	Edit Delete
76269	Marvel	Avengers Tower	Edit Delete

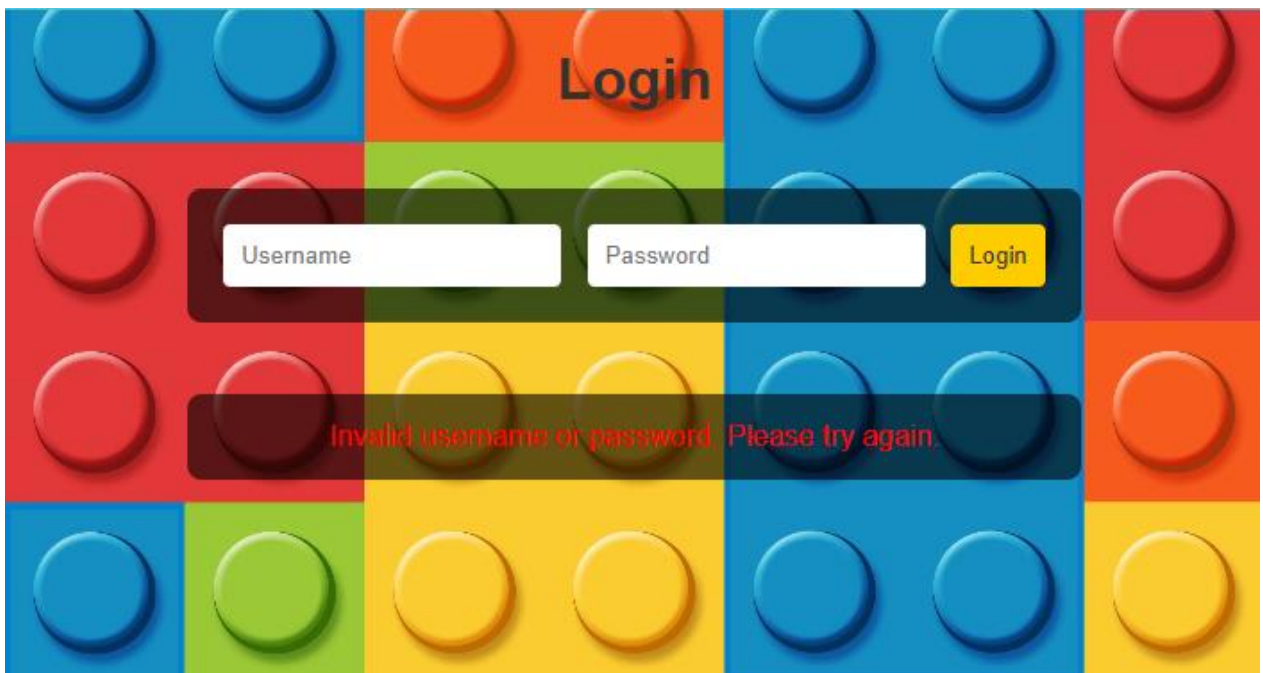
ID Theme Set Name [Add Lego Set](#)

II. CHỨC NĂNG

1. Login



Chức năng Login yêu cầu người dùng đăng nhập trước khi vào được trang ứng dụng chính. Nếu người dùng nhập username hoặc password không hợp lệ thì sẽ hiện dòng chữ màu đỏ nhắc nhở.



Sau khi người dùng nhập đúng username và password thì trang đăng nhập sẽ đóng và sẽ dẫn đến trang tiếp theo là Loading.

2. Loading



Mục đích của trang này là giống như chờ để đăng nhập vào trang ứng dụng chính. Trang này sử dụng hình ảnh background được cài đặt phóng to thu nhỏ theo kích thước của cửa sổ và loading trong vòng khoảng 2 giây sẽ chuyển hướng đến trang ứng dụng chính.

