def display\_sales\_details(self, employee\_id):

if employee\_id == self.\_employeeID:

if len(self.\_sales) == 0:

print("This employee has no sales.")

else:

print("Sales details for employee ID {}:".format(self.\_employeeID))

for sale in self.\_sales:

print("Car ID: {}, Sale Price: {}".format(sale.\_car, sale.\_price))

else:

print("Employee with ID {} not found".format(employee\_id))

import tkinter as tk  
from tkinter import ttk  
import pickle  
  
class EmployeeManagementApp(tk.Tk):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
  
 self.geometry("1500x500")  
 self.title('Employee Management App')  
 self.resizable(0, 0)  
  
 # Create buttons for adding, modifying, and deleting employees  
 btn\_add\_employee = tk.Button(self, text="Add Employee", command=self.open\_add\_employee\_window)  
 btn\_add\_employee.pack(padx=10, pady=10)  
  
 btn\_modify\_employee = tk.Button(self, text="Modify Employee", command=self.open\_modify\_employee\_window)  
 btn\_modify\_employee.pack(padx=10, pady=10)  
  
 btn\_delete\_employee = tk.Button(self, text="Delete Employee", command=self.open\_delete\_employee\_window)  
 btn\_delete\_employee.pack(padx=10, pady=10)  
  
 # Create a list to store employee details  
 self.employees = []  
  
 # Load the employee data from the pickle file  
 try:  
 with open('employee\_data.pkl', 'rb') as f:  
 self.employees = pickle.load(f)  
 except FileNotFoundError:  
 # Create the pickle file if it doesn't exist  
 with open('employee\_data.pkl', 'wb') as f:  
 pickle.dump(self.employees, f)  
  
 # Create a frame to display employee data  
 self.employee\_data\_frame = ttk.Treeview(self, columns=('ID', 'First Name', 'Last Name', 'Gender', 'Date of Birth', 'Job Title', 'Department', 'Salary', 'Passport Number'), show='headings')  
 self.employee\_data\_frame.pack(side=tk.LEFT)  
  
 self.employee\_data\_frame.heading('ID', text='ID')  
 self.employee\_data\_frame.heading('First Name', text='First Name')  
 self.employee\_data\_frame.heading('Last Name', text='Last Name')  
 self.employee\_data\_frame.heading('Gender', text='Gender')  
 self.employee\_data\_frame.heading('Date of Birth', text='Date of Birth')  
 self.employee\_data\_frame.heading('Job Title', text='Job Title')  
 self.employee\_data\_frame.heading('Department', text='Department')  
 self.employee\_data\_frame.heading('Salary', text='Salary')  
 self.employee\_data\_frame.heading('Passport Number', text='Passport Number')  
  
 # Update the employee data table  
 self.update\_employee\_data\_table()  
  
 def update\_employee\_data\_table(self):  
 # Clear the existing employee data table  
 for row in self.employee\_data\_frame.get\_children():  
 self.employee\_data\_frame.delete(row)  
  
 # Load employee data from pickle file  
 try:  
 with open('employee\_data.pkl', 'rb') as f:  
 self.employees = pickle.load(f)  
 except FileNotFoundError:  
 self.employees = []  
  
 # Insert employee data into the table  
 for employee in self.employees:  
 self.employee\_data\_frame.insert('', tk.END, values=(  
 employee['id'],  
 employee['first\_name'],  
 employee['last\_name'],  
 employee['gender'],  
 employee['date\_of\_birth'],  
 employee['job\_title'],  
 employee['department'],  
 employee['salary'],  
 employee['passport\_number']  
 ))  
  
 def open\_add\_employee\_window(self):  
 # Create a new window for adding a new employee  
 add\_employee\_window = tk.Toplevel(self)  
 add\_employee\_window.title('Add Employee')  
  
 # Create labels and entry widgets for employee details  
 lbl\_employee\_id = tk.Label(add\_employee\_window, text="Employee ID:")  
 lbl\_employee\_id.pack()  
  
 entry\_employee\_id = tk.Entry(add\_employee\_window)  
 entry\_employee\_id.pack()  
  
 lbl\_first\_name = tk.Label(add\_employee\_window, text="First Name:")  
 lbl\_first\_name.pack()  
  
 entry\_first\_name = tk.Entry(add\_employee\_window)  
 entry\_first\_name.pack()  
  
 lbl\_last\_name = tk.Label(add\_employee\_window, text="Last Name:")  
 lbl\_last\_name.pack()  
  
 entry\_last\_name = tk.Entry(add\_employee\_window)  
 entry\_last\_name.pack()  
  
 lbl\_gender = tk.Label(add\_employee\_window, text="Gender:")  
 lbl\_gender.pack()  
  
 entry\_gender = tk.Entry(add\_employee\_window)  
 entry\_gender.pack()  
  
 lbl\_date\_of\_birth = tk.Label(add\_employee\_window, text="Date of Birth:")  
 lbl\_date\_of\_birth.pack()  
  
