

TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN CUỐI KỲ
MÔN HỌC LẬP TRÌNH PYTHON NÂNG CAO

Sinh viên thực hiện:

Trần Ngọc Phước – 2274802010699

LỚP: 241_71ITSE31003_02

Giảng viên hướng dẫn:

Huỳnh Thái Học

TP. Hồ Chí Minh – 11/2024

LỜI CẢM ƠN

Trước hết, em xin gửi lời cảm ơn chân thành đến Thầy Huỳnh Thái Học - người đã tận tình hướng dẫn, cung cấp kiến thức cũng như những lời khuyên quý báu để em hoàn thành đồ án này. Nhờ sự nhiệt huyết và tận tâm của thầy, chúng em đã học hỏi được nhiều kỹ năng cũng như kiến thức thực tiễn bổ ích, giúp hoàn thành đồ án môn học và phát triển thêm kiến thức cho bản thân.

Cuối cùng, em cũng xin gửi lời cảm ơn đến tất cả các bạn cùng lớp đã cùng em trao đổi và thực hiện đồ án đồng thời giúp em có cơ hội học hỏi và nghiên cứu. Những kiến thức từ các môn học đã tạo nền tảng vững chắc cho em trong quá trình thực hiện đồ án.

Em xin chân thành cảm ơn.