3. Kết nối database và thực thi lệnh

Lego Store Management		Logout
Themes		
Theme	Actions	
Architecture	Edit Delete	
City	Edit Delete	
Classic	Edit Delete	
Creator 3in1	Edit Delete	
Creator Expert	Edit Delete	
DC	Edit Delete	
Disney	Edit Delete	
Friends	Edit Delete	
Ideas	Edit Delete	
Jurassic World	Edit Delete	
Marvel	Edit Delete	
Minecraft	Edit Delete	
Technic	Edit Delete	
<input type="text" value="Add New Theme"/> Add Theme		

Lego Sets

ID	Theme	Name	Actions
10297	Creator Expert	Boutique Hotel	Edit Delete
10302	Creator Expert	Optimus Prime	Edit Delete
21042	Architecture	Statue of Liberty	Edit Delete
21060	Architecture	Himeji Castle	Edit Delete
21061	Architecture	Notre-Dame de Paris	Edit Delete
42154	Technic	2022 Ford GT	Edit Delete
42159	Technic	Yamaha MT-10 SP	Edit Delete
42160	Technic	Audi RS Q e-tron	Edit Delete
60368	City	Arctic Explorer Ship	Edit Delete
60440	City	Yellow Delivery Truck	Edit Delete
76269	Marvel	Avengers Tower	Edit Delete

ID Theme Set Name [Add Lego Set](#)

© 2024 Lego Store Management

Sau khi Loading hết 2 giây thì sẽ chuyển hướng đến trang ứng dụng chính. Ứng dụng quản lý này kết nối với database “lego” với 2 bảng: “themes” và “legoset”. Bảng “themes” dùng để lưu trữ tên chủ đề của 1 bộ Lego. Bảng “legoset” dùng để lưu trữ thông tin của 1 bộ Lego bao gồm ID, chủ đề và tên.

Khi người dùng thêm 1 chủ đề mới, dữ liệu được thêm vào “themes” sẽ hiển thị trong ComboBox chủ đề của bảng “legoset” để người dùng lựa chọn.

21061	Architecture
42154	Technic
42159	Technic
42160	Technic
60368	
60440	
76269	Marvel

Architecture

City

Classic

Creator 3in1

Creator Expert

DC

Disney

Friends

Ideas

Jurassic World

Marvel

Minecraft

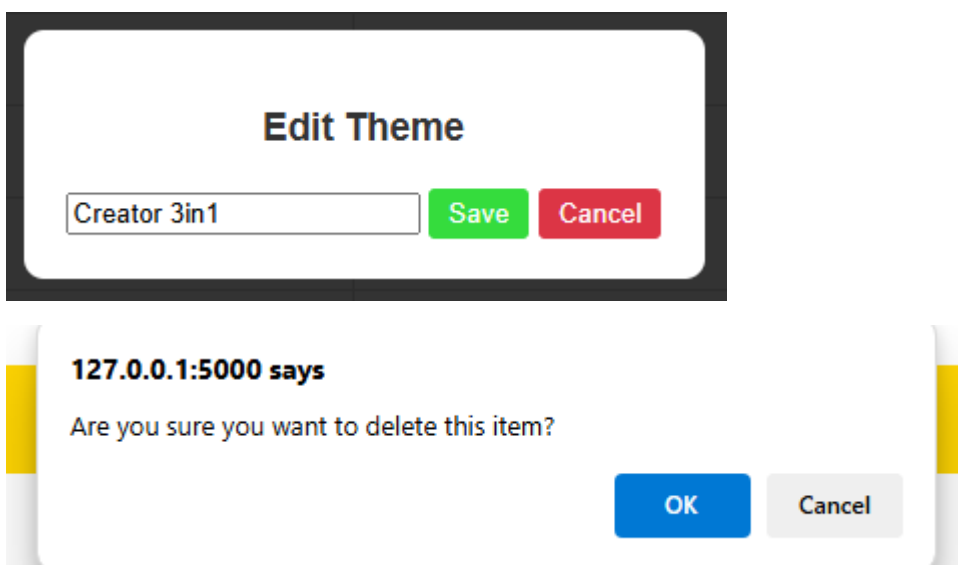
Technic

ID Set Name [Add Lego Set](#)

