

Họ tên: Lê Thúy Hiền  
MSSV: 2274802010236

## BÀI 2: DATABASE

### 1. Giao diện

The image displays two screenshots of a Python Tkinter application titled "Quản lý Sinh viên" (Student Management). The left screenshot shows the "Kết nối CSDL" (Connect Database) tab, which contains input fields for "Tên CSDL" (students), "User" (postgres), "Mật khẩu" (password), "Host" (localhost), "Cổng" (5432), and "Tên bảng" (students). Below these fields are buttons for "Kết nối" (Connect) and "Tải dữ liệu" (Load data). The right screenshot shows the "Quản lý Sinh viên" tab, which features a table with columns "ID", "Tên", "Tuổi", "Giới tính", and "Ngành". Below the table are input fields for "Tên", "Tuổi", "Giới tính", and "Ngành học", and buttons for "Thêm sinh viên", "Cập nhật sinh viên", "Xóa sinh viên", and "Thống kê".

### 2. Chức năng

- Kết nối database
- Thêm mới sinh viên
- Cập nhật sinh viên
- Xóa sinh viên
- Thống kê: số sinh viên, ngành học và giới tính

Ngoài ra, ứng dụng còn hiển thị các thông báo thành công, lỗi trong các trường hợp khác.

### 3. Mã nguồn

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

class DatabaseApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Quản lý Sinh viên")
        self.root.geometry("650x500")

        # Tạo Notebook cho các tab
        self.notebook = ttk.Notebook(self.root)
        self.notebook.pack(fill='both', expand=True)

        # Tạo tab Kết nối CSDL
        self.tab_connect = ttk.Frame(self.notebook)
        self.notebook.add(self.tab_connect, text='Kết nối CSDL')

        # Tạo tab Quản lý Sinh viên
```

```

self.tab_student = ttk.Frame(self.notebook)
self.notebook.add(self.tab_student, text='Quản lý Sinh viên')

self.create_connect_tab()
self.create_student_tab()

def create_connect_tab(self):
    # Các trường kết nối cơ sở dữ liệu
    self.db_name = tk.StringVar(value='students')
    self.user = tk.StringVar(value='postgres')
    self.password = tk.StringVar(value='123456')
    self.host = tk.StringVar(value='localhost')
    self.port = tk.StringVar(value='5432')
    self.table_name = tk.StringVar(value='students')

    # Frame chứa các widget
    connect_frame = tk.Frame(self.tab_connect)
    connect_frame.pack(pady=20)

    tk.Label(connect_frame, text="Tên CSDL:").grid(row=0, column=0, padx=5,
pady=5)
    tk.Entry(connect_frame, textvariable=self.db_name).grid(row=0, column=1,
padx=5, pady=5)

    tk.Label(connect_frame, text="User:").grid(row=1, column=0, padx=5, pady=5)
    tk.Entry(connect_frame, textvariable=self.user).grid(row=1, column=1, padx=5,
pady=5)

    tk.Label(connect_frame, text="Mật khẩu:").grid(row=2, column=0, padx=5,
pady=5)
    tk.Entry(connect_frame, textvariable=self.password, show="*").grid(row=2,
column=1, padx=5, pady=5)

    tk.Label(connect_frame, text="Host:").grid(row=3, column=0, padx=5, pady=5)
    tk.Entry(connect_frame, textvariable=self.host).grid(row=3, column=1, padx=5,
pady=5)

    tk.Label(connect_frame, text="Cổng:").grid(row=4, column=0, padx=5, pady=5)
    tk.Entry(connect_frame, textvariable=self.port).grid(row=4, column=1, padx=5,
pady=5)

    tk.Button(connect_frame, text="Kết nối", command=self.connect_db).grid(row=5,
columnspan=2, pady=10)

    tk.Label(connect_frame, text="Tên bảng:").grid(row=6, column=0, padx=5,
pady=5)
    tk.Entry(connect_frame, textvariable=self.table_name).grid(row=6, column=1,
padx=5, pady=5)

```

