

TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN CUỐI KỲ
MÔN HỌC LẬP TRÌNH PYTHON NÂNG CAO

Bài tập 02:

Quản lý sinh viên

SVTH: Trần Ngọc Phước – 2274802010699

LỚP: 241_71ITSE31003_02

GVHD: Huỳnh Thái Học

TP. Hồ Chí Minh – 10/2024

MỤC LỤC

1. Giới thiệu về psycopg2.....	3
2. Giao diện ứng dụng Quản lý sinh viên.....	4
3. Chức năng.....	6
3.1 Load Data từ cơ sở dữ liệu:	6
3.2 Tìm kiếm sinh viên dựa vào lớp học:	6
3.3 Thêm sinh viên mới vào cơ sở dữ liệu:.....	6
4. Mã chương trình.....	7
5. Đường link github.....	10
--- HẾT---	10

1. Giới thiệu về psycopg2.

psycopg2 là một thư viện Python nổi bật được sử dụng để kết nối và tương tác với cơ sở dữ liệu PostgreSQL. Nó cung cấp các API mạnh mẽ, tuân thủ chuẩn DB-API 2.0 của Python, giúp các lập trình viên thực hiện các thao tác cơ sở dữ liệu như kết nối, truy vấn, và quản lý dữ liệu một cách hiệu quả.

Với khả năng hỗ trợ các tính năng nâng cao của PostgreSQL như xử lý giao dịch, kiểu dữ liệu tùy chỉnh (như ARRAY, JSON, UUID), và hstore, psycopg2 trở thành lựa chọn lý tưởng cho các ứng dụng yêu cầu hiệu suất cao và quản lý dữ liệu an toàn. Bên cạnh đó, thư viện này cũng hỗ trợ các mô hình lập trình đa luồng và bất đồng bộ, giúp tối ưu hóa hiệu năng trong các ứng dụng phức tạp.

2. Giao diện ứng dụng Quản lý sinh viên.

Giao diện để login vào ứng dụng



Giao diện khi login thành công vào ứng dụng

QUẢN LÝ SINH VIÊN

LOAD DATA

Nhập tên bảng:

SEARCH STUDENT

Nhập lớp:


BẢNG THÔNG TIN SINH VIÊN

STT	LỚP	HỌ VÀ TÊN
1	CNTT20	Trần Ngọc Phước
2	CNTT20	Nguyễn Thanh Quốc
3	CNTT20	Nguyễn Nhật Nam
4	CNTT21	Trần Văn Bảy
5	CNTT21	Lê Thị Thanh
6	CNTT21	Phan Lâm Tuyển
7	CNTT21	Vũ Gia Bảo
8	CNTT22	Nguyễn Huệ
9	CNTT22	Nguyễn Văn An
10	CNTT22	Trần Thị Thắm

INSERT DATA

HỌ VÀ TÊN:

LỚP:



Giao diện khi Search thông tin sinh viên dựa vào mã lớp học

QUẢN LÝ SINH VIÊN

LOAD DATA

Nhập tên bảng:

SEARCH STUDENT

Nhập lớp:


BẢNG THÔNG TIN SINH VIÊN

STT	LỚP	HỌ VÀ TÊN
1	CNTT22	Nguyễn Huệ
2	CNTT22	Nguyễn Văn An
3	CNTT22	Trần Thị Thắm

INSERT DATA

HỌ VÀ TÊN:

LỚP:



3.Chức năng.

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

3.1 Load Data từ cơ sở dữ liệu:

- Người dùng có thể nhập tên danh sách sinh viên và ô “Nhập tên bảng”
- Phần này người dùng cung cấp dữ liệu đầu vào là “danhsach” và nhấn nút “Load Data”, ứng dụng sẽ tải toàn bộ danh sách sinh viên có trong cơ sở dữ liệu và hiển thị vào “BẢNG THÔNG TIN SINH VIÊN”

3.2 Tìm kiếm sinh viên dựa vào lớp học:

- Người dùng nhập lớp cần tìm vào ô “Nhập lớp”
- Phần này người dùng cung cấp dữ liệu đầu vào là “mã lớp” và nhấn nút “Seacrh”, ứng dụng sẽ hiển thị toàn bộ danh sách sinh viên của lớp học đó vào “BẢNG THÔNG TIN SINH VIÊN”.

3.3 Thêm sinh viên mới vào cơ sở dữ liệu:

- Người dùng nhập tên và lớp vào hai ô nhập liệu tương ứng.
- Phần này người dùng cung cấp dữ liệu đầu vào là “tên sinh viên mới” và “Mã lớp học” khi nhấn nút “Insert Data”, ứng dụng sẽ thêm sinh viên mới đó vào mã lớp học tương ứng.

