import tkinter as tk

from tkinter import ttk, messagebox

import csv

import os

# Directory and file paths

DATA\_DIR = 'hospital\_data'

PATIENT\_FILE = os.path.join(DATA\_DIR, 'patients.csv')

DOCTOR\_FILE = os.path.join(DATA\_DIR, 'doctors.csv')

NURSE\_FILE = os.path.join(DATA\_DIR, 'nurses.csv')

ROOM\_FILE = os.path.join(DATA\_DIR, 'rooms.csv')

APPOINTMENT\_FILE = os.path.join(DATA\_DIR, 'appointments.csv') # New file

os.makedirs(DATA\_DIR, exist\_ok=True)

# Initialize CSVs with headers

def init\_csv(file\_path, headers, default\_data=None):

if not os.path.exists(file\_path):

with open(file\_path, 'w', newline='') as f:

writer = csv.writer(f)

writer.writerow(headers)

if default\_data:

writer.writerows(default\_data)

init\_csv(PATIENT\_FILE, ['ID', 'Name', 'Age', 'Gender', 'Disease'])

init\_csv(DOCTOR\_FILE, ['ID', 'Name', 'Specialization', 'Phone'])

init\_csv(NURSE\_FILE, ['ID', 'Name', 'Shift', 'Phone'])

init\_csv(ROOM\_FILE, ['Room No', 'Type', 'Status'], [

['101', 'ICU', 'Occupied'],

['102', 'General', 'Available'],

['103', 'Emergency', 'Available'],

['104', 'ICU', 'Available'],

['105', 'General', 'Occupied'],

])

init\_csv(APPOINTMENT\_FILE, ['Appointment ID', 'Patient ID', 'Doctor ID', 'Date', 'Time']) # Initialize appointments

# ---------------- Utility: Center Window ---------------- #

def center\_window(win, width, height):

screen\_w = win.winfo\_screenwidth()

screen\_h = win.winfo\_screenheight()

x = (screen\_w // 2) - (width // 2)

y = (screen\_h // 2) - (height // 2)

win.geometry(f"{width}x{height}+{x}+{y}")

# ---------------- Hospital Management Main App ---------------- #

class HospitalApp:

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

self.root = root

self.root.title("Hospital Management System")

center\_window(self.root, 800, 600)

self.root.configure(bg="#e6f2ff")

title = tk.Label(root, text="🏥 Hospital Management System", font=("Arial", 26, "bold"), bg="#e6f2ff", fg="navy")

title.pack(pady=15)

# Buttons frame

btn\_frame = tk.Frame(root, bg="#e6f2ff")

btn\_frame.pack(pady=20)

options = [

("➕ Add Patient", self.add\_patient),

("➕ Add Doctor", self.add\_doctor),

("➕ Add Nurse", self.add\_nurse),

("📋 View Patients", self.view\_patients),

("📋 View Doctors", self.view\_doctors),

("📋 View Nurses", self.view\_nurses),

("🏠 View Available Rooms", self.view\_rooms),

("📅 Manage Appointments", self.manage\_appointments), # New option

]

for i, (text, command) in enumerate(options):

btn = tk.Button(btn\_frame, text=text, width=25, font=("Arial", 13, "bold"),

bg="#b3d9ff", fg="black", relief="raised", command=command)

btn.grid(row=i // 2, column=i % 2, padx=15, pady=10, sticky="nsew")

def add\_entry(self, fields, file\_path, title):

def submit():

data = [entry.get() for entry in entries]

if any(not val for val in data):

messagebox.showerror("Error", "All fields are required.")

return

with open(file\_path, 'a', newline='') as f:

writer = csv.writer(f)

writer.writerow(data)

messagebox.showinfo("Success", f"{title} added successfully.")