TP.HCM, ngày 18 tháng 11 năm 2024

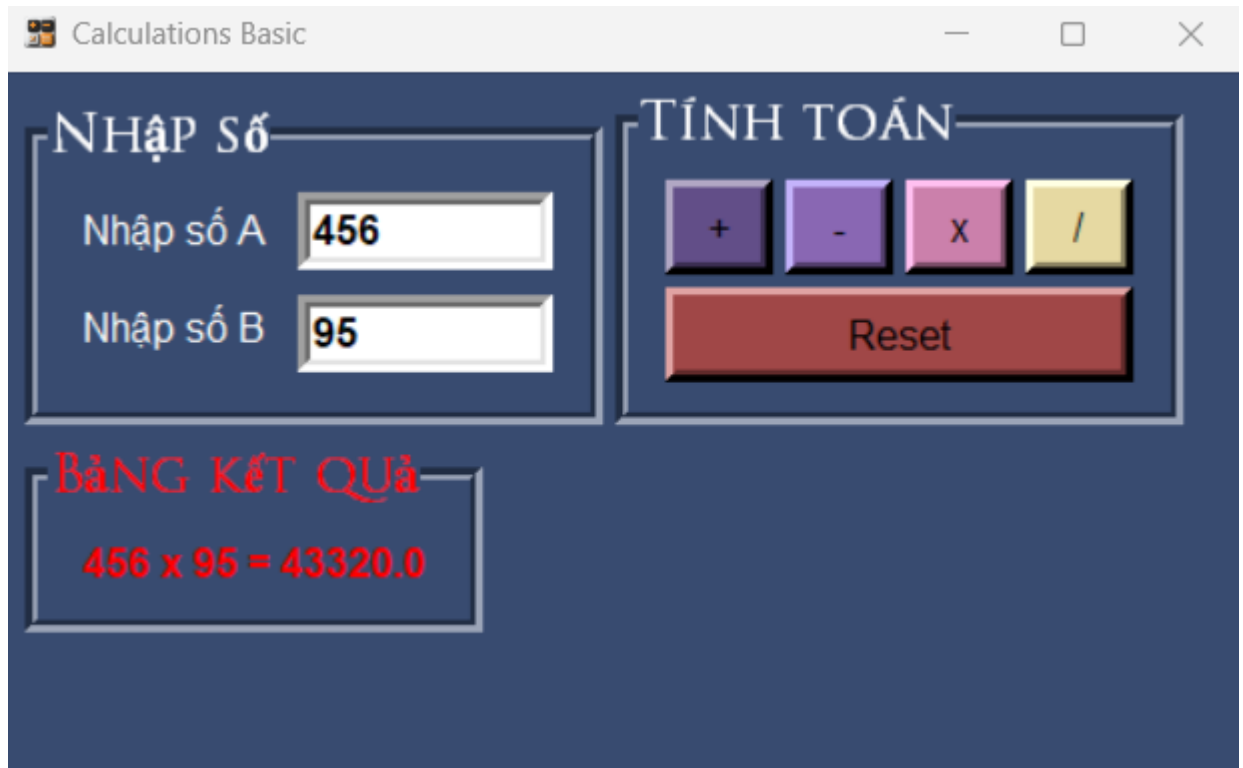
Trần Ngọc Phước

MỤC LỤC

1. Ứng dụng máy tính Calculations Basic.....	4
1.1 Giao diện.....	4
1.2 Chức năng	4
1.2.1 Nhập liệu:	4
1.2.2 Thực hiện các phép toán cơ bản:	4
1.2.3 Nút Reset kết quả:	6
2. Ứng dụng Quản lý sinh viên.	6
2.1 Giao diện.....	6
2.2 Chức năng	6
2.2.1 Load Data từ cơ sở dữ liệu:.....	7
2.2.2 Lọc sinh viên dựa vào lớp học	7
2.2.3 Thêm sinh viên mới vào cơ sở dữ liệu	8
2.3 Mô tả database	8
2.4 Mô tả kiến trúc phần mềm	9
3 Website Quản lý sinh viên.....	10
3.1 Giao diện.....	10
3.2 Chức năng	11
3.2.1 Login vào cơ sở dữ liệu.....	12
3.2.2 Thêm sinh viên mới vào cơ sở dữ liệu	12
3.2.3 Xóa dữ liệu sinh viên.....	13
3.2.4 Tìm kiếm sinh viên dựa vào MSSV.....	13
3.3 Mô tả database	14
3.4 Mô tả kiến trúc phần mềm	14
4 Đường link github.....	15
--- HẾT---	15

1. Ứng dụng máy tính Calculations Basic.

1.1 Giao diện



1.2 Chức năng

Ứng dụng Calculations Basic được xây dựng bằng Python và thư viện Tkinter có các chức năng chính sau:

1.2.1 Nhập liệu:

- Người dùng có thể nhập hai số A và B vào hai ô nhập liệu tương ứng.
- Phần này giúp người dùng cung cấp dữ liệu đầu vào để thực hiện các phép tính.

1.2.2 Thực hiện các phép toán cơ bản:

- Cộng (+): Nút này cho phép thực hiện phép cộng hai số A và B đã nhập. Khi người dùng nhấn vào, kết quả của phép cộng sẽ hiển thị ở phần kết quả.

```
# create function cong
def cong():
    try:
        kq = float(entryA.get()) + float(entryB.get())
        label_kq.config(text= entryA.get() + ' + ' + entryB.get() + ' = ' + str(kq), fg=color_text_kq)
        label_Frame3.config(fg=color_text_kq)
    except:
        messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ!")
        entryA.delete(0, tk.END)
        entryB.delete(0, tk.END)
```

- Trừ (-): Thực hiện phép trừ giữa số A và số B, với kết quả cũng hiển thị ngay lập tức.

```
# create function tru
def tru():
    try:
        kq = float(entryA.get()) - float(entryB.get())
        label_kq.config(text=entryA.get() + ' - ' + entryB.get() + ' = ' + str(kq), fg=color_text_kq)
        label_Frame3.config(fg=color_text_kq)
    except:
        messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ!")
        entryA.delete(0, tk.END)
        entryB.delete(0, tk.END)
```

- Nhân (x): Tính tích của hai số A và B.

```
# create function nhan
def nhan():
    try:
        kq = float(entryA.get()) * float(entryB.get())
        label_kq.config(text=entryA.get() + ' x ' + entryB.get() + ' = ' + str(kq), fg=color_text_kq)
        label_Frame3.config(fg=color_text_kq)
    except:
        messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ!")
        entryA.delete(0, tk.END)
        entryB.delete(0, tk.END)
```

- Chia (/): Tính thương của số A chia cho số B. Nếu người dùng nhập số B bằng 0, thì ứng dụng hiển thị thông báo lỗi, "Không thể chia cho 0"

```
# create function chia
def chia():
    try:
        b = float(entryB.get())
        if b==0:
            messagebox.showerror('Lỗi', "Không thể chia cho 0")
        else:
            kq = float(entryA.get()) / b
            label_kq.config(text=entryA.get() + ' / ' + entryB.get() + ' = ' + str(kq), fg=color_text_kq)
            label_Frame3.config(fg=color_text_kq)
    except:
        messagebox.showerror("Lỗi", "Vui lòng nhập số hợp lệ!")
        entryA.delete(0, tk.END)
        entryB.delete(0, tk.END)
```