4. Mã chương trình.

Về giao diện:

- Toàn bộ giao diện được sắp xếp gọn gàng, sử dụng các phương pháp quản lý bố cục grid() và LabelFrame để tạo các khối nội dung phân tách rõ ràng, gồm ba phần chính: giao diện login, giao diện hệ thống chứa các chức năng chính.

```
28 def create_login_screen(self):
29     font_title = font.Font(family='Trajan Pro', size=16,)
30     font_text = font.Font(family='Eccentric Std', size=12)
31     font_bold = font.Font(family='Eccentric Std', size=12, 'bold')
32
33     # Mờ ảnh nền bằng Pillow
34     image = Image.open("imgs/dai-hoc-tot-o-vn-2.png") # Đường dẫn đến file ảnh
35     image = image.resize((500, 500), Image.ANTIALIAS) # Điều chỉnh kích thước ảnh
36     bg_image = ImageTk.PhotoImage(image)
37
38     # Tạo một Label để chứa ảnh nền
39     self.background_label = tk.Label(self.root, image=bg_image)
40     self.background_label.place(x=0, y=0, relwidth=1, relheight=1)
41     # Giữ tham chiếu đến ảnh nền để tránh bị garbage collected
42     self.background_label.image = bg_image
43
44     self.connection_frame = tk.LabelFrame(self.root, text="Login", font=font_title, bd=5, bg="#FEF9F2", pady=10, padx=10, labelanchor='n')
45     self.connection_frame.grid(row=0, column=0, padx=70, pady=(100,200))
46
47     tk.Label(self.connection_frame, text="User:", font=font_text, bg="#FEF9F2", ).grid(row=0, column=0, padx=5, pady=5, sticky='w')
48     tk.Entry(self.connection_frame, textvariable=self.user, font=font_text, bd=5).grid(row=0, column=1, padx=5, pady=5)
49
50     tk.Label(self.connection_frame, text="Password:", font=font_text, bg="#FEF9F2", ).grid(row=1, column=0, padx=5, pady=5, sticky='w')
51     tk.Entry(self.connection_frame, textvariable=self.password, show="*", font=font_text, bd=5).grid(row=1, column=1, padx=5, pady=5)
52
53     tk.Button(self.connection_frame, text="Connect", font=font_bold, command=self.connect_db, bd=5, width=10, bg='#77CDDF').grid(row=2, column=1,
54
```

```
72 def create_main_screen(self):
73     font_title = font.Font(family='Trajan Pro', size=16,)
74     font_text = font.Font(family='Eccentric Std', size=12)
75     font_bold = font.Font(family='Eccentric Std', size=12, 'bold')
76
77     # Đặt màu nền là màu #384B70 cho giao diện chính
78     self.root['background'] = '#384B70'
79
80     label_title = tk.Label(self.root, text="QUẢN LÝ SINH VIÊN", font=font_title, fg="white", bg='#384B70')
81     label_title.grid(row=0, column=0, colspan=2, pady=(10,0))
82
83     # Query section
84     query_frame = tk.LabelFrame(self.root, text="Load Data", width=200, font=font_title, fg="white", bg='#384B70', bd=5)
85     query_frame.grid(row=1, column=0, padx=10, pady=10, sticky='w')
86
87     tk.Label(query_frame, text="Nhập tên bảng:", font=font_text, bg='#384B70', fg='white').grid(row=0, column=0, padx=5, pady=5)
88     tk.Entry(query_frame, textvariable=self.table_name, font=font_text, bd=5).grid(row=0, column=1, padx=5, pady=5)
89     tk.Button(query_frame, text="Load Data", command=self.load_data, font=font_bold, bg='#D4BDAC', bd=2).grid(row=0, column=2, columnspan=1,
90
91     label_titledb = tk.Label(self.root, text="BẢNG THÔNG TIN SINH VIÊN", font=font_title, fg="#FFB74D", bg='#384B70')
92     label_titledb.grid(row=2, column=0, padx=10, sticky='w')
93
94     # Data display table
95     style = ttk.Style()
96     style.configure("Treeview.Hheading", background="#384B70", foreground="#9A7E6F", font=font_bold)
97
98     self.tree = ttk.Treeview(self.root, columns=('STT', 'LỚP', 'HỌ VÀ TÊN'), show='headings', height=10)
```

```

100 self.tree.column('STT', width=50, anchor='center')
101 self.tree.column('LỚP', width=140, anchor='center')
102 self.tree.column('HỌ VÀ TÊN', width=230, anchor='w')
103
104 # Set heading name
105 self.tree.heading('STT', text='STT')
106 self.tree.heading('LỚP', text='LỚP')
107 self.tree.heading('HỌ VÀ TÊN', text='HỌ VÀ TÊN')
108
109 self.tree.grid(row=3, column=0, padx=5, pady=5, )
110
111 # Insert section
112 insert_frame = tk.LabelFrame(self.root, text="Insert Data", font=font_title, fg="white", bg='#384B70', labelanchor='n', bd=5)
113 insert_frame.grid(row=4, column=0, columnspan=1, padx=10, pady=10)
114
115 self.column1 = tk.StringVar()
116 self.column2 = tk.StringVar()
117
118 tk.Label(insert_frame, text="HỌ VÀ TÊN:", font=font_text, bg='#384B70', fg='white').grid(row=0, column=0, padx=5, pady=5)
119 tk.Entry(insert_frame, textvariable=self.column1, width=33, font=font_text, bd=5).grid(row=0, column=1, padx=5, pady=5)
120
121 tk.Label(insert_frame, text="LỚP", font=font_text, bg='#384B70', fg='white').grid(row=1, column=0, padx=5, pady=5, sticky='w')
122 tk.Entry(insert_frame, textvariable=self.column2, width=33, font=font_text, bd=5).grid(row=1, column=1, padx=5, pady=5)
123
124 tk.Button(insert_frame, text="Insert Data", command=self.insert_data, width=15, font=font_bold, bg="#D4BDAC", bd=5).grid(row=2, column=1, padx=5, pady=5)
125
126 # Search section
127 search_frame = tk.LabelFrame(self.root, text="Search Student", font=font_title, fg="white", bg='#384B70', bd=5)
128 search_frame.grid(row=1, column=1, sticky='w', padx=(0,10))