Khi người dùng muốn thêm thông tin của 1 bộ Lego mới, người dùng sẽ nhập ID (ô này không cho phép nhập chữ), chọn tên chủ đề, nhập tên của bộ Lego và nhấn nút

“Add Lego Set”. Người dùng cũng có thể nhấn vào tên cột để sắp xếp dữ liệu theo A-Z.

Trong bảng “themes”, người dùng cũng có thể sửa hoặc xóa chủ đề mình muốn. Khi người dùng nhấn “Edit” thì sẽ hiện 1 cửa sổ chỉnh sửa để người dùng có thể thay đổi tên của chủ đề. Và khi người dùng nhấn “Delete” thì sẽ hiển thị 1 thông báo hỏi người dùng có chắc chắn muốn xóa không. Tương tự đối với bảng “legoset”.



Edit Theme

Creator 3in1

127.0.0.1:5000 says

Are you sure you want to delete this item?

III. MÃ CHƯƠNG TRÌNH

```
from flask import Flask, render_template, request, redirect,
url_for, flash, session
import psycopg2
from psycopg2 import sql

app = Flask(__name__)
app.secret_key = "your_secret_key"

#Thiết lập kết nối với database
DB_CONFIG = {
    'dbname': 'lego',
    'user': 'postgres',
    'password': '123456',
    'host': 'localhost',
    'port': '5432'
}

def get_db_connection():
    return psycopg2.connect(**DB_CONFIG)

#Login Route
@app.route('/', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        if username == DB_CONFIG['user'] and password ==
DB_CONFIG['password']:
            session['logged_in'] = True
            return redirect(url_for('loading'))
        else:
            return render_template('login.html', title='Login',
error=True)
        return render_template('login.html', title='Login', error=False)

#Loading screen
@app.route('/loading')
def loading():
```

```

    if 'logged_in' not in session:
        return redirect(url_for('login'))
    return render_template('loading.html', title='Loading')

#Main Application Route
@app.route('/main', methods=['GET', 'POST'])
def main():
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    connection = get_db_connection()
    cursor = connection.cursor()

    #Thêm theme
    if request.method == 'POST' and 'new_theme' in request.form:
        new_theme = request.form['new_theme']
        try:
            cursor.execute(sql.SQL("INSERT INTO themes (theme)
VALUES (%s)"), [new_theme])
            connection.commit()
            flash("Theme added successfully!", "success")
        except Exception as e:
            connection.rollback()
            flash(f"Error adding theme: {e}", "danger")

    #Lấy dữ liệu từ themes và legoset để hiển thị
    cursor.execute("SELECT row_number() OVER () as id, theme FROM
themes ORDER BY theme ASC")
    themes = cursor.fetchall()

    cursor.execute("SELECT id, theme, name FROM legoset ORDER BY id
ASC")
    lego_sets = cursor.fetchall()

    connection.close()

    return render_template(
        'main.html',
        themes=themes,
        lego_sets=lego_sets,

```

```

        title='Lego Store Management'
    )

@app.route('/theme/edit', methods=['POST'])
def edit_theme():
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    theme_id = request.form['theme_id']
    theme_name = request.form['theme_name']

    connection = get_db_connection()
    cursor = connection.cursor()
    try:
        cursor.execute("UPDATE themes SET theme = %s WHERE theme = %s", (theme_name, theme_id))
        connection.commit()
        flash("Theme updated successfully!", "success")
    except Exception as e:
        connection.rollback()
        flash(f"Error editing theme: {e}", "danger")
    connection.close()
    return redirect(url_for('main'))

@app.route('/theme/delete/<theme>')
def delete_theme(theme):
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    connection = get_db_connection()
    cursor = connection.cursor()
    try:
        cursor.execute("DELETE FROM themes WHERE theme = %s",
            (theme,))
        connection.commit()
        flash("Theme deleted successfully!", "success")
    except Exception as e:
        connection.rollback()
        flash(f"Error deleting theme: {e}", "danger")
    connection.close()