1.2.3 Nút Reset kết quả:

- Chức năng này xóa toàn bộ dữ liệu đã nhập và kết quả đang hiển thị, giúp người dùng làm mới ứng dụng và bắt đầu tính toán lại từ đầu.

```
# create function reset
def reset():
    entryA.delete(0, tk.END)
    entryB.delete(0, tk.END)
    label_kq.config(text='', fg=color_text)
    label_Frame3.config(fg=color_text)
```

2. Ứng dụng Quản lý sinh viên.

2.1 Giao diện

STT	HỌ VÀ TÊN	MSSV	LỚP
1	Trần Ngọc phước	2274802010699	CNTT20
2	Trần Ngọc Phước	123456789	CNTT20
3	Trần Ngọc Phước	13456789	CNTT20
4	Trần Ngọc Phước	1315456789	CNTT22
5	Trần Ngọc Phước	13515456789	CNTT22
6	Trần Nhật Nam	151511	CNTT23
7	Vũ Thành Đạt	156518489	CNTT23
8	Vũ Thành Đạt	1658165616	CNTT23
9	Nguyễn Văn Bảo	16581656165	CNTT23
10	Lê Bảo Bình	215115515	CNTT24

2.2 Chức năng

Ứng dụng Quản lý sinh viên được xây dựng bằng Python và thư viện psycopg2 có các chức năng chính sau:

2.2.1 Load Data từ cơ sở dữ liệu:

```
55 def connect_db(self):
56     try:
57         self.conn = psycopg2.connect(
58             dbname=self.db_name.get(),
59             user=self.user.get(),
60             password=self.password.get(),
61             host=self.host.get(),
62             port=self.port.get()
63         )
64         self.cur = self.conn.cursor()
65         messagebox.showinfo("Success", "Connected to the database successfully!")
66         self.connection_frame.grid_forget() # Ẩn khung đăng nhập sau khi thành công
67         self.background_label.place_forget() # Ẩn hình nền
68         self.create_main_screen()
69     except Exception as e:
70         messagebox.showerror("Error", f"Error connecting to the database: {e}")
```

```
162 def load_data(self):
163     try:
164         query = sql.SQL("SELECT * FROM {}").format(sql.Identifier(self.table_name.get()))
165         self.database.cur.execute(query)
166         rows = self.database.cur.fetchall()
167         self.tree.delete(*self.tree.get_children()) # Clear previous data
168         for index, row in enumerate(rows, start=1):
169             self.tree.insert('', 'end', values=(index, row[1], row[2], row[3])) # Assumes columns are [mssv, name, class]
170     except Exception as e:
171         messagebox.showerror("Error", f"Không tìm thấy dữ liệu: {e}")
172
```

2.2.2 Lọc sinh viên dựa vào lớp học

```
201 def search_data(self):
202     try:
203         search_query = sql.SQL("SELECT * FROM {} WHERE class = %s").format(sql.Identifier(self.table_name.get()))
204         self.database.cur.execute(search_query, (self.search_value.get(),))
205         rows = self.database.cur.fetchall()
206
207         # Clear previous data from Treeview
208         for item in self.tree.get_children():
209             self.tree.delete(item)
210
211         # Display search results
212         if rows:
213             for idx, row in enumerate(rows, start=1):
214                 self.tree.insert("", "end", values=(idx, row[1], row[2], row[3])) # Assuming the order of columns is (mssv, name, class)
215         else:
216             messagebox.showinfo("No Data", "Không tìm thấy lớp này!")
217     except Exception as e:
218         messagebox.showerror("Error", f"Error searching data: {e}")
```