```

```

126 # Search section
127 search_frame = tk.LabelFrame(self.root, text="Search Student", font=font_title, fg="white", bg='#384B70', bd=5)
128 search_frame.grid(row=1, column=1, sticky='w', padx=(0,10))
129
130 self.search_value = tk.StringVar()
131
132 tk.Label(search_frame, text="Nhập lớp:", font=font_text, bg='#384B70', fg='white').grid(row=0, column=0, padx=5, pady=5)
133 tk.Entry(search_frame, textvariable=self.search_value, font=font_text, bd=5).grid(row=0, column=1, padx=5, pady=5)
134
135 tk.Button(search_frame, text="Search", command=self.search_data, font=font_bold, bg="#D4BDAC", bd=2).grid(row=0, column=2, padx=5, pady=5)
136
137 # thêm ảnh vào root
138 image = Image.open("imgs/dai-hoc-tot-o-vn-2.png") # Đường dẫn đến ảnh của bạn
139 # Thay đổi kích thước ảnh nếu cần
140 image = image.resize((380, 420), Image.ANTIALIAS) # Điều chỉnh kích thước (nếu cần)
141 # Chuyển ảnh thành định dạng tkinter
142 photo = ImageTk.PhotoImage(image)
143 # Tạo Label và chèn ảnh vào
144 label = tk.Label(root, image=photo)
145 label.grid(row=3, column=1, rowspan=2, pady=10)
146 # Giữ tham chiếu tới ảnh để tránh bị garbage collected
147 label.image = photo
148

```


Về các chức năng chính:

- Kết nối vào database

```
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}")
```

- Load Data

```
149 def load_data(self):
150     try:
151         query = sql.SQL("SELECT * FROM {}").format(sql.Identifier(self.table_name.get()))
152         self.cur.execute(query)
153         rows = self.cur.fetchall()
154         self.tree.delete(*self.tree.get_children()) # Clear previous data
155         for index, row in enumerate(rows, start=1):
156             self.tree.insert('', 'end', values=(index, row[0], row[1])) # Assumes columns are [name, mssv, major]
157     except Exception as e:
158         messagebox.showerror("Error", f"Error loading data: {e}")
```

- Insert data

```
160 def insert_data(self):
161     try:
162         insert_query = sql.SQL("INSERT INTO {} (hoten, class) VALUES (%s, %s)").format(sql.Identifier(self.table_name.get()))
163         data_to_insert = (self.column1.get(), self.column2.get())
164         self.cur.execute(insert_query, data_to_insert)
165         self.conn.commit()
166         messagebox.showinfo("Success", "Data inserted successfully!")
167     except Exception as e:
168         messagebox.showerror("Error", f"Error inserting data: {e}")
```

- Search data

```
171 def search_data(self):
172     try:
173         search_query = sql.SQL("SELECT * FROM {} WHERE class = %s").format(sql.Identifier(self.table_name.get()))
174         self.cur.execute(search_query, (self.search_value.get(),))
175         rows = self.cur.fetchall()
176
177         # Clear previous data from Treeview
178         for item in self.tree.get_children():
179             self.tree.delete(item)
180
181         # Display search results
182         if rows:
183             for idx, row in enumerate(rows, start=1):
184                 self.tree.insert("", "end", values=(idx, row[0], row[1])) # Assuming the order of columns is (hoten, mssv)
185         else:
186             messagebox.showinfo("No Data", "No data found for the given MSSV.")
187     except Exception as e:
188         messagebox.showerror("Error", f"Error searching data: {e}")
```

5.Đường link github.

<https://github.com/phuocsasc/PythonNC/tree/master/Baitap01>

--- HẾT---