

**TRƯỜNG ĐẠI HỌC HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN**



VAN LANG
UNIVERSITY



**BÁO CÁO ĐỒ ÁN MÔN HỌC HK241
LẬP TRÌNH PYTHON NÂNG CAO**

Nội dung thực hiện:

- 1. XÂY DỰNG ỨNG DỤNG TÍNH TOÁN ĐƠN GIẢN**
- 2. XÂY DỰNG APPLICATION QUẢN LÝ SINH VIÊN**
- 3. XÂY DỰNG WEBSITE QUẢN LÝ SINH VIÊN**

SVTH: Lý Trần Quốc Bảo – 197CT09571

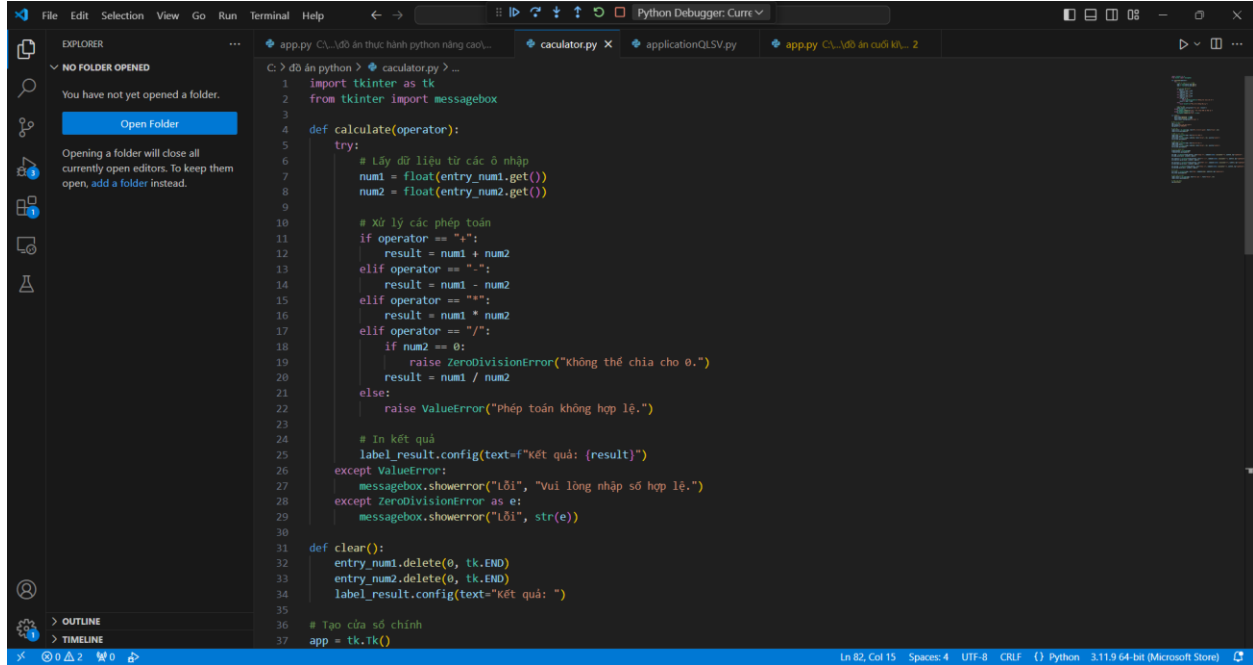
LỚP: 241_71ITSE31003_01

GVHD: Huỳnh Thái Học

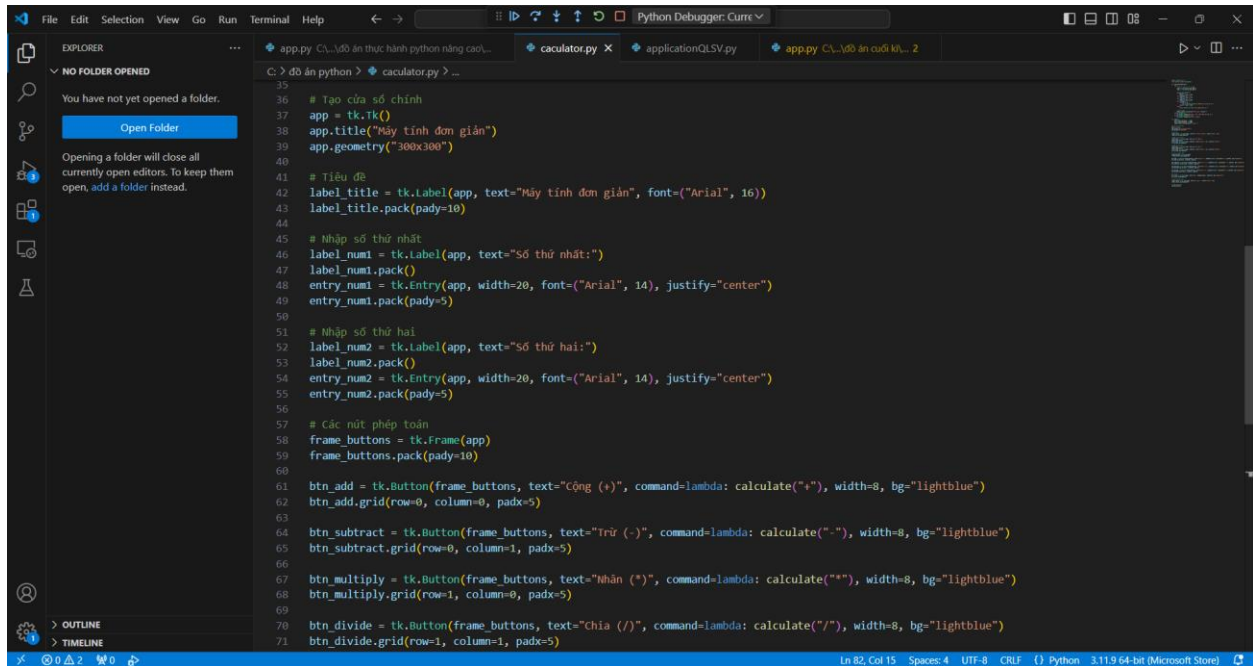
TP. Hồ Chí Minh – năm 2024

1. XÂY DỰNG ỨNG DỤNG TÍNH TOÁN ĐƠN GIẢN

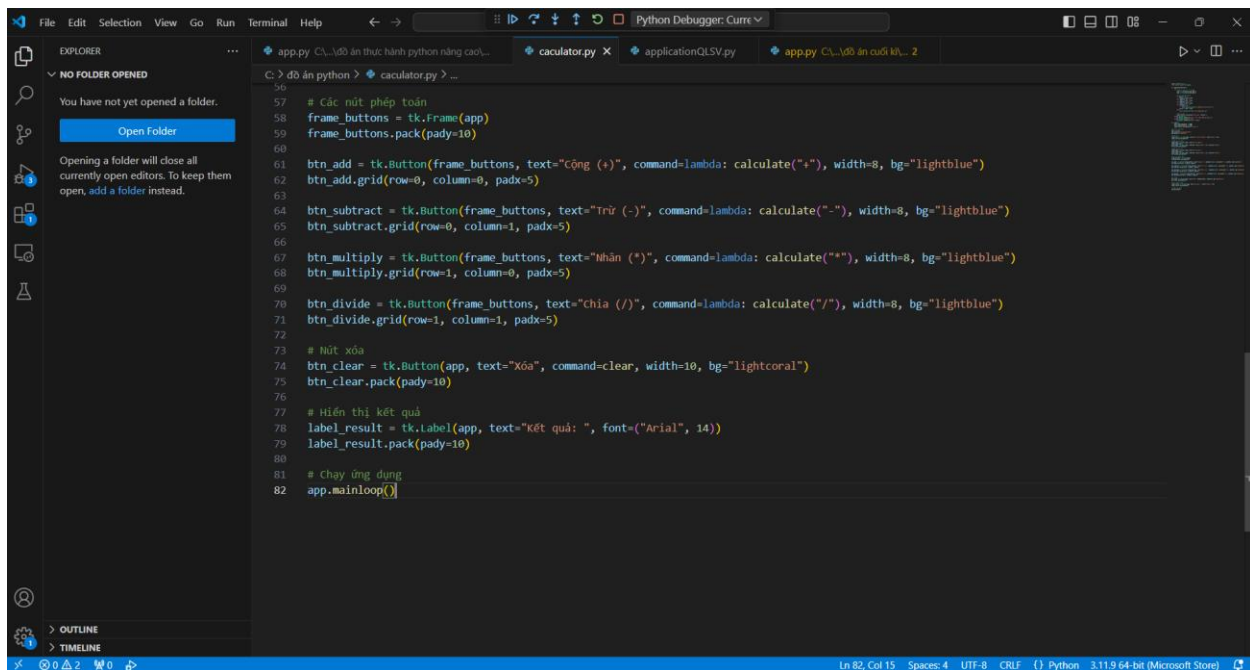
-Phần code:



```
1 import tkinter as tk
2 from tkinter import messagebox
3
4 def calculate(operator):
5     try:
6         # Lấy dữ liệu từ các ô nhập
7         num1 = float(entry_num1.get())
8         num2 = float(entry_num2.get())
9
10        # Xử lý các phép toán
11        if operator == "+":
12            result = num1 + num2
13        elif operator == "-":
14            result = num1 - num2
15        elif operator == "*":
16            result = num1 * num2
17        elif operator == "/":
18            if num2 == 0:
19                raise ZeroDivisionError("Không thể chia cho 0.")
20            result = num1 / num2
21        else:
22            raise ValueError("Phép toán không hợp lệ.")
23
24        # In kết quả
25        label_result.config(text=f"Kết quả: {result}")
26    except ValueError:
27        messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ.")
28    except ZeroDivisionError as e:
29        messagebox.showerror("Lỗi", str(e))
30
31 def clear():
32     entry_num1.delete(0, tk.END)
33     entry_num2.delete(0, tk.END)
34     label_result.config(text="Kết quả: ")
35
36 # Tạo cửa sổ chính
37 app = tk.Tk()
```