2.2.3 Thêm sinh viên mới vào cơ sở dữ liệu

```
175 def insert_data(self):
176     name = self.column1.get()
177     mssv = self.column2.get()
178     student_class = self.column3.get()
179
180     if not name or not mssv or not student_class:
181         messagebox.showerror("Error", "Không được bỏ trống ô nhập liệu!")
182         return
183
184     try:
185         # Check if MSSV already exists
186         check_query = sql.SQL("SELECT 1 FROM {} WHERE mssv = %s").format(sql.Identifier(self.table_name.get()))
187         self.database.cur.execute(check_query, (mssv,))
188         if self.database.cur.fetchone():
189             messagebox.showerror("Error", "MSSV đã tồn tại!")
190             return
191
192         insert_query = sql.SQL("INSERT INTO {} (name, mssv, class) VALUES (%s, %s, %s)").format(sql.Identifier(self.table_name.get()))
193         self.database.cur.execute(insert_query, (name, mssv, student_class))
194         self.database.conn.commit()
195         messagebox.showinfo("Success", "Thêm sinh viên mới thành công!")
196
197     except Exception as e:
198         messagebox.showerror("Error", f"Error inserting data: {e}")
```

2.3 Mô tả database

Ứng dụng quản lý sinh viên có một bảng chính trong cơ sở dữ liệu:

khoacntt

General

Columns

Advanced

Constraints

Partitions

Parameters

Security

SQL

Inherited from table(s)

Select to inherit from...

Columns

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
<div><div></div><div></div></div>	id	bigint <div></div>			<div></div>	<div></div>	nextval('')
<div><div></div><div></div></div>	name	character varying <div></div>	200		<div></div>	<div></div>	
<div><div></div><div></div></div>	mssv	character varying <div></div>	13		<div></div>	<div></div>	
<div><div></div><div></div></div>	class	character varying <div></div>	20		<div></div>	<div></div>	

Quan hệ dữ liệu:

Bảng khoacntt là bảng lưu trữ chính cho thông tin sinh viên.

Trường “mssv” được định nghĩa là khóa chính, đảm bảo không có sự trùng lặp trong mã số sinh viên.

2.4 Mô tả kiến trúc phần mềm

Ứng dụng này được xây dựng dựa trên kiến trúc **Client-Server**, trong đó:

a. Phần Client (Giao diện người dùng):

- Được triển khai bằng **Tkinter**, tạo giao diện trực quan cho người dùng.
- Các thành phần chính bao gồm:
 - **Màn hình đăng nhập:** Nhập thông tin cơ sở dữ liệu (username, password) để kết nối vào database.
 - **Màn hình chính:** Cung cấp các chức năng:
 - Hiển thị danh sách sinh viên.
 - Thêm sinh viên mới.
 - Lọc danh sách sinh viên theo lớp.
- Các chức năng giao diện tương tác với cơ sở dữ liệu thông qua các lệnh được thực thi bởi lớp Database.

b. Phần Server (Quản lý dữ liệu):

- Sử dụng **PostgreSQL** để quản lý dữ liệu sinh viên.
- Các thao tác như thêm, truy vấn, kiểm tra dữ liệu đều được thực hiện thông qua kết nối cơ sở dữ liệu bằng thư viện **psycopg2**.
- Lớp Database (được import từ **database.py**) chịu trách nhiệm:
 - Kết nối và quản lý phiên làm việc với cơ sở dữ liệu.
 - Thực thi các truy vấn SQL cần thiết (như SELECT, INSERT, WHERE).

3 Website Quản lý sinh viên.

3.1 Giao diện

The image displays two screenshots of a web application interface for student management.

