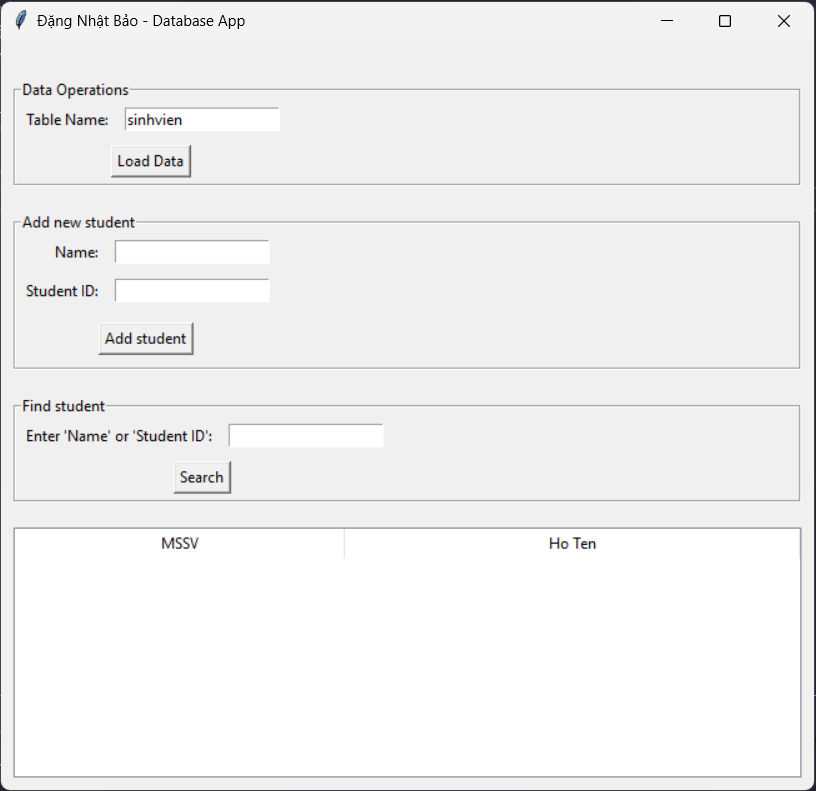
**Cài thư viện và tạo form đăng nhập:**

**A screenshot of a computer screen

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

****

**Nhập tên bảng cần truy xuất và tải dữ liệu lên:**

**A screenshot of a computer

Description automatically generated**

**Thêm mới dữ liệu và tải lại dữ liệu:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Tìm kiếm dữ liệu:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Code:**

**main.py**

import tkinter as tk

from login\_window import LoginWindow

if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    login = LoginWindow(root)

    root.mainloop()

**login\_window.py**

import tkinter as tk

from tkinter import messagebox

import psycopg2

from psycopg2 import sql

from main\_app import open\_main\_app  # Import từ mô-đun khác

class LoginWindow:

    def \_\_init\_\_(*self*, *root*):

*self*.root = *root*

*self*.root.title("Đặng Nhật Bảo - Login")

*self*.root.geometry("300x150")

*self*.root.resizable(False, False)

        # Login credentials

*self*.valid\_username = "admin"

*self*.valid\_password = "123"

        # Database connection info

*self*.db\_name = 'dbtest'

*self*.user = 'postgres'

*self*.password = '123456'

*self*.host = 'localhost'

*self*.port = '5432'

        # Tạo các trường nhập liệu

*self*.username = tk.StringVar()

*self*.password\_var = tk.StringVar()

*self*.create\_widgets()

    def create\_widgets(*self*):

        frame = tk.Frame(*self*.root)

        frame.pack(*pady*=20)

        tk.Label(frame, *text*="Tên tài khoản:").grid(*row*=0, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(frame, *textvariable*=*self*.username).grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

        tk.Label(frame, *text*="Mật khẩu:").grid(*row*=1, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(frame, *textvariable*=*self*.password\_var, *show*="\*").grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

        tk.Button(frame, *text*="Đăng nhập", *command*=*self*.authenticate).grid(*row*=2, *columnspan*=2, *pady*=10)

    def authenticate(*self*):

        user = *self*.username.get()

        pwd = *self*.password\_var.get()

        if user == *self*.valid\_username and pwd == *self*.valid\_password:

            try:

*self*.conn = *self*.connect\_to\_db()

                messagebox.showinfo("Thành công", "Đăng nhập thành công!")

*self*.root.destroy()

                open\_main\_app(*self*.conn)

            except Exception as e:

                messagebox.showerror("Lỗi", f"Không thể kết nối cơ sở dữ liệu: {e}")

        else:

            messagebox.showerror("Lỗi", "Thông tin đăng nhập không hợp lệ!")

    def connect\_to\_db(*self*):

        conn = psycopg2.connect(

*dbname*=*self*.db\_name,

*user*=*self*.user,

*password*=*self*.password,

*host*=*self*.host,

*port*=*self*.port

        )

        return conn

if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    login = LoginWindow(root)

    root.mainloop()

**main\_app.py**

import tkinter as tk

from tkinter import ttk, messagebox

from psycopg2 import sql

class DatabaseApp:

    def \_\_init\_\_(*self*, *root*, *conn*):