```

        tk.Button(connect_frame, text="Tải dữ liệu",
command=self.load_data).grid(row=7, columnspan=2, pady=10)

        self.data_display = tk.Text(connect_frame, height=10, width=50)
        self.data_display.grid(row=8, column=0, columnspan=2, padx=10, pady=10)

    def create_student_tab(self):
        # Thiết lập Treeview
        columns = ("ID", "Tên", "Tuổi", "Giới tính", "Ngành")
        self.tree = ttk.Treeview(self.tab_student, columns=columns, show="headings",
height=8)

        for col in columns:
            self.tree.heading(col, text=col)
            self.tree.column(col, anchor=tk.CENTER, width=120)

        self.tree.grid(row=0, column=0, columnspan=4, padx=10, pady=10)

        # Các trường thông tin sinh viên
        tk.Label(self.tab_student, text="Tên:").grid(row=1, column=0, padx=10,
pady=5)
        self.entry_name = tk.Entry(self.tab_student)
        self.entry_name.grid(row=1, column=1, padx=10, pady=5)

        tk.Label(self.tab_student, text="Tuổi:").grid(row=1, column=2, padx=10,
pady=5)
        self.entry_age = tk.Entry(self.tab_student)
        self.entry_age.grid(row=1, column=3, padx=10, pady=5)

        tk.Label(self.tab_student, text="Giới tính:").grid(row=2, column=0, padx=10,
pady=5)
        self.entry_gender = tk.Entry(self.tab_student)
        self.entry_gender.grid(row=2, column=1, padx=10, pady=5)

        tk.Label(self.tab_student, text="Ngành học:").grid(row=2, column=2, padx=10,
pady=5)
        self.entry_major = tk.Entry(self.tab_student)
        self.entry_major.grid(row=2, column=3, padx=10, pady=5)

        # Các nút chức năng
        tk.Button(self.tab_student, text="Thêm sinh viên",
command=self.add_student).grid(row=3, column=0, padx=10, pady=10)
        tk.Button(self.tab_student, text="Cập nhật sinh viên",
command=self.update_student).grid(row=3, column=1, padx=10, pady=10)
        tk.Button(self.tab_student, text="Xóa sinh viên",
command=self.delete_student).grid(row=3, column=2, padx=10, pady=10)

```

```

        tk.Button(self.tab_student, text="Thống kê",
command=self.show_statistics).grid(row=3, column=3, padx=10, pady=10)

        # Sự kiện chọn dòng
        self.tree.bind("<<TreeviewSelect>>", self.on_row_select)

def connect_db(self):
    try:
        self.conn = psycopg2.connect(
            dbname=self.db_name.get(),
            user=self.user.get(),
            password=self.password.get(),
            host=self.host.get(),
            port=self.port.get()
        )
        self.cur = self.conn.cursor()
        messagebox.showinfo("Thành công", "Kết nối đến cơ sở dữ liệu thành
công!")
        self.load_students()
    except Exception as e:
        messagebox.showerror("Lỗi", f"Không thể kết nối: {e}")

def load_data(self):
    if hasattr(self, 'cur'):
        try:
            table_name = self.table_name.get()
            self.cur.execute(f"SELECT * FROM {table_name};")
            rows = self.cur.fetchall()
            self.data_display.delete(1.0, tk.END)

            for row in rows:
                self.data_display.insert(tk.END, f"{row}\n")
        except Exception as e:
            messagebox.showerror("Lỗi", f"Không thể tải dữ liệu: {e}")

def load_students(self):
    if hasattr(self, 'cur'):
        self.cur.execute("SELECT * FROM students")
        rows = self.cur.fetchall()
        for row in self.tree.get_children():
            self.tree.delete(row)
        for row in rows:
            self.tree.insert("", tk.END, values=row)

def show_statistics(self):
    if hasattr(self, 'cur'):
        # Đếm tổng số sinh viên
        self.cur.execute("SELECT COUNT(*) FROM students")

```