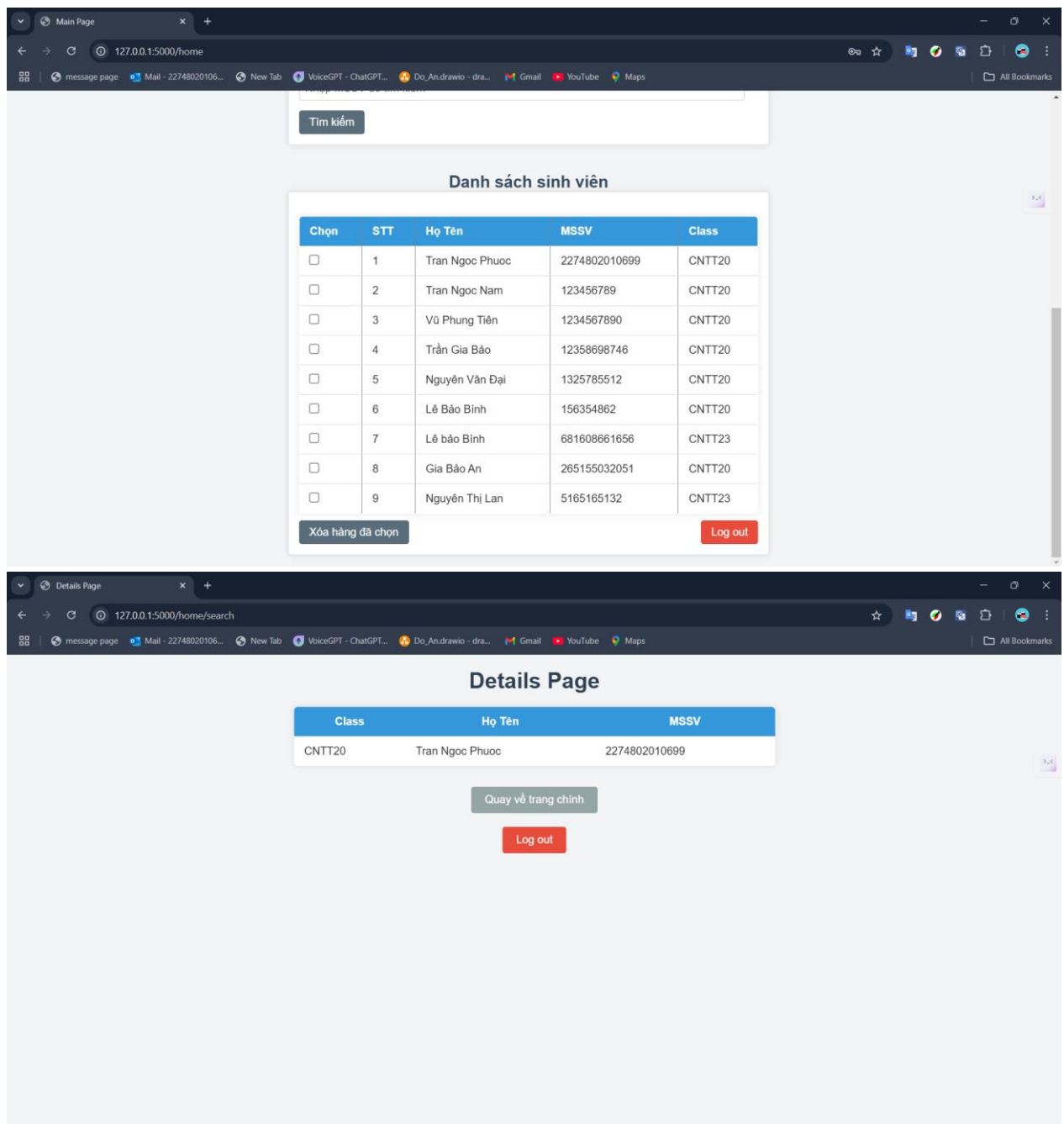
The top screenshot shows a login screen overlaid on a background image of Van Lang University. The login form is titled "Login" and contains the following fields:

- User: postgres
- Password:

Below the password field is a "Log in" button.

The bottom screenshot shows the "Main Page" of the application. It features two main sections:

- Thêm mới dữ liệu** (Add new data): This section contains three input fields for "Họ Tên:" (Full Name), "MSSV:" (Student ID), and "Class:". Below these fields is a "Thêm" (Add) button.
- Tìm kiếm sinh viên** (Search student): This section contains an input field labeled "Nhập MSSV:" (Enter MSSV) with the placeholder text "Nhập MSSV để tìm kiếm" (Enter MSSV to search). Below the input field is a "Tìm kiếm" (Search) button.



3.2 Chức năng

Website Quản lý sinh viên được xây dựng bằng Python và thư viện flask có các chức năng chính sau:

3.2.1 Login vào cơ sở dữ liệu

```
10 # Trang đăng nhập
11 @app.route('/', methods=['GET', 'POST'])
12 def login():
13     if request.method == 'POST':
14         user = request.form['user']
15         password = request.form['password']
16         db = Database(db_name='quanlysinhvien', user=user, password=password, host='localhost', port=5432)
17         if db.connect():
18             session['user'] = user
19             session['password'] = password
20             db.cur.close()
21             db.conn.close()
22             return redirect(url_for('home'))
23         else:
24             flash("Kết nối không thành công. Vui lòng kiểm tra thông tin đăng nhập.", 'error')
25     return render_template('index.html')

28 # Trang chính
29 @app.route('/home', methods=['GET', 'POST'])
30 def home():
31     if 'user' in session and 'password' in session:
32         db = Database(db_name='quanlysinhvien', user=session['user'], password=session['password'], host='localhost', port=5432)
33         if db.connect():
34             db.cur.execute('SELECT * FROM khoaCNTT ORDER BY id')
35             data = db.cur.fetchall()
36             db.cur.close()
37             db.conn.close()
38             return render_template('home.html', data=data)
39     flash("Bạn cần đăng nhập để truy cập trang này.", 'error')
40     return redirect(url_for('login'))
```

3.2.2 Thêm sinh viên mới vào cơ sở dữ liệu

```
43 # Thêm dữ liệu mới
44 @app.route('/home/add', methods=['POST'])
45 def add_data():
46     if 'user' in session and 'password' in session:
47         new_name = request.form.get('name', '').strip()
48         new_mssv = request.form.get('mssv', '').strip()
49         new_class = request.form.get('class', '').strip()
50
51         if not new_name or not new_mssv or not new_class:
52             flash("Tất cả các trường (name, mssv, class) đều bắt buộc.", 'error')
53             return redirect(url_for('home'))
54         db = Database(db_name='quanlysinhvien', user=session['user'], password=session['password'], host='localhost', port=5432)
55         if db.connect():
56             try:
57                 # Kiểm tra mssv trùng lặp
58                 db.cur.execute("SELECT COUNT(*) FROM khoaCNTT WHERE mssv = %s", (new_mssv,))
59                 if db.cur.fetchone()[0] > 0:
60                     flash("MSSV đã tồn tại, vui lòng nhập MSSV khác.", 'error')
61                 else:
62                     # Thêm dữ liệu mới
63                     db.cur.execute("INSERT INTO khoaCNTT (name, mssv, class) VALUES (%s, %s, %s)",
64                                   (new_name, new_mssv, new_class))
65                     db.conn.commit()
66                     flash("Thêm dữ liệu thành công.", 'success')
67             except Exception as e:
68                 flash(f"Lỗi khi thêm dữ liệu: {e}", 'error')
69             finally:
70                 db.cur.close()
71                 db.conn.close()
72         return redirect(url_for('home'))
```

3.2.3 Xóa dữ liệu sinh viên

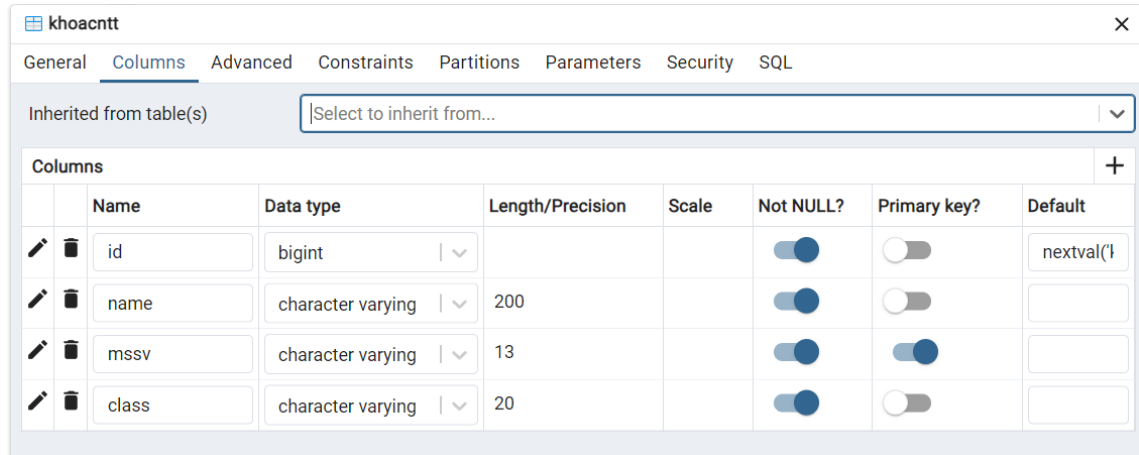
```
75 # Xóa dữ liệu đã chọn
76 @app.route('/home/delete', methods=['POST'])
77 def delete_data():
78     if 'user' in session and 'password' in session:
79         selected_ids = request.form.getlist('selected')
80         if not selected_ids:
81             flash("Vui lòng chọn hàng cần xóa.", 'error')
82             return redirect(url_for('home'))
83
84         db = Database(db_name='quanlysinhvien', user=session['user'], password=session['password'], host='localhost', port=5432)
85         if db.connect():
86             try:
87                 db.cur.execute(
88                     "DELETE FROM khoacntt WHERE id IN (%s)" % ','.join(map(str, selected_ids))
89                 )
90                 db.conn.commit()
91                 flash("Xóa dữ liệu thành công.", 'success')
92             except Exception as e:
93                 flash(f"Lỗi khi xóa dữ liệu: {e}", 'error')
94             finally:
95                 db.cur.close()
96                 db.conn.close()
97
98     return redirect(url_for('home'))
```