```

```

        return redirect(url_for('main'))

@app.route('/insert_lego', methods=['POST'])
def insert_lego():
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    lego_id = request.form['lego_id']
    lego_theme = request.form['lego_theme']
    lego_name = request.form['lego_name']

    connection = get_db_connection()
    cursor = connection.cursor()
    try:
        cursor.execute(
            sql.SQL("INSERT INTO legoset (id, theme, name) VALUES (%s, %s, %s)"),
            [lego_id, lego_theme, lego_name]
        )
        connection.commit()
        flash("Lego set added successfully!", "success")
    except Exception as e:
        connection.rollback()
        flash(f"Error adding Lego set: {e}", "danger")
    finally:
        connection.close()

    return redirect(url_for('main'))

@app.route('/lego/edit', methods=['POST'])
def edit_lego():
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    lego_id = request.form['lego_id']
    lego_theme = request.form['lego_theme']
    lego_name = request.form['lego_name']

    connection = get_db_connection()
    cursor = connection.cursor()

```

```

    try:
        cursor.execute(
            "UPDATE legoset SET theme = %s, name = %s WHERE id = %s",
            (lego_theme, lego_name, lego_id)
        )
        connection.commit()
        cursor.close()
        flash("Lego set updated successfully!", "success")
    except Exception as e:
        connection.rollback()
        flash(f"Error editing Lego set: {e}", "danger")
    connection.close()
    return redirect(url_for('main'))

@app.route('/lego/delete/<int:id>')
def delete_lego(id):
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    connection = get_db_connection()
    cursor = connection.cursor()
    try:
        cursor.execute("DELETE FROM legoset WHERE id = %s", (id,))
        connection.commit()
        flash("Lego set deleted successfully!", "success")
    except Exception as e:
        connection.rollback()
        flash(f"Error deleting Lego set: {e}", "danger")
    connection.close()
    return redirect(url_for('main'))

@app.route('/find_lego', methods=['GET'])
def find_lego():
    if 'logged_in' not in session:
        return redirect(url_for('login'))

    query = request.args.get('query', '').strip()

    connection = get_db_connection()

```

```

cursor = connection.cursor()

try:
    if query.isdigit(): #Nếu truy vấn là số thì tìm theo id
        cursor.execute("""
            SELECT id, theme, name
            FROM legoset
            WHERE id = %s
            ORDER BY id ASC
            """, (query,))
    else: #Không thì tìm theo chủ đề hoặc tên
        cursor.execute("""
            SELECT id, theme, name
            FROM legoset
            WHERE LOWER(theme) LIKE LOWER(%s) OR LOWER(name)
            LIKE LOWER(%s)
            ORDER BY id ASC
            """, (f"%{query}%", f"%{query}%"))

    lego_sets = cursor.fetchall()

    cursor.execute("SELECT row_number() OVER () as id, theme
FROM themes ORDER BY theme ASC")
    themes = cursor.fetchall()
except Exception as e:
    flash(f"Error searching Lego sets: {e}", "danger")
    lego_sets = []
    themes = []
finally:
    connection.close()

return render_template(
    'main.html',
    lego_sets=lego_sets,
    themes=themes,
    title='Lego Store Management'
)

#Logout
@app.route('/logout')

```



```
def logout():
    session.pop('logged_in', None)
    session.clear()
    flash("You have been logged out.", "info")
    return redirect(url_for('login'))

if __name__ == '__main__':
    app.run(debug=True)
```

IV. GITHUB

Nguồn: <https://github.com/YueTruong/pythonnc>

HẾT