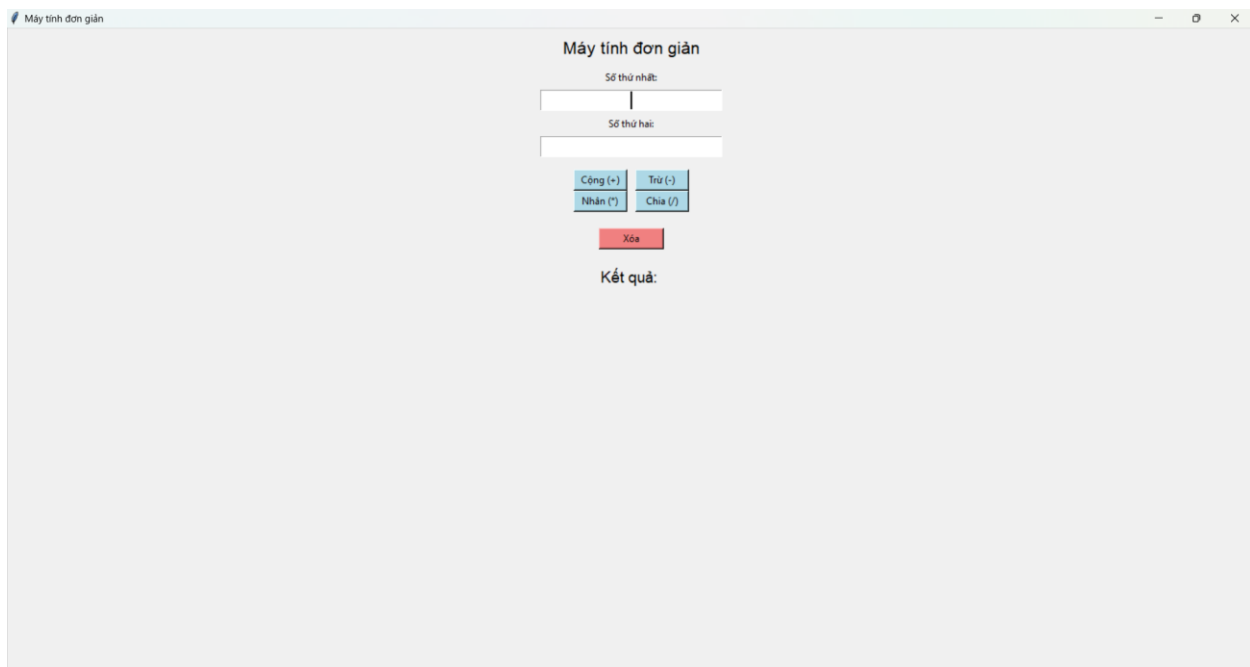


```
35
36 # Tạo cửa sổ chính
37 app = tk.Tk()
38 app.title("Máy tính đơn giản")
39 app.geometry("300x300")
40
41 # Tiêu đề
42 label_title = tk.Label(app, text="Máy tính đơn giản", font=("Arial", 16))
43 label_title.pack(pady=10)
44
45 # Nhập số thứ nhất
46 label_num1 = tk.Label(app, text="Số thứ nhất:")
47 label_num1.pack()
48 entry_num1 = tk.Entry(app, width=20, font=("Arial", 14), justify="center")
49 entry_num1.pack(pady=5)
50
51 # Nhập số thứ hai
52 label_num2 = tk.Label(app, text="Số thứ hai:")
53 label_num2.pack()
54 entry_num2 = tk.Entry(app, width=20, font=("Arial", 14), justify="center")
55 entry_num2.pack(pady=5)
56
57 # Các nút phép toán
58 frame_buttons = tk.Frame(app)
59 frame_buttons.pack(pady=10)
60
61 btn_add = tk.Button(frame_buttons, text="Cộng (+)", command=lambda: calculate("+"), width=8, bg="lightblue")
62 btn_add.grid(row=0, column=0, padx=5)
63
64 btn_subtract = tk.Button(frame_buttons, text="Trừ (-)", command=lambda: calculate("-"), width=8, bg="lightblue")
65 btn_subtract.grid(row=0, column=1, padx=5)
66
67 btn_multiply = tk.Button(frame_buttons, text="Nhân (*)", command=lambda: calculate("*"), width=8, bg="lightblue")
68 btn_multiply.grid(row=1, column=0, padx=5)
69
70 btn_divide = tk.Button(frame_buttons, text="Chia (/)", command=lambda: calculate("/"), width=8, bg="lightblue")
71 btn_divide.grid(row=1, column=1, padx=5)
```

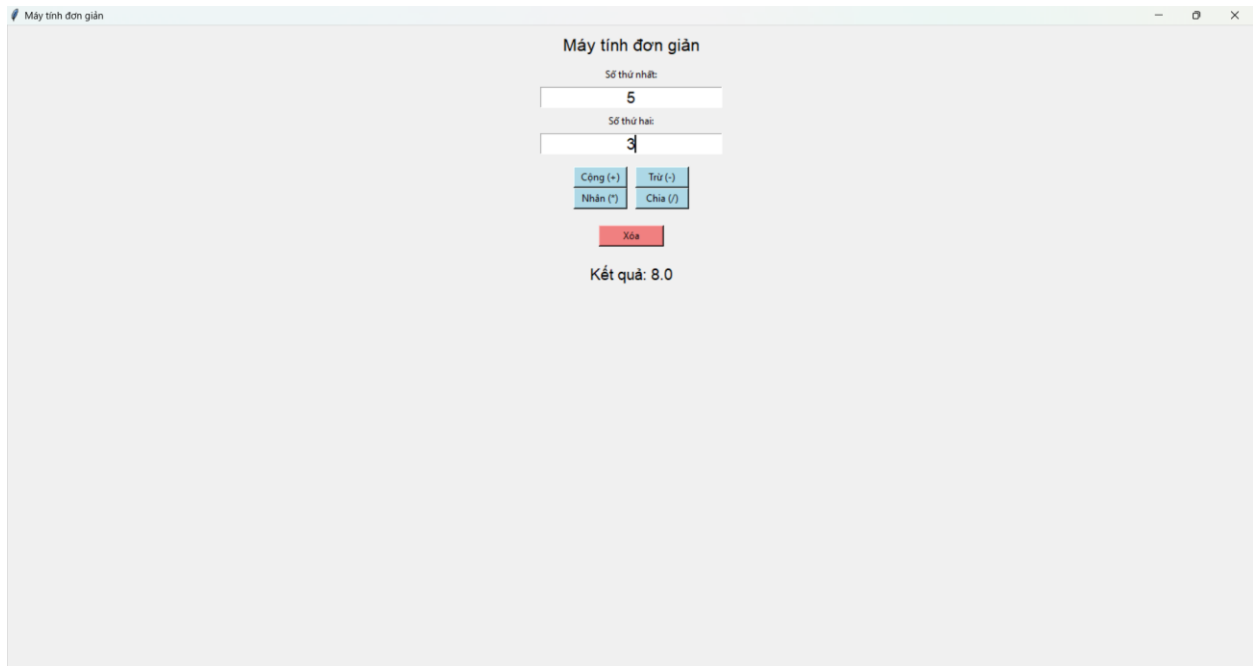


```
57 # Các nút phép toán
58 frame_buttons = tk.Frame(app)
59 frame_buttons.pack(pady=10)
60
61 btn_add = tk.Button(frame_buttons, text="Cộng (+)", command=lambda: calculate("+"), width=8, bg="lightblue")
62 btn_add.grid(row=0, column=0, padx=5)
63
64 btn_subtract = tk.Button(frame_buttons, text="Trừ (-)", command=lambda: calculate("-"), width=8, bg="lightblue")
65 btn_subtract.grid(row=0, column=1, padx=5)
66
67 btn_multiply = tk.Button(frame_buttons, text="Nhân (*)", command=lambda: calculate("*"), width=8, bg="lightblue")
68 btn_multiply.grid(row=1, column=0, padx=5)
69
70 btn_divide = tk.Button(frame_buttons, text="Chia (/)", command=lambda: calculate("/"), width=8, bg="lightblue")
71 btn_divide.grid(row=1, column=1, padx=5)
72
73 # Nút xóa
74 btn_clear = tk.Button(app, text="Xóa", command=clear, width=10, bg="lightcoral")
75 btn_clear.pack(pady=10)
76
77 # Hiển thị kết quả
78 label_result = tk.Label(app, text="Kết quả: ", font=("Arial", 14))
79 label_result.pack(pady=10)
80
81 # Chạy ứng dụng
82 app.mainloop()
```