*self*.root = *root*

*self*.conn = *conn*

*self*.cur = *self*.conn.cursor()

*self*.root.title("Đặng Nhật Bảo - Database App")

*self*.root.geometry("650x600")

*self*.create\_widgets()

    def create\_widgets(*self*):

        # Main frame

        main\_frame = tk.Frame(*self*.root)

        main\_frame.pack(*padx*=10, *pady*=10, *fill*="both", *expand*=True)

        # Query section

        query\_frame = tk.LabelFrame(*self*.root, *text*="Data Operations")

        query\_frame.pack(*padx*=10, *pady*=10, *fill*="x")

*self*.table\_name = tk.StringVar(*value*='sinhvien')

        tk.Label(query\_frame, *text*="Table Name:").grid(*row*=0, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(query\_frame, *textvariable*=*self*.table\_name).grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

        tk.Button(query\_frame, *text*="Load Data", *command*=*self*.load\_data).grid(*row*=1, *columnspan*=2, *pady*=5)

        # Insert section

*self*.create\_insert\_section()

        # Search section

*self*.create\_search\_section()

        # Data Treeview

*self*.create\_treeview()

    def create\_insert\_section(*self*):

        insert\_frame = tk.LabelFrame(*self*.root, *text*="Add new student")

        insert\_frame.pack(*padx*=10, *pady*=10, *fill*="x")

*self*.column1 = tk.StringVar()

*self*.column2 = tk.StringVar()

        tk.Label(insert\_frame, *text*="Name:").grid(*row*=0, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(insert\_frame, *textvariable*=*self*.column1).grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

        tk.Label(insert\_frame, *text*="Student ID:").grid(*row*=1, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(insert\_frame, *textvariable*=*self*.column2).grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

        tk.Button(insert\_frame, *text*="Add student", *command*=*self*.insert\_data).grid(*row*=2, *columnspan*=2, *pady*=10)

    def create\_search\_section(*self*):

        search\_frame = tk.LabelFrame(*self*.root, *text*="Find student")

        search\_frame.pack(*padx*=10, *pady*=10, *fill*="x")

*self*.search\_value = tk.StringVar()

        tk.Label(search\_frame, *text*="Enter 'Name' or 'Student ID':").grid(*row*=0, *column*=0, *padx*=5, *pady*=5, *sticky*='e')

        tk.Entry(search\_frame, *textvariable*=*self*.search\_value).grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

        tk.Button(search\_frame, *text*="Search", *command*=*self*.search\_data).grid(*row*=1, *columnspan*=2, *pady*=5)

    def create\_treeview(*self*):

*self*.tree = ttk.Treeview(*self*.root, *columns*=("MSSV", "Ho Ten"), *show*="headings")

*self*.tree.heading("MSSV", *text*="MSSV")

*self*.tree.column("MSSV", *width*=100)

*self*.tree.heading("Ho Ten", *text*="Ho Ten")

*self*.tree.column("Ho Ten", *width*=200)

*self*.tree.pack(*padx*=10, *pady*=10, *fill*="both", *expand*=True)

*self*.scrollbar = ttk.Scrollbar(*self*.root, *orient*="vertical", *command*=*self*.tree.yview)

*self*.scrollbar.pack(*side*=tk.RIGHT, *fill*=tk.Y)

*self*.tree.configure(*yscrollcommand*=*self*.scrollbar.set)

    def load\_data(*self*):

        try:

            query = sql.SQL("SELECT \* FROM {}").format(sql.Identifier(*self*.table\_name.get()))

*self*.cur.execute(query)

            rows = *self*.cur.fetchall()

*self*.tree.delete(\**self*.tree.get\_children())

            for row in rows:

*self*.tree.insert("", tk.END, *values*=row)

        except Exception as e:

            messagebox.showerror("Error", f"Error loading data: {e}")

    def insert\_data(*self*):

        try:

            insert\_query = sql.SQL("INSERT INTO {} (hoten, mssv) VALUES (%s, %s)").format(sql.Identifier(*self*.table\_name.get()))

            data\_to\_insert = (*self*.column1.get(), *self*.column2.get())

*self*.cur.execute(insert\_query, data\_to\_insert)

*self*.conn.commit()

            messagebox.showinfo("Success", "Data inserted successfully!")

*self*.load\_data()

        except Exception as e:

*self*.conn.rollback()

            messagebox.showerror("Error", f"Error inserting data: {e}")

    def search\_data(*self*):

        try:

            search\_value = *self*.search\_value.get()

            search\_query = sql.SQL("SELECT \* FROM {} WHERE hoten ILIKE %s OR CAST(mssv AS TEXT) ILIKE %s").format(sql.Identifier(*self*.table\_name.get()))

            search\_pattern = f"%{search\_value}%"

*self*.cur.execute(search\_query, (search\_pattern, search\_pattern))

            rows = *self*.cur.fetchall()

*self*.tree.delete(\**self*.tree.get\_children())

            for row in rows:

*self*.tree.insert("", tk.END, *values*=row)

        except Exception as e:

*self*.conn.rollback()

            messagebox.showerror("Error", f"Error searching data: {e}")

def open\_main\_app(*conn*):

    root = tk.Tk()

    app = DatabaseApp(root, *conn*)

    root.mainloop()