 entry\_date\_of\_birth = tk.Entry(add\_employee\_window)  
 entry\_date\_of\_birth.pack()  
  
 lbl\_job\_title = tk.Label(add\_employee\_window, text="Job Title:")  
 lbl\_job\_title.pack()  
  
 entry\_job\_title = tk.Entry(add\_employee\_window)  
 entry\_job\_title.pack()  
  
 lbl\_department = tk.Label(add\_employee\_window, text="Department:")  
 lbl\_department.pack()  
  
 entry\_department = tk.Entry(add\_employee\_window)  
 entry\_department.pack()  
  
 lbl\_salary = tk.Label(add\_employee\_window, text="Salary:")  
 lbl\_salary.pack()  
  
 entry\_salary = tk.Entry(add\_employee\_window)  
 entry\_salary.pack()  
  
 lbl\_passport\_number = tk.Label(add\_employee\_window, text="Passport Number:")  
 lbl\_passport\_number.pack()  
  
 entry\_passport\_number = tk.Entry(add\_employee\_window)  
 entry\_passport\_number.pack()  
  
 # Create a button to add the employee  
 btn\_add = tk.Button(add\_employee\_window, text="Add", command=lambda: self.add\_employee(  
 entry\_employee\_id.get(),  
 entry\_first\_name.get(),  
 entry\_last\_name.get(),  
 entry\_gender.get(),  
 entry\_date\_of\_birth.get(),  
 entry\_job\_title.get(),  
 entry\_department.get(),  
 entry\_salary.get(),  
 entry\_passport\_number.get(),  
 add\_employee\_window  
 ))  
 btn\_add.pack(padx=10, pady=10)  
  
  
 def add\_employee(self, employee\_id, first\_name, last\_name, gender, date\_of\_birth, job\_title, department, salary,  
 passport\_number, add\_employee\_window):  
 # Convert the employee ID and salary to integers  
 employee\_id = int(employee\_id)  
 salary = int(salary)  
  
 # Create a dictionary to store the employee details  
 employee = {"id": employee\_id,  
 "first\_name": first\_name,  
 "last\_name": last\_name,  
 "gender": gender,  
 "date\_of\_birth": date\_of\_birth,  
 "job\_title": job\_title,  
 "department": department,  
 "salary": salary,  
 "passport\_number": passport\_number}  
  
 # Add the employee to the list  
 self.employees.append(employee)  
  
 # Print a message to confirm that the employee was added  
 print("Employee added successfully:", employee)  
  
 # Save the updated employee data to the pickle file  
 with open('employee\_data.pkl', 'wb') as f:  
 pickle.dump(self.employees, f)  
  
 # Close the add employee window  
 add\_employee\_window.destroy()  
  
 def open\_modify\_employee\_window(self):  
 # Check if an item has been selected in the employee data frame  
 if not self.employee\_data\_frame.selection():  
 print("No item selected.")  
 return  
  
 # Get the ID of the selected employee from the employee data frame  
 selected\_employee\_id = self.employee\_data\_frame.item(self.employee\_data\_frame.selection())['values'][0]  
  
 # Find the selected employee in the list of employees  
 for employee in self.employees:  
 if employee["id"] == selected\_employee\_id:  
 # Create a new window for modifying the employee  
 modify\_employee\_window = tk.Toplevel(self)  
 modify\_employee\_window.title('Modify Employee')  
  
 # Create a label and entry widget for the employee ID  
 lbl\_employee\_id = tk.Label(modify\_employee\_window, text="Employee ID:")  
 lbl\_employee\_id.pack()  
  
 entry\_employee\_id = tk.Entry(modify\_employee\_window)  
 entry\_employee\_id.insert(0, employee['id'])  
 entry\_employee\_id.pack()  
  
 # Create labels and entry widgets for employee details  
 lbl\_first\_name = tk.Label(modify\_employee\_window, text="First Name:")  
 lbl\_first\_name.pack()  
  
 entry\_first\_name = tk.Entry(modify\_employee\_window)  
 entry\_first\_name.insert(0, employee['first\_name'])  
 entry\_first\_name.pack()  
  
 lbl\_last\_name = tk.Label(modify\_employee\_window, text="Last Name:")  
 lbl\_last\_name.pack()  
  
 entry\_last\_name = tk.Entry(modify\_employee\_window)  
 entry\_last\_name.insert(0, employee['last\_name'])  
 entry\_last\_name.pack()  
  
 lbl\_gender = tk.Label(modify\_employee\_window, text="Gender:")  
 lbl\_gender.pack()  
  
 entry\_gender = tk.Entry(modify\_employee\_window)  
 entry\_gender.insert(0, employee['gender'])  
 entry\_gender.pack()  
  
 lbl\_date\_of\_birth = tk.Label(modify\_employee\_window, text="Date of Birth:")  
 lbl\_date\_of\_birth.pack()  
  