-Kết quả của app tính toán:



_Kết quả sau khi thực hiện một phép tính:



-Chức năng xây dựng app tính toán đơn giản bằng Python thường tập trung vào việc tạo một ứng dụng có giao diện cơ bản để thực hiện các phép tính. Các bước chính gồm:

1. Tính năng chính

- **Phép toán cơ bản:** Cộng, trừ, nhân, chia.
- **Tính năng nâng cao (tùy chọn):** Số mũ, căn bậc hai, phần trăm.

2. Giao diện

- **Sử dụng thư viện:**
 - **Terminal/Console:** Input/output trực tiếp qua dòng lệnh.
 - **Giao diện đồ họa (GUI):** Thư viện như tkinter, PyQt, hoặc Kivy để thiết kế giao diện trực quan với các nút bấm, ô nhập liệu và màn hình hiển thị kết quả.

3. Luồng hoạt động cơ bản

- Người dùng nhập số và chọn phép toán.
- Ứng dụng thực hiện tính toán dựa trên logic được lập trình sẵn.
- Hiển thị kết quả trên màn hình (console hoặc GUI).

4. Lợi ích

- Giúp người học Python thực hành các khái niệm cơ bản như: hàm, điều kiện, vòng lặp và xử lý lỗi.
- Cung cấp nền tảng để phát triển ứng dụng phức tạp hơn sau này.

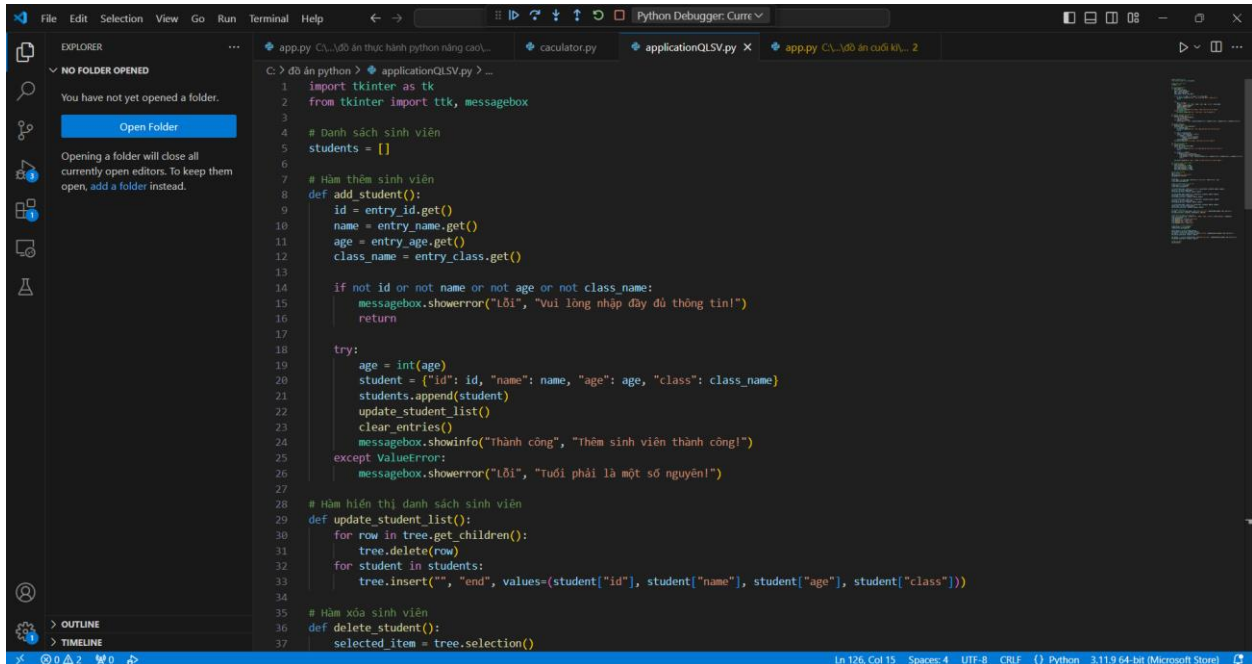
Python là ngôn ngữ phù hợp cho ứng dụng kiểu này nhờ tính đơn giản và thư viện phong phú.

-Link github Bài 1: App Tính Toán Đơn Giản

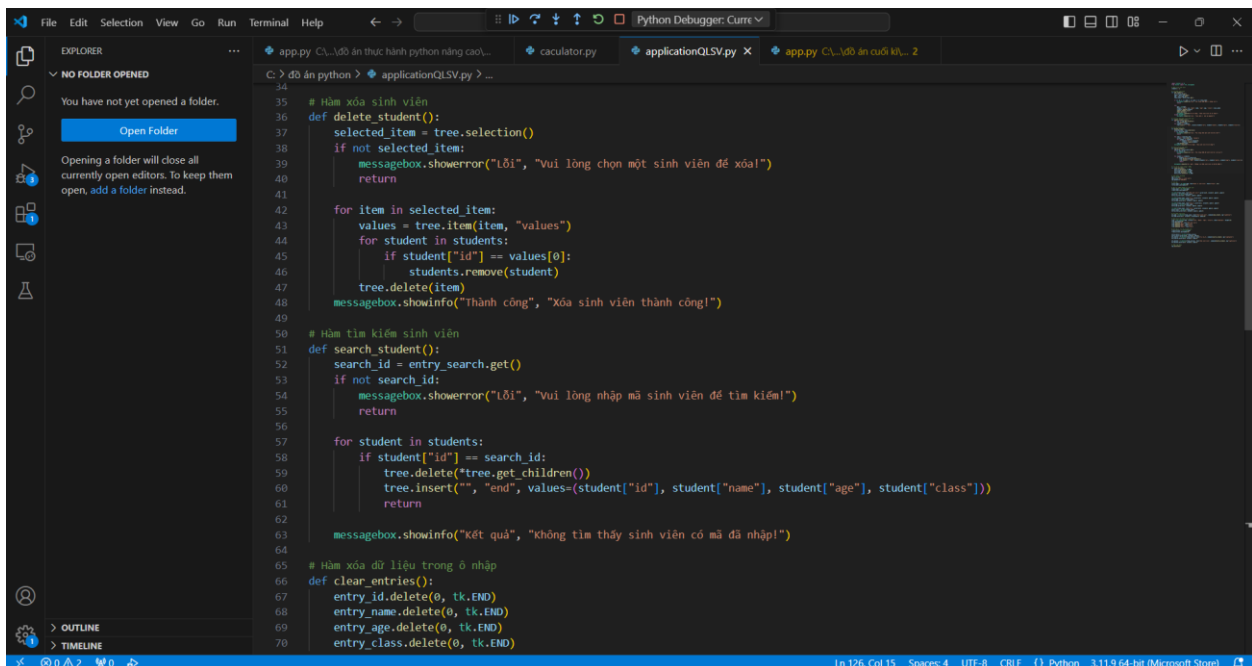
<https://github.com/QuocBao0103/DOANCUOIKIPYTHONNC/blob/master/caculator.py>

