CODE DOCUMENTATION:

main.py

import tkinter as tk  
from tkinter import Label, Entry, Button, Listbox, simpledialog, messagebox, END, Frame, Canvas, Scrollbar  
import sqlite3  
from PIL import Image, ImageTk  
import os  
  
NETFLIX\_BLACK = "#141414"  
NETFLIX\_RED = "#E50914"  
NETFLIX\_WHITE = "#FFFFFF"  
NETFLIX\_GRAY = "#757575"  
  
# Connect to the SQLite database  
conn = sqlite3.connect('icecream.db')  
cursor = conn.cursor()  
  
# Create a table to store flavors if it doesn't exist  
cursor.execute('''CREATE TABLE IF NOT EXISTS flavors  
 (name TEXT, seasonal INTEGER, ingredients TEXT, allergens TEXT)''')  
conn.commit()  
  
# Create a table to store cart items if it doesn't exist  
cursor.execute('''CREATE TABLE IF NOT EXISTS cart  
 (name TEXT, allergens TEXT)''')  
conn.commit()  
  
# Create a temporary table for suggested flavors  
cursor.execute('''CREATE TEMPORARY TABLE suggested\_flavors  
 (name TEXT, seasonal INTEGER, ingredients TEXT, allergens TEXT)''')  
conn.commit()  
  
# Clear the cart table on application start  
cursor.execute("DELETE FROM cart")  
conn.commit()  
  
# Clear the flavors and suggested\_flavors tables on application start  
cursor.execute("DELETE FROM flavors")  
conn.commit()  
cursor.execute("DELETE FROM suggested\_flavors")  
conn.commit()  
  
# Insert values for flavors  
flavors\_to\_insert = [  
 ("Vanilla", 0, "Milk, Sugar, Vanilla Extract", "Milk"),  
 ("Chocolate", 0, "Milk, Sugar, Chocolate", "Milk"),  
 ("Strawberry", 1, "Milk, Sugar, Strawberry", "Milk"),  
 ("Butterscotch", 0, "Milk, Sugar, Butterscotch Flavoring", "Milk"),  
 ("Cookies and Cream", 0, "Milk, Sugar, Chocolate Cookies", "Milk, Wheat"),  
 ("Pista", 0, "Milk, Sugar, Pistachio Nuts", "Milk, Nuts"),  
 ("Mango", 1, "Milk, Sugar, Mango Pulp", "Milk")  
]  
  
cursor.executemany("INSERT INTO flavors (name, seasonal, ingredients, allergens) VALUES (?, ?, ?, ?)", flavors\_to\_insert)  
conn.commit()  
print("Inserted default flavors into the flavors table")  
  
class IceCreamParlorApp:  
 def \_\_init\_\_(self, root):  
 self.root = root  
 self.root.title("Ice Cream Parlor")  
 self.root.geometry("800x600")  
 self.root.configure(bg=NETFLIX\_BLACK) # Setting background color  
  
 self.create\_main\_window()  
  
 def create\_main\_window(self):  
 self.clear\_window()  
  
 self.home\_label = Label(self.root, text="SHARKS ICE CREAM PARLOUR", font=("Arial", 24), fg=NETFLIX\_WHITE, bg=NETFLIX\_BLACK)  
 self.home\_label.pack(pady=20)  
  
 self.flavors\_button = Button(self.root, text="View Flavors", command=self.open\_flavors\_window, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.flavors\_button.pack(pady=10)  
  
 self.cart\_button = Button(self.root, text="View Cart", command=self.open\_cart\_window, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.cart\_button.pack(pady=10)  
  
 self.suggest\_button = Button(self.root, text="Suggest a Flavor", command=self.open\_suggest\_window, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.suggest\_button.pack(pady=10)  
  
 def clear\_window(self):  
 for widget in self.root.winfo\_children():  
 widget.destroy()  
  
 def open\_flavors\_window(self):  
 self.clear\_window()  
  
 self.frame = Frame(self.root, bg=NETFLIX\_BLACK)  
 self.frame.pack(expand=True, fill="both")  
 self.frame.columnconfigure(0, weight=1)  
  
 self.scrollbar = Scrollbar(self.frame, orient="vertical")  
 self.canvas = Canvas(self.frame, bg=NETFLIX\_BLACK, yscrollcommand=self.scrollbar.set)  
 self.scrollbar.config(command=self.canvas.yview)  
  
 self.scrollbar.pack(side="right", fill="y")  
 self.canvas.pack(side="left", fill="both", expand=True)  
  
 self.inner\_frame = Frame(self.canvas, bg=NETFLIX\_BLACK)  
 self.canvas.create\_window((0, 0), window=self.inner\_frame, anchor="nw")  
  
 self.load\_flavors()  
  
 self.search\_label = Label(self.inner\_frame, text="Search Flavors:", font=("Arial", 12), fg=NETFLIX\_WHITE,  
 bg=NETFLIX\_BLACK)  
 self.search\_label.grid(row=0, column=0, pady=(10, 5), sticky="w")  
  
 self.search\_entry = Entry(self.inner\_frame, width=30)  
 self.search\_entry.grid(row=0, column=1, pady=(10, 5), padx=(5, 0))  
 # Bind the <Return> event to call the search\_flavors method  
 self.search\_entry.bind('<Return>', lambda event: self.search\_flavors())  
  