top.destroy()

top = tk.Toplevel(self.root)

top.title(f"Add {title}")

center\_window(top, 400, 300)

entries = []

for idx, field in enumerate(fields):

tk.Label(top, text=field + ":", font=("Arial", 12)).grid(row=idx, column=0, pady=5, padx=10, sticky='e')

e = tk.Entry(top, font=("Arial", 12), width=25)

e.grid(row=idx, column=1, pady=5, padx=10)

entries.append(e)

tk.Button(top, text="Submit", command=submit, bg="#66cc66", fg="white",

font=("Arial", 12, "bold")).grid(row=len(fields), columnspan=2, pady=15)

def view\_data(self, file\_path, title):

top = tk.Toplevel(self.root)

top.title(title)

center\_window(top, 600, 400)

top.configure(bg="#f9f9f9")

frame = tk.Frame(top)

frame.pack(expand=True, fill='both', padx=10, pady=10)

tree\_scroll\_y = tk.Scrollbar(frame, orient="vertical")

tree\_scroll\_y.pack(side="right", fill="y")

tree\_scroll\_x = tk.Scrollbar(frame, orient="horizontal")

tree\_scroll\_x.pack(side="bottom", fill="x")

tree = ttk.Treeview(frame, yscrollcommand=tree\_scroll\_y.set, xscrollcommand=tree\_scroll\_x.set)

tree.pack(expand=True, fill='both')

tree\_scroll\_y.config(command=tree.yview)

tree\_scroll\_x.config(command=tree.xview)

with open(file\_path, newline='') as f:

reader = csv.reader(f)

headers = next(reader)

tree["columns"] = headers

tree["show"] = "headings"

for header in headers:

tree.heading(header, text=header)

tree.column(header, width=150, anchor="center")

for row in reader:

tree.insert("", "end", values=row)

tk.Button(top, text="⬅ Back to Home", bg="orange", fg="black",

font=("Arial", 12, "bold"), command=top.destroy).pack(pady=10)

# ---------------- Appointment Management ---------------- #

def manage\_appointments(self):

top = tk.Toplevel(self.root)

top.title("Appointments")

center\_window(top, 600, 400)

top.configure(bg="#f9f9f9")

frame = tk.Frame(top)

frame.pack(expand=True, fill='both', padx=10, pady=10)

tree\_scroll\_y = tk.Scrollbar(frame, orient="vertical")

tree\_scroll\_y.pack(side="right", fill="y")

tree\_scroll\_x = tk.Scrollbar(frame, orient="horizontal")

tree\_scroll\_x.pack(side="bottom", fill="x")

tree = ttk.Treeview(frame, yscrollcommand=tree\_scroll\_y.set, xscrollcommand=tree\_scroll\_x.set)

tree.pack(expand=True, fill='both')

tree\_scroll\_y.config(command=tree.yview)

tree\_scroll\_x.config(command=tree.xview)

def load\_data():

for i in tree.get\_children():

tree.delete(i)

with open(APPOINTMENT\_FILE, newline='') as f:

reader = csv.reader(f)

headers = next(reader)

tree["columns"] = headers

tree["show"] = "headings"

for header in headers:

tree.heading(header, text=header)

tree.column(header, width=150, anchor="center")

for row in reader:

tree.insert("", "end", values=row)

def add\_appointment():

def submit():

data = [entry.get() for entry in entries]

if any(not val for val in data):

messagebox.showerror("Error", "All fields are required.")

return

with open(APPOINTMENT\_FILE, 'a', newline='') as f:

writer = csv.writer(f)

writer.writerow(data)

messagebox.showinfo("Success", "Appointment added successfully.")

load\_data()

top\_add.destroy()

top\_add = tk.Toplevel(top)

top\_add.title("Add Appointment")

center\_window(top\_add, 400, 350)

fields = ['Appointment ID', 'Patient ID', 'Doctor ID', 'Date', 'Time']

entries = []

for idx, field in enumerate(fields):

tk.Label(top\_add, text=field+":", font=("Arial", 12)).grid(row=idx, column=0, padx=10, pady=5, sticky="e")

e = tk.Entry(top\_add, font=("Arial", 12), width=25)

e.grid(row=idx, column=1, padx=10, pady=5)

entries.append(e)

tk.Button(top\_add, text="Submit", command=submit, bg="#66cc66", fg="white",

font=("Arial", 12, "bold")).grid(row=len(fields), columnspan=2, pady=15)

def print\_appointment():

selected = tree.focus()

if not selected:

messagebox.showerror("Error", "Please select an appointment to print.")