2. XÂY DỰNG APPLICATION QUẢN LÝ SINH VIÊN

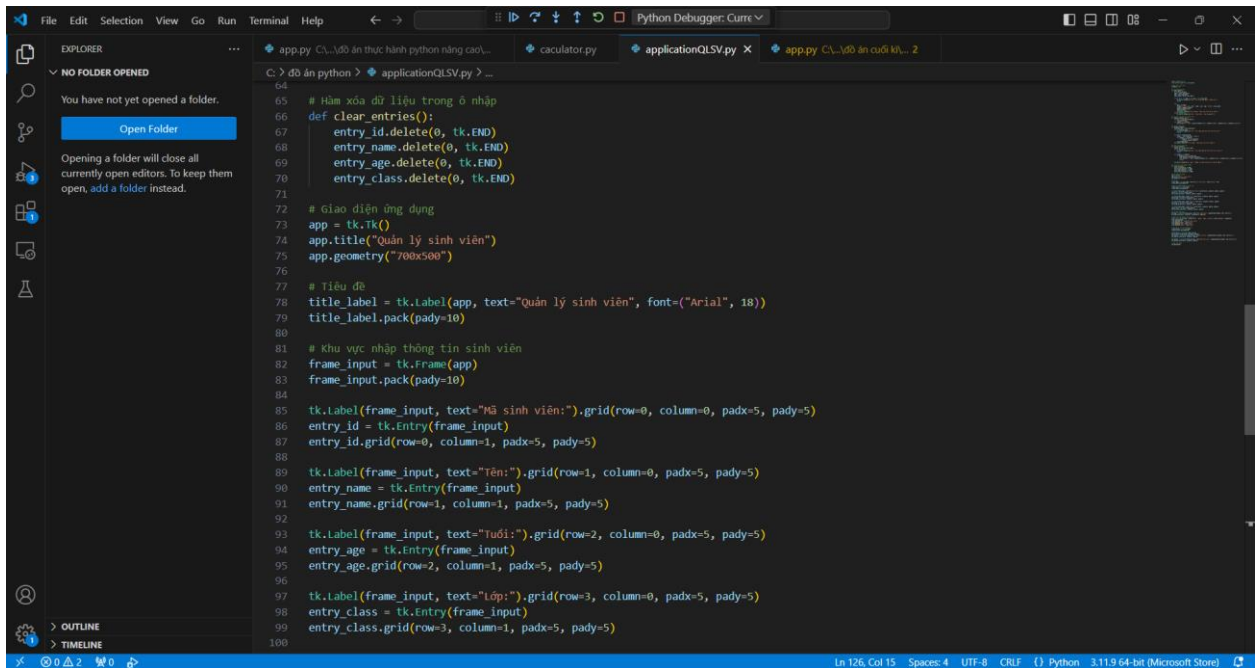
-Phần code:



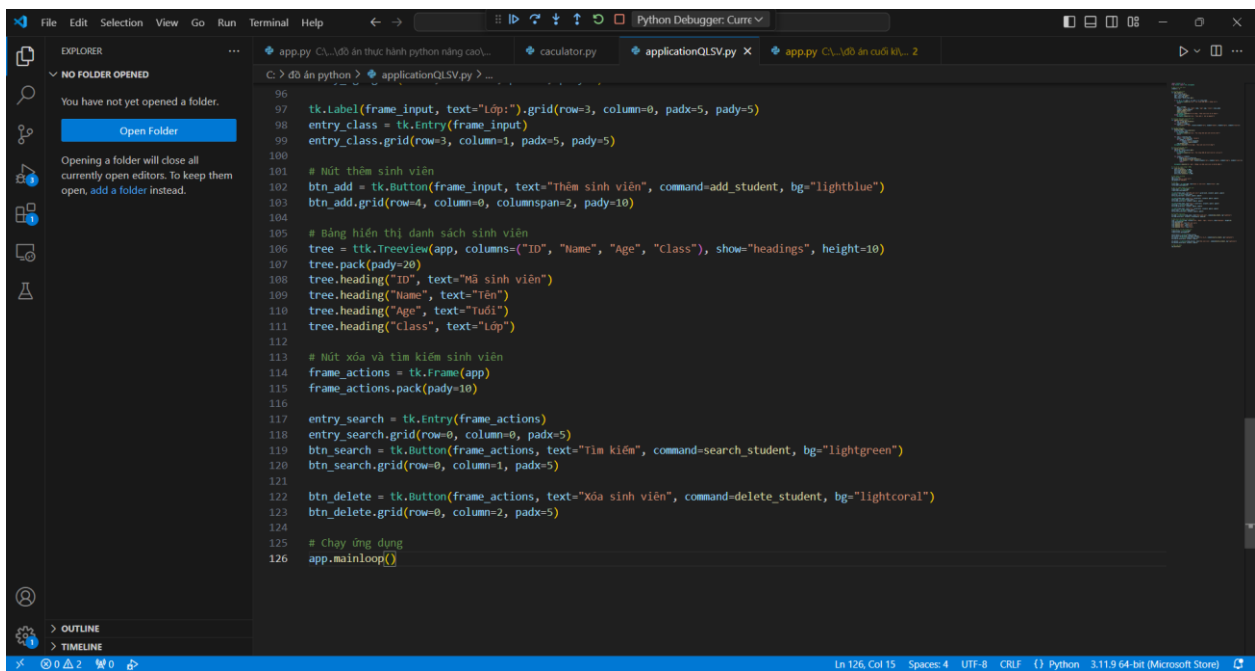
```
1 import tkinter as tk
2 from tkinter import ttk, messagebox
3
4 # Danh sách sinh viên
5 students = []
6
7 # Hàm thêm sinh viên
8 def add_student():
9     id = entry_id.get()
10    name = entry_name.get()
11    age = entry_age.get()
12    class_name = entry_class.get()
13
14    if not id or not name or not age or not class_name:
15        messagebox.showerror("Lỗi", "Vui lòng nhập đầy đủ thông tin!")
16        return
17
18    try:
19        age = int(age)
20        student = {"id": id, "name": name, "age": age, "class": class_name}
21        students.append(student)
22        update_student_list()
23        clear_entries()
24        messagebox.showinfo("Thành công", "Thêm sinh viên thành công!")
25    except ValueError:
26        messagebox.showerror("Lỗi", "Tuổi phải là một số nguyên!")
27
28 # Hàm hiển thị danh sách sinh viên
29 def update_student_list():
30     for row in tree.get_children():
31         tree.delete(row)
32     for student in students:
33         tree.insert("", "end", values=(student["id"], student["name"], student["age"], student["class"]))
34
35 # Hàm xóa sinh viên
36 def delete_student():
37     selected_item = tree.selection()
```



```
34
35 # Hàm xóa sinh viên
36 def delete_student():
37     selected_item = tree.selection()
38     if not selected_item:
39         messagebox.showerror("Lỗi", "Vui lòng chọn một sinh viên để xóa!")
40         return
41
42     for item in selected_item:
43         values = tree.item(item, "values")
44         for student in students:
45             if student["id"] == values[0]:
46                 students.remove(student)
47                 tree.delete(item)
48     messagebox.showinfo("Thành công", "Xóa sinh viên thành công!")
49
50 # Hàm tìm kiếm sinh viên
51 def search_student():
52     search_id = entry_search.get()
53     if not search_id:
54         messagebox.showerror("Lỗi", "Vui lòng nhập mã sinh viên để tìm kiếm!")
55         return
56
57     for student in students:
58         if student["id"] == search_id:
59             tree.delete(*tree.get_children())
60             tree.insert("", "end", values=(student["id"], student["name"], student["age"], student["class"]))
61             return
62
63     messagebox.showinfo("Kết quả", "Không tìm thấy sinh viên có mã đã nhập!")
64
65 # Hàm xóa dữ liệu trong ô nhập
66 def clear_entries():
67     entry_id.delete(0, tk.END)
68     entry_name.delete(0, tk.END)
69     entry_age.delete(0, tk.END)
70     entry_class.delete(0, tk.END)
```



```
64
65 # Hàm xóa dữ liệu trong ô nhập
66 def clear_entries():
67     entry_id.delete(0, tk.END)
68     entry_name.delete(0, tk.END)
69     entry_age.delete(0, tk.END)
70     entry_class.delete(0, tk.END)
71
72 # Giao diện ứng dụng
73 app = tk.Tk()
74 app.title("Quản lý sinh viên")
75 app.geometry("700x500")
76
77 # Tiêu đề
78 title_label = tk.Label(app, text="Quản lý sinh viên", font=("Arial", 18))
79 title_label.pack(pady=10)
80
81 # Khu vực nhập thông tin sinh viên
82 frame_input = tk.Frame(app)
83 frame_input.pack(pady=10)
84
85 tk.Label(frame_input, text="Mã sinh viên:").grid(row=0, column=0, padx=5, pady=5)
86 entry_id = tk.Entry(frame_input)
87 entry_id.grid(row=0, column=1, padx=5, pady=5)
88
89 tk.Label(frame_input, text="Tên:").grid(row=1, column=0, padx=5, pady=5)
90 entry_name = tk.Entry(frame_input)
91 entry_name.grid(row=1, column=1, padx=5, pady=5)
92
93 tk.Label(frame_input, text="Tuổi:").grid(row=2, column=0, padx=5, pady=5)
94 entry_age = tk.Entry(frame_input)
95 entry_age.grid(row=2, column=1, padx=5, pady=5)
96
97 tk.Label(frame_input, text="Lớp:").grid(row=3, column=0, padx=5, pady=5)
98 entry_class = tk.Entry(frame_input)
99 entry_class.grid(row=3, column=1, padx=5, pady=5)
100
```



```
96
97 tk.Label(frame_input, text="Lớp:").grid(row=3, column=0, padx=5, pady=5)
98 entry_class = tk.Entry(frame_input)
99 entry_class.grid(row=3, column=1, padx=5, pady=5)
100
101 # Nút thêm sinh viên
102 btn_add = tk.Button(frame_input, text="Thêm sinh viên", command=add_student, bg="lightblue")
103 btn_add.grid(row=4, column=0, colspan=2, padx=5, pady=10)
104
105 # Bảng hiển thị danh sách sinh viên
106 tree = ttk.Treeview(app, columns=("ID", "Name", "Age", "Class"), show="headings", height=10)
107 tree.pack(pady=20)
108 tree.heading("ID", text="Mã sinh viên")
109 tree.heading("Name", text="Tên")
110 tree.heading("Age", text="Tuổi")
111 tree.heading("Class", text="Lớp")
112
113 # Nút xóa và tìm kiếm sinh viên
114 frame_actions = tk.Frame(app)
115 frame_actions.pack(pady=10)
116
117 entry_search = tk.Entry(frame_actions)
118 entry_search.grid(row=0, column=0, padx=5)
119 btn_search = tk.Button(frame_actions, text="Tìm kiếm", command=search_student, bg="lightgreen")
120 btn_search.grid(row=0, column=1, padx=5)
121
122 btn_delete = tk.Button(frame_actions, text="Xóa sinh viên", command=delete_student, bg="lightcoral")
123 btn_delete.grid(row=0, column=2, padx=5)
124
125 # Chạy ứng dụng
126 app.mainloop()
```