```

total_students = self.cur.fetchone()[0]

# Đếm số sinh viên theo giới tính
self.cur.execute("SELECT gender, COUNT(*) FROM students GROUP BY gender")
gender_stats = self.cur.fetchall()

# Đếm số sinh viên theo ngành học
self.cur.execute("SELECT major, COUNT(*) FROM students GROUP BY major")
major_stats = self.cur.fetchall()

stats_message = f"Tổng số sinh viên: {total_students}\n\nGiới tính:\n"
for gender, count in gender_stats:
    stats_message += f"{gender}: {count}\n"

stats_message += "\nNgành học:\n"
for major, count in major_stats:
    stats_message += f"{major}: {count}\n"

messagebox.showinfo("Thống kê", stats_message)

def add_student(self):
    name = self.entry_name.get()
    age = self.entry_age.get()
    gender = self.entry_gender.get()
    major = self.entry_major.get()

    if name and age.isdigit() and gender and major:
        self.cur.execute("SELECT COUNT(*) FROM students WHERE name = %s AND age = %s AND gender = %s AND major = %s",
                        (name, int(age), gender, major))
        if self.cur.fetchone()[0] > 0:
            messagebox.showerror("Lỗi", "Sinh viên với thông tin này đã tồn tại.")
            return

        self.cur.execute("INSERT INTO students (name, age, gender, major) VALUES (%s, %s, %s, %s)",
                        (name, int(age), gender, major))
        self.conn.commit()
        self.load_students()
        self.clear_entries()
    else:
        messagebox.showerror("Lỗi", "Vui lòng điền đầy đủ thông tin một cách chính xác.")

def update_student(self):
    selected = self.tree.selection()
    if selected:

```

```

        student_id = self.tree.item(selected[0])['values'][0]
        name = self.entry_name.get()
        age = self.entry_age.get()
        gender = self.entry_gender.get()
        major = self.entry_major.get()

        if name and age.isdigit() and gender and major:
            self.cur.execute("""UPDATE students
                               SET name = %s, age = %s, gender = %s, major = %s
                               WHERE id = %s""",
                              (name, int(age), gender, major, student_id))
            self.conn.commit()
            self.load_students()
            self.clear_entries()
        else:
            messagebox.showerror("Lỗi", "Vui lòng điền đầy đủ thông tin một cách chính xác.")
        else:
            messagebox.showerror("Lỗi", "Vui lòng chọn một sinh viên để cập nhật.")

def delete_student(self):
    selected = self.tree.selection()
    if selected:
        student_id = self.tree.item(selected[0])['values'][0]
        self.cur.execute("DELETE FROM students WHERE id = %s", (student_id,))
        self.conn.commit()
        self.load_students()
        self.clear_entries()
    else:
        messagebox.showerror("Lỗi", "Vui lòng chọn một sinh viên để xóa.")

def clear_entries(self):
    self.entry_name.delete(0, tk.END)
    self.entry_age.delete(0, tk.END)
    self.entry_gender.delete(0, tk.END)
    self.entry_major.delete(0, tk.END)

def on_row_select(self, event):
    selected = self.tree.selection()
    if selected:
        student = self.tree.item(selected[0])['values']
        self.entry_name.delete(0, tk.END)
        self.entry_name.insert(0, student[1])
        self.entry_age.delete(0, tk.END)
        self.entry_age.insert(0, student[2])
        self.entry_gender.delete(0, tk.END)
        self.entry_gender.insert(0, student[3])
        self.entry_major.delete(0, tk.END)

```

```
self.entry_major.insert(0, student[4])
```

```
root = tk.Tk()  
app = DatabaseApp(root)  
root.mainloop()
```

#### 4. Github

[Bai2 Database](#)