return

data = tree.item(selected)['values']

info = f"""

Appointment Details

Appointment ID: {data[0]}

Patient ID: {data[1]}

Doctor ID: {data[2]}

Date: {data[3]}

Time: {data[4]}

"""

print\_win = tk.Toplevel(top)

print\_win.title("Print Appointment")

center\_window(print\_win, 400, 300)

tk.Label(print\_win, text="📄 Appointment Details", font=("Arial", 14, "bold")).pack(pady=10)

text = tk.Text(print\_win, font=("Arial", 12), width=40, height=10)

text.pack(padx=10, pady=10)

text.insert("1.0", info)

text.config(state="disabled")

tk.Button(print\_win, text="Close", command=print\_win.destroy,

bg="orange", fg="black", font=("Arial", 12, "bold")).pack(pady=10)

# Buttons

btn\_frame = tk.Frame(top, bg="#f9f9f9")

btn\_frame.pack(pady=10)

tk.Button(btn\_frame, text="➕ Add Appointment", command=add\_appointment,

bg="#66cc66", fg="white", font=("Arial", 12, "bold")).grid(row=0, column=0, padx=10)

tk.Button(btn\_frame, text="🖨 Print Appointment", command=print\_appointment,

bg="#3399ff", fg="white", font=("Arial", 12, "bold")).grid(row=0, column=1, padx=10)

tk.Button(btn\_frame, text="⬅ Back", command=top.destroy,

bg="orange", fg="black", font=("Arial", 12, "bold")).grid(row=0, column=2, padx=10)

load\_data()

# ---------------- Button functions ---------------- #

def add\_patient(self): self.add\_entry(['ID', 'Name', 'Age', 'Gender', 'Disease'], PATIENT\_FILE, "Patient")

def add\_doctor(self): self.add\_entry(['ID', 'Name', 'Specialization', 'Phone'], DOCTOR\_FILE, "Doctor")

def add\_nurse(self): self.add\_entry(['ID', 'Name', 'Shift', 'Phone'], NURSE\_FILE, "Nurse")

def view\_patients(self): self.view\_data(PATIENT\_FILE, "Patients List")

def view\_doctors(self): self.view\_data(DOCTOR\_FILE, "Doctors List")

def view\_nurses(self): self.view\_data(NURSE\_FILE, "Nurses List")

def view\_rooms(self): self.view\_data(ROOM\_FILE, "Available Rooms")

# ---------------- Login Page ---------------- #

class LoginApp:

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

self.root = root

self.root.title("Login - Hospital Management System")

center\_window(self.root, 400, 250)

self.root.configure(bg="#f2f2f2")

tk.Label(root, text="🔐 Login", font=("Arial", 22, "bold"), bg="#f2f2f2", fg="black").pack(pady=10)

login\_frame = tk.Frame(root, bg="#f2f2f2")

login\_frame.pack(pady=10)

tk.Label(login\_frame, text="Username:", font=("Arial", 12), bg="#f2f2f2").grid(row=0, column=0, padx=10, pady=5, sticky="e")

self.username\_entry = tk.Entry(login\_frame, font=("Arial", 12), width=20)

self.username\_entry.grid(row=0, column=1, pady=5)

tk.Label(login\_frame, text="Password:", font=("Arial", 12), bg="#f2f2f2").grid(row=1, column=0, padx=10, pady=5, sticky="e")

self.password\_entry = tk.Entry(login\_frame, show="\*", font=("Arial", 12), width=20)

self.password\_entry.grid(row=1, column=1, pady=5)

tk.Button(root, text="Login", font=("Arial", 13, "bold"), bg="#66ccff", fg="black",

width=12, command=self.check\_login).pack(pady=15)

def check\_login(self):

username = self.username\_entry.get()

password = self.password\_entry.get()

if username == "admin" and password == "admin123":

self.root.destroy()

main\_root = tk.Tk()

HospitalApp(main\_root)

main\_root.mainloop()

else:

messagebox.showerror("Error", "Invalid Username or Password")

# ---------------- Run the Program ---------------- #

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

root = tk.Tk()

app = LoginApp(root)

root.mainloop()