-Kết quả của application quản lý sinh viên:

Quản lý sinh viên

Quản lý sinh viên

Mã sinh viên:

Tên:

Tuổi:

Lớp:

Mã sinh viên	Tên	Tuổi	Lớp
--------------	-----	------	-----

-Kết quả sau khi thêm sinh viên:

Quản lý sinh viên

Quản lý sinh viên

Mã sinh viên:

Tên:

Tuổi:

Lớp:

Mã sinh viên	Tên	Tuổi	Lớp
197CT09571	Lý Trần Quốc Bảo	23	K25-IT05

Thành công

Thêm sinh viên thành công!

-Kết quả tìm kiếm sinh viên:

Quản lý sinh viên

Mã sinh viên:

Tên:

Tuổi:

Lớp:

Mã sinh viên	Tên	Tuổi	Lớp
197CT09571	Lý Trần Quốc Bảo	23	K25-IT05

Ứng dụng quản lý sinh viên bằng Python là một chương trình cho phép quản lý thông tin sinh viên một cách hiệu quả. Các chức năng cơ bản thường bao gồm:

1. Tính năng chính

- **Thêm sinh viên:** Nhập thông tin (mã số, tên, lớp, tuổi, điểm, v.v.).
- **Hiển thị danh sách sinh viên:** Liệt kê tất cả sinh viên cùng thông tin chi tiết.
- **Tìm kiếm sinh viên:** Tìm theo mã số, tên, hoặc các tiêu chí khác.
- **Chỉnh sửa thông tin:** Cập nhật thông tin sinh viên đã lưu.
- **Xóa sinh viên:** Xóa thông tin của sinh viên khỏi danh sách.
- **Tính toán và xếp hạng:** Tính điểm trung bình, xếp loại học lực.

2. Lưu trữ dữ liệu

- **Dữ liệu tạm thời:** Sử dụng danh sách hoặc từ điển Python trong bộ nhớ.
- **Lưu trữ lâu dài:** Lưu dữ liệu vào file (.txt, .csv, .json) hoặc cơ sở dữ liệu (SQLite, MySQL).

3. Giao diện

- **Dòng lệnh (CLI):** Dễ phát triển, phù hợp với người mới học.

- **Giao diện đồ họa (GUI):** Sử dụng thư viện như tkinter, PyQt, hoặc Kivy để tạo trải nghiệm người dùng thân thiện hơn.

4. Luồng hoạt động cơ bản

- Người dùng nhập lựa chọn (thêm, sửa, xóa, tìm kiếm).
- Chương trình xử lý yêu cầu và hiển thị kết quả.
- Dữ liệu được lưu trữ để sử dụng sau (nếu cần).

5. Lợi ích

- Củng cố kiến thức Python (hàm, cấu trúc dữ liệu, xử lý file, GUI).
- Ứng dụng thực tế, dễ mở rộng để phù hợp với các dự án lớn hơn.

Ứng dụng này có thể phát triển từ đơn giản (CLI) đến phức tạp (kết nối cơ sở dữ liệu, giao diện chuyên nghiệp).

-Link GitHub Bài 2: Application Quản Lý Sinh Viên

<https://github.com/QuocBao0103/DOANCUOIKIPYTHONNC/blob/master/applicationQLSV.py>

3.XÂY DỰNG WEBSITE QUẢN LÝ SINH VIÊN

-Phần code:

The screenshot shows a VS Code editor with a Python project. The Explorer sidebar on the left shows a file structure with a `templates` folder, a `static` folder, and several HTML files. The main editor displays the `app.py` file, which is a Flask application using SQLAlchemy for database integration. The code includes imports for `Flask`, `Flask-Login`, `Flask-Redirect`, `Flask-Session`, and `Flask-SQLAlchemy`. It defines a `User` model, initializes the application with a secret key, session timeout, and database URI, and sets up routes for `home` and `login`.

```
C:\> cd an thuc hinh python nang cao > flask-login > app.py > ...
1 from flask import Flask, redirect, url_for, render_template, request, session, flash
2 from datetime import timedelta
3 from flask_sqlalchemy import SQLAlchemy
4 from os import path
5
6 app = Flask(__name__)
7 app.config["SECRET_KEY"] = "Bao"
8 app.permanent_session_lifetime = timedelta(minutes=1)
9 app.config["SQLALCHEMY_DATABASE_URI"] = 'sqlite:///user.db'
10 app.config["SQLALCHEMY_TRACK_MODIFICATIONS"] = False
11 db = SQLAlchemy(app)
12
13 class User(db.Model):
14     user_id = db.Column(db.Integer, primary_key=True)
15     name = db.Column(db.String(100))
16     email = db.Column(db.String(100))
17
18     def __init__(self, name, email):
19         self.name = name
20         self.email = email
21
22 @app.route("/home")
23 @app.route("/")
24 def home():
25     return render_template("home.html")
26
27 @app.route("/login", methods=["POST", "GET"])
28 def login():
29     if request.method == "POST":
30         user_name = request.form["name"]
31         session.permanent = True
32         if user_name:
33             session["user"] = user_name
34             found_user = User.query.filter_by(name=user_name).first()
35             if found_user:
36                 session["email"] = found_user.email
37             else:
```

The screenshot shows a Python IDE (PyCharm) with a Flask application. The Explorer panel on the left shows a project structure with templates, static files, and views. The main editor displays the app.py file, which contains the Flask app logic for user registration, login, and profile updates. The code uses SQLAlchemy for database operations and Jinja2 for templating. The status bar at the bottom indicates the current file is app.py, line 7, column 33, and the Python version is 3.13.0.

```

File Edit Selection View Go Run Terminal Help
Python Debugger: Curre...

EXPLORER
TEMPLATES
  .venv
    > Include
    > Lib
    > Scripts
    > gitignore
    > pyenvm.ctg
  base.html
  home.html
  index.html
  login.html
  user.html

OUTLINE
TIMELINE

C:\> f) db ấn thực hành python nâng cao > Flask-login > app.py ~
28 def login():
34     found_user = User.query.filter_by(name=user_name).first()
35     if found_user:
36         session["email"] = found_user.email
37     else:
38         user = User(user_name, "temp@gmail.com")
39         db.session.add(user)
40         db.session.commit()
41         flash("New user created!")
42         return redirect(url_for("user", user=user_name))
43     if "user" in session:
44         name = session["user"]
45         return redirect(url_for("user", user=name))
46     return render_template("login.html")
47
48 @app.route("/user", methods=["POST", "GET"])
49 def user():
50     email = None
51     if "user" in session:
52         name = session["user"]
53         if request.method == "POST":
54             if not request.form["email"] and request.form["name"]:
55                 User.query.filter_by(name=name).delete()
56                 db.session.commit()
57                 flash("User deleted!")
58                 return redirect(url_for("log_out"))
59             else:
60                 email = request.form["email"]
61                 session["email"] = email
62                 found_user = User.query.filter_by(name=name).first()
63                 found_user.email = email
64                 db.session.commit()
65                 flash("Email updated!")
66             elif "email" in session:
67                 email = session["email"]
68                 return render_template("user.html", user=name, email=email)
69         else:

```

```
File Edit Selection View Go Run Terminal Help
Python Debugger: Curre...

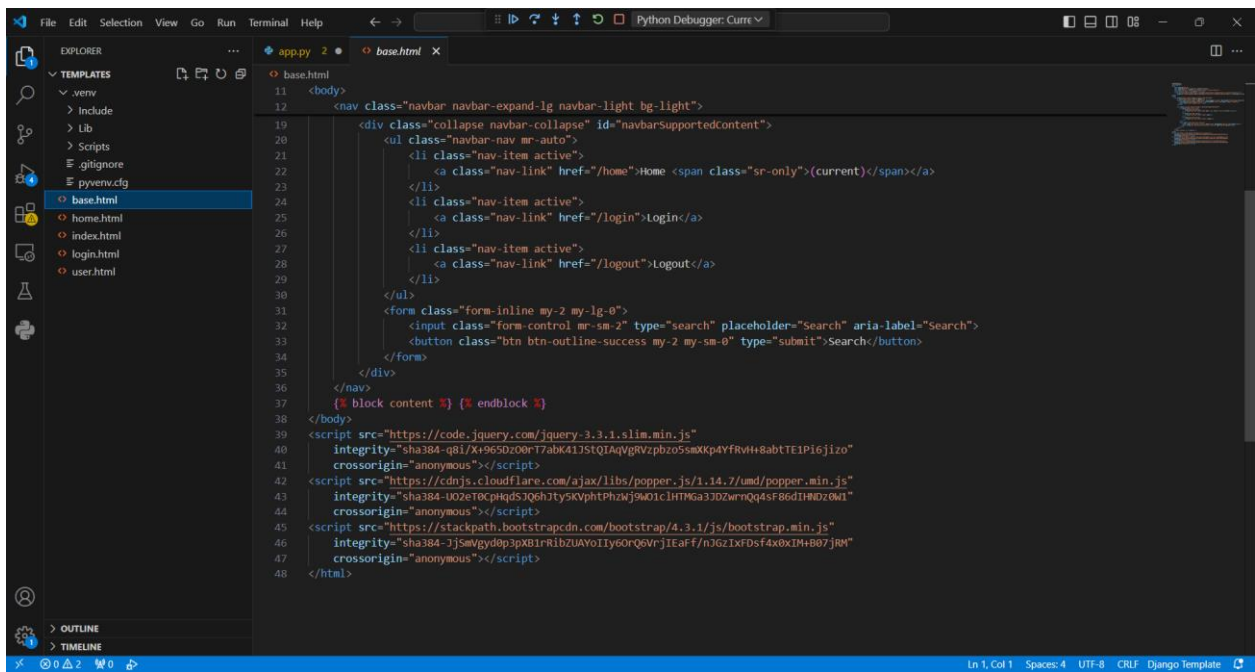
EXPLORER
TEMPLATES
  .venv
    > Include
    > Lib
    > Scripts
    E .gitignore
    E pyvenv.cfg
    base.html
    home.html
    index.html
    login.html
    user.html

base.html
C:\> f > đã ấn thực hành python nâng cao > Flask-login > app.py > ...
49 def user():
50     flash(user_delete)
51     return redirect(url_for("log_out"))
52 else:
53     email = request.form["email"]
54     session["email"] = email
55     found user = User.query.filter_by(name=name).first()
56     found user.email = email
57     db.session.commit()
58     flash("Email updated!")
59 elif "email" in session:
60     email = session["email"]
61     return render_template("user.html", user=name, email=email)
62 else:
63     return redirect(url_for("login"))
64
65 @app.route("/logout")
66 def log_out():
67     session.pop("user", None)
68     session.pop("email", None)
69     return redirect(url_for("login"))
70
71 if __name__ == "__main__":
72     if not path.exists("user.db"):
73         with app.app_context():
74             db.create_all()
75             print("Created database!")
76     app.run(debug=True)
```

```
File Edit Selection View Go Run Terminal Help
Python Debugger: Curre...

EXPLORER
TEMPLATES
  .venv
    > Include
    > Lib
    > Scripts
    E .gitignore
    E pyvenv.cfg
    base.html
    home.html
    index.html
    login.html
    user.html

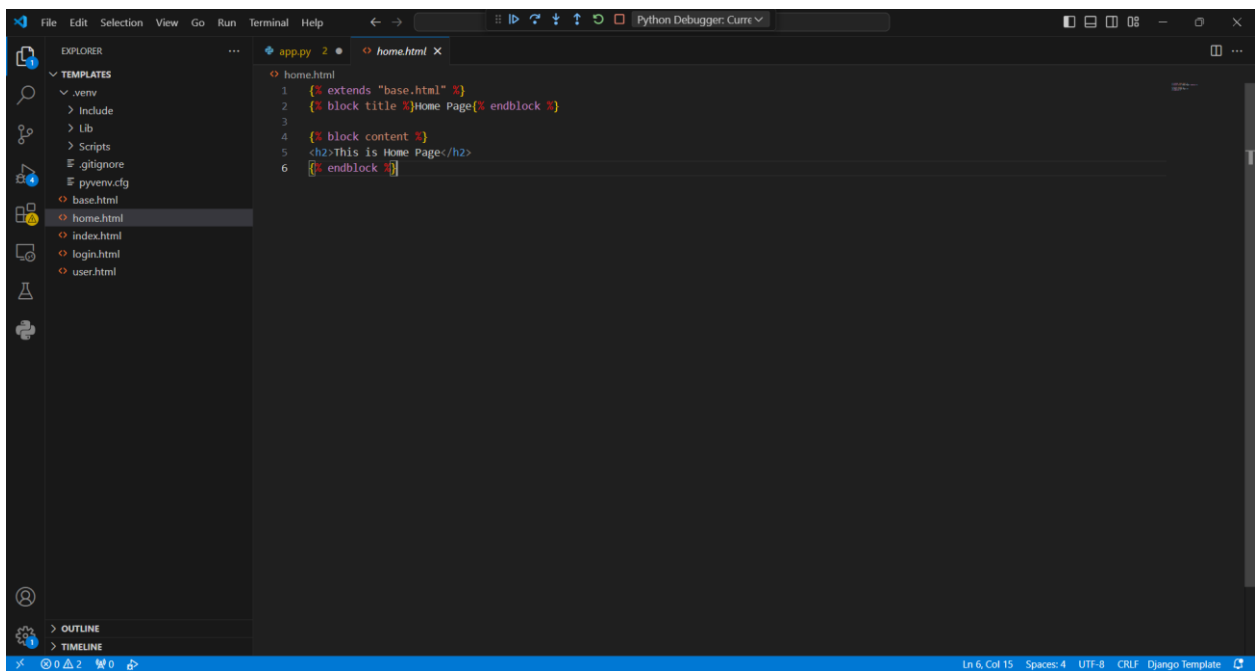
base.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>{{ block title }} {{ endblock }}</title>
8     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
9         integrity="sha384-ggOyR0iXCbMQV3EiRa03evJww6935fa224eR05qXzUW7W8c3fqv7WuIcm07vg" crossorigin="anonymous">
10 </head>
11 <body>
12     <nav class="navbar navbar-expand-lg navbar-light bg-light">
13         <a class="navbar-brand" href="#">Navbar</a>
14         <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent"
15             aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
16             <span class="navbar-toggler-icon"></span>
17         </button>
18
19         <div class="collapse navbar-collapse" id="navbarSupportedContent">
20             <ul class="navbar-nav mr-auto">
21                 <li class="nav-item active">
22                     <a class="nav-link" href="/home">Home <span class="sr-only">(current)</span></a>
23                 </li>
24                 <li class="nav-item active">
25                     <a class="nav-link" href="/login">Login</a>
26                 </li>
27                 <li class="nav-item active">
28                     <a class="nav-link" href="/logout">Logout</a>
29                 </li>
30             </ul>
31             <form class="form-inline my-2 my-lg-0">
32                 <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search">
33                 <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
34             </form>
35         </div>
36     </nav>
37     {{ block content }} {{ endblock }}
```