3.2.4 Tìm kiếm sinh viên dựa vào MSSV

```
99 # Tìm kiếm sinh viên theo MSSV
100 @app.route('/home/search', methods=['POST'])
101 def search_data():
102     if 'user' in session and 'password' in session:
103         mssv = request.form['mssv'] # Lấy MSSV từ form tìm kiếm
104         db = Database(db_name='quanlysinhvien', user=session['user'], password=session['password'], host='localhost', port=5432)
105         if db.connect():
106             try:
107                 # Tìm kiếm sinh viên theo MSSV
108                 query = "SELECT * FROM khoacntt WHERE mssv = %s"
109                 db.cur.execute(query, (mssv,))
110                 data = db.cur.fetchall()
111
112                 db.cur.close()
113                 db.conn.close()
114
115                 if data:
116                     # Truyền kết quả tìm kiếm vào trang details.html
117                     return render_template('details.html', data=data)
118                 else:
119                     flash("Không tìm thấy sinh viên với MSSV đã nhập.", 'error')
120             except Exception as e:
121                 flash(f"Lỗi khi tìm kiếm: {e}", 'error')
122             finally:
123                 db.cur.close()
124                 db.conn.close()
125
126     return redirect(url_for('home'))
```

3.3 Mô tả database

Website quản lý sinh viên có một bảng chính trong cơ sở dữ liệu:



	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
 	id	bigint			<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval(''id_seq'')
 	name	character varying	200		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 	mssv	character varying	13		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
 	class	character varying	20		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Quan hệ dữ liệu:

Bảng khoacntt là bảng lưu trữ chính cho thông tin sinh viên.

Trường “mssv” được định nghĩa là khóa chính, đảm bảo không có sự trùng lặp trong mã số sinh viên.

3.4 Mô tả kiến trúc phần mềm

Website quản lý sinh viên được chia thành các lớp chính:

a. Lớp giao diện người dùng (UI - Frontend):

- **Công nghệ sử dụng:** HTML, CSS (kết hợp với Flash Messages của Flask để hiển thị thông báo).
- **Chức năng:**
 - Cung cấp các trang giao diện như:
 - Trang đăng nhập (index.html).
 - Trang chính hiển thị danh sách sinh viên (home.html).
 - Trang chi tiết tìm kiếm sinh viên (details.html).
 - Giao diện form để thêm, xóa, tìm kiếm sinh viên.

b. Lớp xử lý logic (Backend):

- **Công nghệ sử dụng:** Python, Flask.
- **Chức năng:**
 - Quản lý luồng điều hướng giữa các trang.

- Xử lý form và kiểm tra dữ liệu đầu vào.
- Tương tác với cơ sở dữ liệu thông qua lớp kết nối Database.
- Cung cấp API để tìm kiếm, thêm mới, hoặc xóa sinh viên.

c. Lớp truy cập dữ liệu (Database Layer):

- **Công nghệ sử dụng:** PostgreSQL.
- **Cấu trúc cơ sở dữ liệu:** Thông qua lớp Database được import từ module database.py.
- **Chức năng:**
 - Kết nối với cơ sở dữ liệu PostgreSQL.
 - Thực thi các truy vấn như thêm, xóa, hoặc tìm kiếm dữ liệu sinh viên.

4 Đường link github.

<https://github.com/phuocsasc/PythonNC>

--- HẾT---