import tkinter as tk

from tkinter import ttk, messagebox

import db\_handler

# Colors

BG\_COLOR = "#E0FFE0" # Mint Green

HEADER\_COLOR = "#D8BFD8" # Light Purple

BUTTON\_COLOR = "#98FF98" # Light Mint Green

# Utility Functions

def clear\_window(root):

for widget in root.winfo\_children():

widget.destroy()

# Login Screen

def login\_screen(root):

root.title("Hostel Management Login")

root.geometry("400x300")

root.configure(bg=BG\_COLOR)

tk.Label(root, text="Hostel Warden Login", font=("Helvetica", 18), bg=HEADER\_COLOR).pack(pady=20)

tk.Label(root, text="Username:", bg=BG\_COLOR).pack(pady=5)

username\_entry = tk.Entry(root)

username\_entry.pack()

tk.Label(root, text="Password:", bg=BG\_COLOR).pack(pady=5)

password\_entry = tk.Entry(root, show="\*")

password\_entry.pack()

def attempt\_login():

if username\_entry.get() == "admin" and password\_entry.get() == "admin123":

dashboard(root)

else:

messagebox.showerror("Login Failed", "Incorrect username or password")

tk.Button(root, text="Login", command=attempt\_login, bg=BUTTON\_COLOR).pack(pady=20)

# Dashboard

def dashboard(root):

clear\_window(root)

root.title("Hostel Management Dashboard")

root.geometry("500x400")

root.configure(bg=BG\_COLOR)

tk.Label(root, text="Hostel Dashboard", font=("Helvetica", 20), bg=HEADER\_COLOR).pack(pady=20)

tk.Button(root, text="Manage Students", width=20, height=2, bg=BUTTON\_COLOR, command=lambda: manage\_students(root)).pack(pady=10)

tk.Button(root, text="Manage Rooms", width=20, height=2, bg=BUTTON\_COLOR, command=lambda: manage\_rooms(root)).pack(pady=10)

tk.Button(root, text="Manage Payments", width=20, height=2, bg=BUTTON\_COLOR, command=lambda: manage\_payments(root)).pack(pady=10)

# Manage Students

def manage\_students(root):

clear\_window(root)

tk.Label(root, text="Manage Students", font=("Helvetica", 18), bg=HEADER\_COLOR).pack(pady=10)

frame = tk.Frame(root, bg=BG\_COLOR)

frame.pack()

id\_entry = tk.Entry(frame)

name\_entry = tk.Entry(frame)

age\_entry = tk.Entry(frame)

room\_entry = tk.Entry(frame)

labels = ["ID:", "Name:", "Age:", "Room:"]

entries = [id\_entry, name\_entry, age\_entry, room\_entry]

for idx, text in enumerate(labels):

tk.Label(frame, text=text, bg=BG\_COLOR).grid(row=idx, column=0, sticky="w", pady=2)

entries[idx].grid(row=idx, column=1, pady=2)

def manage\_students(root):

clear\_window(root)

tk.Label(root, text="Manage Students", font=("Helvetica", 18), bg=HEADER\_COLOR).pack(pady=10)

frame = tk.Frame(root, bg=BG\_COLOR)

frame.pack()

id\_entry = tk.Entry(frame)

name\_entry = tk.Entry(frame)

age\_entry = tk.Entry(frame)

labels = ["ID:", "Name:", "Age:"]

entries = [id\_entry, name\_entry, age\_entry]

for idx, text in enumerate(labels):

tk.Label(frame, text=text, bg=BG\_COLOR).grid(row=idx, column=0, sticky="w", pady=2)

entries[idx].grid(row=idx, column=1, pady=2)

def add\_student():

try:

db\_handler.add\_student(id\_entry.get(), name\_entry.get(), age\_entry.get())

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

refresh\_tree()

except Exception as e:

messagebox.showerror("Error", str(e))

...

def delete\_student():

selected = tree.focus()

if selected:

student\_id = tree.item(selected)['values'][0]

db\_handler.delete\_student(student\_id)

refresh\_tree()

def search\_student():

result = db\_handler.search\_student(search\_entry.get())

refresh\_tree(result)

button\_frame = tk.Frame(root, bg=BG\_COLOR)

button\_frame.pack(pady=10)

tk.Button(button\_frame, text="Add", command=add\_student, bg=BUTTON\_COLOR).grid(row=0, column=0, padx=5)

tk.Button(button\_frame, text="Delete", command=delete\_student, bg=BUTTON\_COLOR).grid(row=0, column=1, padx=5)

tk.Button(button\_frame, text="Back", command=lambda: dashboard(root), bg=BUTTON\_COLOR).grid(row=0, column=2, padx=5)

search\_entry = tk.Entry(root)

search\_entry.pack()

tk.Button(root, text="Search", command=search\_student, bg=BUTTON\_COLOR).pack()

tree\_frame = tk.Frame(root)

tree\_frame.pack(pady=10)

tree\_scroll = tk.Scrollbar(tree\_frame)

tree\_scroll.pack(side=tk.RIGHT, fill=tk.Y)

tree = ttk.Treeview(tree\_frame, columns=("ID", "Name", "Age", "Room"), show="headings", yscrollcommand=tree\_scroll.set)

tree.heading("ID", text="ID")

tree.heading("Name", text="Name")

tree.heading("Age", text="Age")

tree.heading("Room", text="Room")

tree.pack()

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

def refresh\_tree(data=None):

for item in tree.get\_children():

tree.delete(item)

records = data if data else db\_handler.view\_students()

for row in records:

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

refresh\_tree()

# Manage Rooms

def manage\_rooms(root):

clear\_window(root)

tk.Label(root, text="Manage Rooms", font=("Helvetica", 18), bg=HEADER\_COLOR).pack(pady=10)

frame = tk.Frame(root, bg=BG\_COLOR)

frame.pack()

room\_no\_entry = tk.Entry(frame)

capacity\_entry = tk.Entry(frame)

labels = ["Room No:", "Capacity:"]

entries = [room\_no\_entry, capacity\_entry]

for idx, text in enumerate(labels):

tk.Label(frame, text=text, bg=BG\_COLOR).grid(row=idx, column=0, sticky="w", pady=2)

entries[idx].grid(row=idx, column=1, pady=2)

def add\_room():

try:

db\_handler.add\_room(room\_no\_entry.get(), capacity\_entry.get())

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

refresh\_tree()

except Exception as e:

messagebox.showerror("Error", str(e))

def delete\_room():

selected = tree.focus()

if selected:

room\_no = tree.item(selected)['values'][0]

db\_handler.delete\_room(room\_no)

refresh\_tree()

def search\_room():

result = db\_handler.search\_room(search\_entry.get())

refresh\_tree(result)

button\_frame = tk.Frame(root, bg=BG\_COLOR)

button\_frame.pack(pady=10)

tk.Button(button\_frame, text="Add", command=add\_room, bg=BUTTON\_COLOR).grid(row=0, column=0, padx=5)

tk.Button(button\_frame, text="Delete", command=delete\_room, bg=BUTTON\_COLOR).grid(row=0, column=1, padx=5)

tk.Button(button\_frame, text="Back", command=lambda: dashboard(root), bg=BUTTON\_COLOR).grid(row=0, column=2, padx=5)

search\_entry = tk.Entry(root)

search\_entry.pack()

tk.Button(root, text="Search", command=search\_room, bg=BUTTON\_COLOR).pack()

tree\_frame = tk.Frame(root)

tree\_frame.pack(pady=10)

tree\_scroll = tk.Scrollbar(tree\_frame)

tree\_scroll.pack(side=tk.RIGHT, fill=tk.Y)

tree = ttk.Treeview(tree\_frame, columns=("Room No", "Capacity"), show="headings", yscrollcommand=tree\_scroll.set)

tree.heading("Room No", text="Room No")

tree.heading("Capacity", text="Capacity")

tree.pack()

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

def refresh\_tree(data=None):

for item in tree.get\_children():

tree.delete(item)

records = data if data else db\_handler.view\_rooms()

for row in records:

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

refresh\_tree()

# Manage Payments

def manage\_payments(root):

clear\_window(root)

tk.Label(root, text="Manage Payments", font=("Helvetica", 18), bg=HEADER\_COLOR).pack(pady=10)

frame = tk.Frame(root, bg=BG\_COLOR)

frame.pack()

student\_id\_entry = tk.Entry(frame)

amount\_entry = tk.Entry(frame)

date\_entry = tk.Entry(frame)

labels = ["Student ID:", "Amount:", "Date:"]

entries = [student\_id\_entry, amount\_entry, date\_entry]

for idx, text in enumerate(labels):

tk.Label(frame, text=text, bg=BG\_COLOR).grid(row=idx, column=0, sticky="w", pady=2)

entries[idx].grid(row=idx, column=1, pady=2)

def add\_payment():

try:

db\_handler.add\_payment(student\_id\_entry.get(), amount\_entry.get(), date\_entry.get())

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

refresh\_tree()

except Exception as e:

messagebox.showerror("Error", str(e))

def delete\_payment():

selected = tree.focus()

if selected:

payment\_id = tree.item(selected)['values'][0]

db\_handler.delete\_payment(payment\_id)

refresh\_tree()

def search\_payment():

result = db\_handler.search\_payment(search\_entry.get())

refresh\_tree(result)

button\_frame = tk.Frame(root, bg=BG\_COLOR)

button\_frame.pack(pady=10)

tk.Button(button\_frame, text="Add", command=add\_payment, bg=BUTTON\_COLOR).grid(row=0, column=0, padx=5)

tk.Button(button\_frame, text="Delete", command=delete\_payment, bg=BUTTON\_COLOR).grid(row=0, column=1, padx=5)

tk.Button(button\_frame, text="Back", command=lambda: dashboard(root), bg=BUTTON\_COLOR).grid(row=0, column=2, padx=5)

search\_entry = tk.Entry(root)

search\_entry.pack()

tk.Button(root, text="Search", command=search\_payment, bg=BUTTON\_COLOR).pack()

tree\_frame = tk.Frame(root)

tree\_frame.pack(pady=10)

tree\_scroll = tk.Scrollbar(tree\_frame)

tree\_scroll.pack(side=tk.RIGHT, fill=tk.Y)

tree = ttk.Treeview(tree\_frame, columns=("Payment ID", "Student ID", "Amount", "Date"), show="headings", yscrollcommand=tree\_scroll.set)

tree.heading("Payment ID", text="Payment ID")

tree.heading("Student ID", text="Student ID")

tree.heading("Amount", text="Amount")

tree.heading("Date", text="Date")

tree.pack()

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

def refresh\_tree(data=None):

for item in tree.get\_children():

tree.delete(item)

records = data if data else db\_handler.view\_payments()

for row in records:

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

refresh\_tree()

# Main Program

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

root = tk.Tk()

db\_handler.create\_tables()

login\_screen(root)

root.mainloop()