The screenshot shows the Visual Studio Code editor with the 'base.html' file open. The Explorer sidebar on the left shows a project structure with files like 'base.html', 'home.html', 'index.html', 'login.html', and 'user.html'. The main editor area displays the following HTML code:

```
11 <body>
12 <nav class="navbar navbar-expand-lg navbar-light bg-light">
19 <div class="collapse navbar-collapse" id="navbarSupportedContent">
20 <ul class="navbar-nav mr-auto">
21 <li class="nav-item active">
22 <a class="nav-link" href="/home">Home <span class="sr-only">(current)</span></a>
23 </li>
24 <li class="nav-item active">
25 <a class="nav-link" href="/login">Login</a>
26 </li>
27 <li class="nav-item active">
28 <a class="nav-link" href="/logout">Logout</a>
29 </li>
30 </ul>
31 <form class="form-inline my-2 my-lg-0">
32 <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search">
33 <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
34 </form>
35 </div>
36 </nav>
37 {% block content %} {% endblock %}
38 </body>
39 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
40 integrity="sha384-q8i/X+965DzO0rT7abK417StQIAqVgRVzpbzo58Xp4Yfvr4H+8abTTE1Piejizo"
41 crossorigin="anonymous"></script>
42 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
43 integrity="sha384-UO2eT0CpHQdsQq6h7tysKVphtPhZj9WO1cLHTMga33DZwrmQq45F86DTHNDz0W1"
44 crossorigin="anonymous"></script>
45 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
46 integrity="sha384-3JsS4GwB708P16586df727L74U1A6goeWbU6g8IjEaFf6n7GzIXfD5f4x0XIMHB07JRM"
47 crossorigin="anonymous"></script>
48 </html>
```

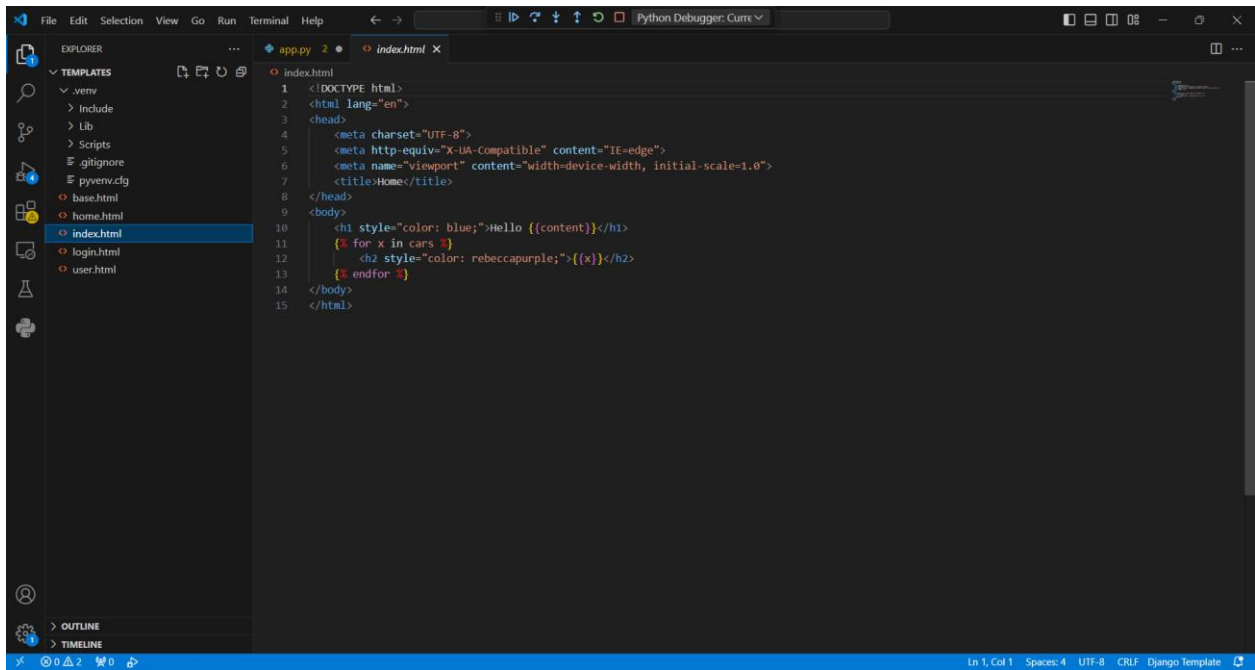
The status bar at the bottom indicates 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Django Template'.



The screenshot shows the Visual Studio Code editor with the 'home.html' file open. The Explorer sidebar on the left shows the same project structure. The main editor area displays the following HTML code:

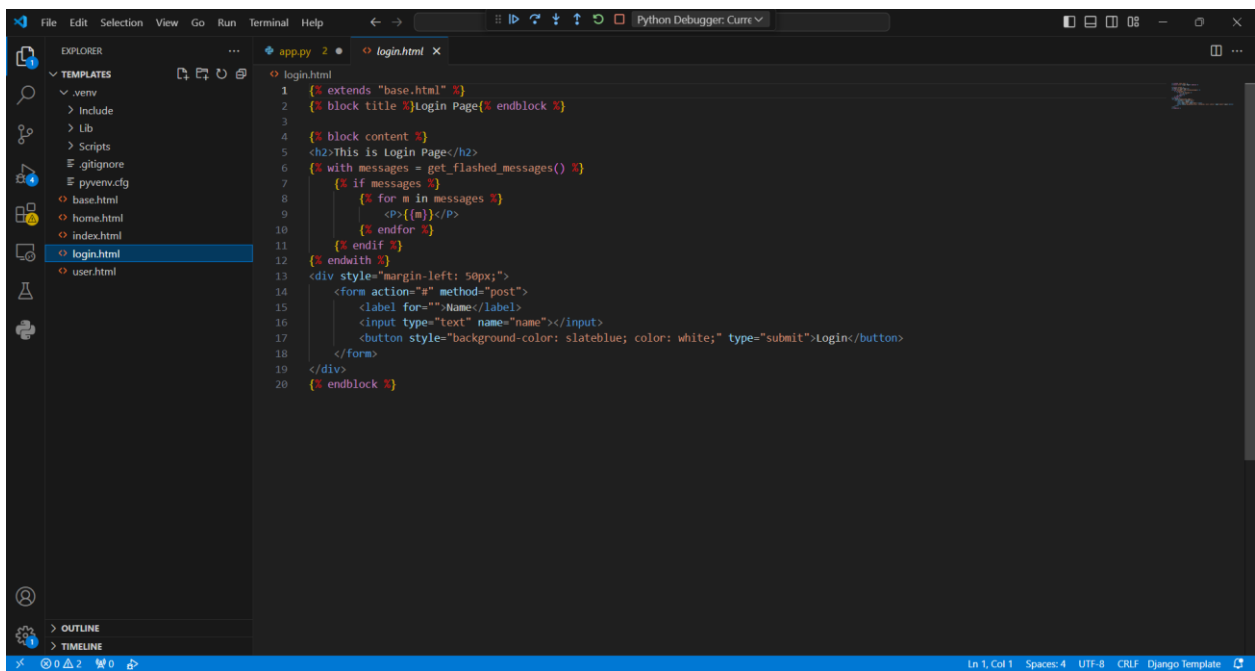
```
1 {% extends "base.html" %}
2 {% block title %}Home Page{% endblock %}
3
4 {% block content %}
5 <h2>this is Home Page</h2>
6 {% endblock %}
```

The status bar at the bottom indicates 'Ln 6, Col 15', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Django Template'.