 entry\_date\_of\_birth = tk.Entry(modify\_employee\_window)  
 entry\_date\_of\_birth.insert(0, employee['date\_of\_birth'])  
 entry\_date\_of\_birth.pack()  
  
 lbl\_job\_title = tk.Label(modify\_employee\_window, text="Job Title:")  
 lbl\_job\_title.pack()  
  
 entry\_job\_title = tk.Entry(modify\_employee\_window)  
 entry\_job\_title.insert(0, employee['job\_title'])  
 entry\_job\_title.pack()  
  
 lbl\_department = tk.Label(modify\_employee\_window, text="Department:")  
 lbl\_department.pack()  
  
 entry\_department = tk.Entry(modify\_employee\_window)  
 entry\_department.insert(0, employee['department'])  
 entry\_department.pack()  
  
 lbl\_salary = tk.Label(modify\_employee\_window, text="Salary:")  
 lbl\_salary.pack()  
  
 entry\_salary = tk.Entry(modify\_employee\_window)  
 entry\_salary.insert(0, employee['salary'])  
 entry\_salary.pack()  
  
 lbl\_passport\_number = tk.Label(modify\_employee\_window, text="Passport Number:")  
 lbl\_passport\_number.pack()  
  
 entry\_passport\_number = tk.Entry(modify\_employee\_window)  
 entry\_passport\_number.insert(0, employee['passport\_number'])  
 entry\_passport\_number.pack()  
  
 # Create a button to modify the employee details  
 btn\_modify\_employee = tk.Button(modify\_employee\_window, text="Modify Employee",  
 command=lambda: self.modify\_employee(  
 entry\_employee\_id.get(),  
 entry\_first\_name.get(),  
 entry\_last\_name.get(),  
 entry\_gender.get(),  
 entry\_date\_of\_birth.get(),  
 entry\_job\_title.get(),  
 entry\_department.get(),  
 entry\_salary.get(),  
 entry\_passport\_number.get(),  
 modify\_employee\_window  
 ))  
 btn\_modify\_employee.pack(padx=10, pady=10)  
 break  
  
 def modify\_employee(self, employee\_id, first\_name, last\_name, gender, date\_of\_birth, job\_title, department, salary,  
 passport\_number, window):  
 # Convert the employee ID to an integer  
 employee\_id = int(employee\_id)  
  
 # Identify the employee to modify by their ID  
 for employee in self.employees:  
 if employee["id"] == employee\_id:  
 # Modify the employee details  
 employee["first\_name"] = first\_name  
 employee["last\_name"] = last\_name  
 employee["gender"] = gender  
 employee["date\_of\_birth"] = date\_of\_birth  
 employee["job\_title"] = job\_title  
 employee["department"] = department  
 employee["salary"] = salary  
 employee["passport\_number"] = passport\_number  
  
 # Print a message to confirm that the employee was modified  
 print("Employee modified successfully:", employee)  
  
 # Save the updated employee data to the pickle file  
 with open('employee\_data.pkl', 'wb') as f:  
 pickle.dump(self.employees, f)  
  
 # Destroy the modify employee window  
 window.destroy()  
  
 # Refresh the employee data frame to display the updated employee data  
 self.update\_employee\_data\_table()  
 return  
  
 # If the employee was not found, print an error message  
 print("Employee not found.")  
  
 def open\_delete\_employee\_window(self):  
 # Create a new window for deleting an existing employee  
 delete\_employee\_window = tk.Toplevel(self)  
 delete\_employee\_window.title('Delete Employee')  
  
 # Create a label and entry widget for the employee ID  
 lbl\_employee\_id = tk.Label(delete\_employee\_window, text="Employee ID:")  
 lbl\_employee\_id.pack()  
  
 entry\_employee\_id = tk.Entry(delete\_employee\_window)  
 entry\_employee\_id.pack()  
  
 # Create a button to confirm the deletion of the employee  
 btn\_confirm\_delete = tk.Button(delete\_employee\_window, text="Delete Employee",  
 command=lambda: self.delete\_employee(entry\_employee\_id.get()))  
 btn\_confirm\_delete.pack(padx=10, pady=10)  
  
 def delete\_employee(self, employee\_id):  
 # Convert the employee ID to an integer  
 employee\_id = int(employee\_id)  
  
 # Identify the employee to delete by their ID  
 for employee in self.employees:  
 if employee["id"] == employee\_id:  
 # Delete the employee from the list  
 self.employees.remove(employee)  
  
 # Print a message to confirm that the employee was deleted  
 print("Employee deleted successfully:", employee)  
  
 # Save the updated employee data to the pickle file  
 with open('employee\_data.pkl', 'wb') as f:  
 pickle.dump(self.employees, f)  
  
 return  
  
 # If the employee was not found, print an error message  
 print("Employee not found.")  
  
app = EmployeeManagementApp()  
app.mainloop()