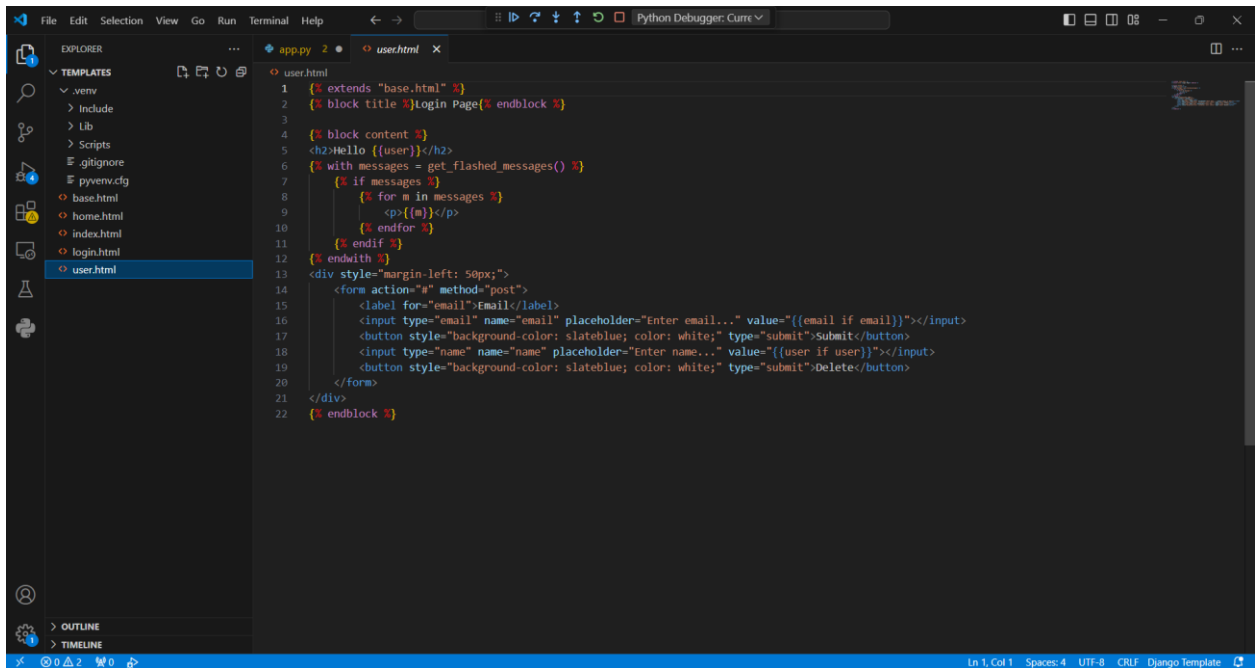
The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left. The Explorer sidebar shows a project structure with folders like .venv, Include, Lib, Scripts, .gitignore, and pyvenv.cfg. Below these are files: base.html, home.html, index.html (selected), login.html, and user.html. The main editor area displays the content of index.html, which is an HTML template. The code includes a DOCTYPE declaration, HTML lang="en", charset="UTF-8", and meta tags for http-equiv="X-UA-compatible" and name="viewport". The title is "Home". The body contains a heading h1 with the text "Hello {{content}}" and a loop for x in cars, which renders two h2 elements with different colors.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Home</title>
8 </head>
9 <body>
10  <h1 style="color: blue;">Hello {{content}}</h1>
11  {% for x in cars %}
12    <h2 style="color: rebeccapurple;">{{x}}</h2>
13  {% endfor %}
14 </body>
15 </html>
```



The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left. The Explorer sidebar shows a project structure with folders like .venv, Include, Lib, Scripts, .gitignore, and pyvenv.cfg. Below these are files: base.html, home.html, index.html, login.html (selected), and user.html. The main editor area displays the content of login.html, which is an HTML template. The code includes a block title "Login Page", a block content, and a form with a text input and a submit button. The form action is "#" and method is "post". The submit button has a background-color of slateblue and color of white.

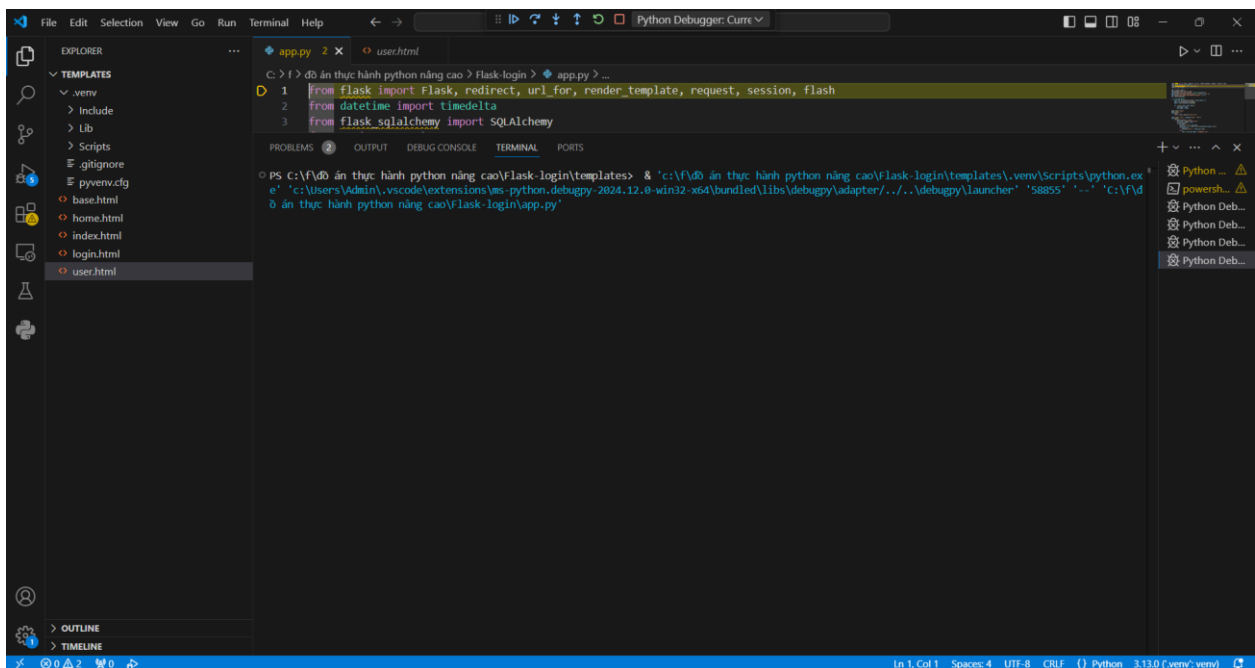
```
1 {% extends "base.html" %}
2 {% block title %}Login Page{% endblock %}
3
4 {% block content %}
5 <h2>this is Login Page</h2>
6 {% with messages = get_flashed_messages() %}
7   {% if messages %}
8     {% for m in messages %}
9       <p>{{m}}</p>
10    {% endfor %}
11  {% endif %}
12 {% endwith %}
13 <div style="margin-left: 50px;">
14   <form action="#" method="post">
15     <label for="">Name</label>
16     <input type="text" name="name"></input>
17     <button style="background-color: slateblue; color: white;" type="submit">Login</button>
18   </form>
19 </div>
20 {% endblock %}
```



The screenshot shows the Visual Studio Code editor with the 'user.html' file open. The file contains Jinja2 template code for a login page. It extends 'base.html' and includes a title 'Login Page'. The main content area contains a form with fields for email and name, and buttons for 'Submit' and 'Delete'. The form is styled with a margin-left of 50px. The code is as follows:

```
1 {% extends "base.html" %}
2 {% block title %}Login Page{% endblock %}
3
4 {% block content %}
5 <h2>Hello {{user}}</h2>
6 {% with messages = get_flashed_messages() %}
7   {% if messages %}
8     {% for m in messages %}
9       <p>{{m}}</p>
10    {% endfor %}
11  {% endif %}
12 {% endwith %}
13 <div style="margin-left: 50px;">
14   <form action="#" method="post">
15     <label for="email">Email</label>
16     <input type="email" name="email" placeholder="Enter email..." value="{{email if email}}"></input>
17     <button style="background-color: slateblue; color: white;" type="submit">Submit</button>
18     <input type="name" name="name" placeholder="Enter name..." value="{{user if user}}"></input>
19     <button style="background-color: slateblue; color: white;" type="submit">Delete</button>
20   </form>
21 </div>
22 {% endblock %}
```

-Kết quả:



The screenshot shows the Visual Studio Code editor with the 'app.py' file open. The file contains the main application logic, including imports for Flask, datetime, and SQLAlchemy. The terminal output shows the command to run the application and the resulting output.

```
1 from flask import Flask, redirect, url_for, render_template, request, session, flash
2 from datetime import timedelta
3 from flask_sqlalchemy import SQLAlchemy
```

The terminal output shows the command to run the application and the resulting output:

```
C:\> f> đđ ản thực hành python nâng cao > Flask login > app.py > ...
PS C:\> đđ ản thực hành python nâng cao\Flask-login\templates> & 'c:\> đđ ản thực hành python nâng cao\Flask-login\templates\venv\Scripts\python.exe' 'c:\Users\Admin\vscode\extensions\ms-python.debugpy-2024.12.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '58855' '-.' 'C:\> đđ ản thực hành python nâng cao\Flask-login\app.py'
```

Website quản lý sinh viên bằng Python cung cấp các chức năng cơ bản sau:

1. Quản lý sinh viên:

- **Thêm mới:** Nhập thông tin sinh viên (họ tên, mã số, lớp, điểm).

- **Hiển thị danh sách:** Liệt kê thông tin tất cả sinh viên.
 - **Tìm kiếm:** Tra cứu sinh viên theo mã số, tên hoặc lớp.
 - **Chỉnh sửa/Xóa:** Cập nhật hoặc xóa thông tin sinh viên.
2. **Xử lý dữ liệu:**
- Tính điểm trung bình, xếp loại học lực.
 - Lưu trữ và truy xuất dữ liệu từ cơ sở dữ liệu (SQLite, MySQL).
3. **Giao diện người dùng:**
- Xây dựng giao diện thân thiện bằng Flask hoặc Django.
 - Cho phép tương tác thông qua trình duyệt (thêm, sửa, tìm kiếm, xóa).
4. **Mở rộng:**
- Tích hợp chức năng đăng nhập, phân quyền quản lý (admin, user).
 - Báo cáo và thống kê dữ liệu sinh viên.

Python hỗ trợ tốt với các framework như Flask/Django để tạo ứng dụng web hiệu quả và dễ bảo trì.