 self.search\_button = Button(self.inner\_frame, text="Search", command=self.search\_flavors, bg=NETFLIX\_RED,  
 fg=NETFLIX\_WHITE, font=("Arial", 12))  
 self.search\_button.grid(row=0, column=2, pady=(10, 5), padx=(5, 0))  
  
 self.back\_button = Button(self.inner\_frame, text="Back", command=self.create\_main\_window, bg=NETFLIX\_RED,  
 fg=NETFLIX\_WHITE, font=("Arial", 12))  
 self.back\_button.grid(row=0, column=3, pady=(10, 5), padx=(5, 0))  
  
 self.canvas.bind("<Configure>", self.on\_canvas\_configure)  
  
 def on\_canvas\_configure(self, event):  
 self.canvas.configure(scrollregion=self.canvas.bbox("all"))  
  
 def load\_flavors(self):  
 cursor.execute("SELECT name FROM flavors")  
 flavors = cursor.fetchall()  
  
 row\_index = 1  
 col\_index = 0  
  
 for flavor in flavors:  
 if col\_index == 3:  
 row\_index += 1  
 col\_index = 0  
  
 self.create\_flavor\_button(flavor[0], row\_index, col\_index)  
 col\_index += 1  
  
 cursor.execute("SELECT name FROM suggested\_flavors")  
 suggested\_flavors = cursor.fetchall()  
  
 for flavor in suggested\_flavors:  
 if col\_index == 3:  
 row\_index += 1  
 col\_index = 0  
  
 self.create\_flavor\_button(flavor[0], row\_index, col\_index)  
 col\_index += 1  
  
 def create\_flavor\_button(self, flavor\_name, row, column):  
 image\_path = os.path.join("C:/Users/admin/PycharmProjects/pythonProject3", f"{flavor\_name}.jpg")  
 if os.path.exists(image\_path):  
 image = Image.open(image\_path)  
 image = image.resize((100, 100), Image.LANCZOS)  
  
 photo = ImageTk.PhotoImage(image)  
 button = Button(self.inner\_frame, text=flavor\_name, image=photo, compound="top", bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 12), command=lambda name=flavor\_name: self.add\_to\_cart(name))  
 button.image = photo # Keep a reference to avoid garbage collection  
 button.grid(row=row, column=column, padx=10, pady=10)  
 else:  
 button = Button(self.inner\_frame, text=flavor\_name, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 12), command=lambda name=flavor\_name: self.add\_to\_cart(name))  
 button.grid(row=row, column=column, padx=10, pady=10)  
  
 def search\_flavors(self):  
 query = self.search\_entry.get().lower()  
 for widget in self.inner\_frame.winfo\_children():  
 if isinstance(widget, Button):  
 widget.destroy()  
  
 cursor.execute("SELECT name FROM flavors WHERE LOWER(name) LIKE ?", ('%' + query + '%',))  
 flavors = cursor.fetchall()  
 row\_index = 1  
 col\_index = 0  
 for flavor in flavors:  
 if col\_index == 3:  
 row\_index += 1  
 col\_index = 0  
  
 self.create\_flavor\_button(flavor[0], row\_index, col\_index)  
 col\_index += 1  
  
 cursor.execute("SELECT name FROM suggested\_flavors WHERE LOWER(name) LIKE ?", ('%' + query + '%',))  
 suggested\_flavors = cursor.fetchall()  
 for flavor in suggested\_flavors:  
 if col\_index == 3:  
 row\_index += 1  
 col\_index = 0  
  
 self.create\_flavor\_button(flavor[0], row\_index, col\_index)  
 col\_index += 1  
  
 # Add the back button again  
 self.back\_button = Button(self.inner\_frame, text="Back", command=self.create\_main\_window, bg=NETFLIX\_RED,  
 fg=NETFLIX\_WHITE, font=("Arial", 12))  
 self.back\_button.grid(row=row\_index, column=0, columnspan=3, pady=(10, 5), padx=(200, 0), sticky="w")  
  
 def add\_to\_cart(self, flavor\_name):  
 cursor.execute("SELECT \* FROM flavors WHERE name=?", (flavor\_name,))  
 flavor = cursor.fetchone()  
 if not flavor:  
 cursor.execute("SELECT \* FROM suggested\_flavors WHERE name=?", (flavor\_name,))  
 flavor = cursor.fetchone()  
  
 if flavor:  
 allergies = simpledialog.askstring("Input", f"Enter Allergies (comma separated) for {flavor\_name}:")  
 if allergies is not None and allergies.strip() != "":  
 cursor.execute("INSERT INTO cart (name, allergens) VALUES (?, ?)", (flavor\_name, allergies))  
 conn.commit()  
 messagebox.showinfo("Cart", f"Added {flavor\_name} to cart with allergies: {allergies}.")  
 else:  
 cursor.execute("INSERT INTO cart (name, allergens) VALUES (?, ?)", (flavor\_name, None))  
 conn.commit()  
 messagebox.showinfo("Cart", f"Added {flavor\_name} to cart.")  
 else:  
 messagebox.showerror("Error", "Flavor not found.")  
  
 def open\_cart\_window(self):  
 self.clear\_window()  
 self.cart\_listbox = Listbox(self.root, width=50, height=10)  
 self.cart\_listbox.pack(padx=10, pady=10)  
 self.load\_cart()  
  
 self.back\_button = Button(self.root, text="Back", command=self.create\_main\_window, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.back\_button.pack(pady=10)  
  
 def load\_cart(self):  
 for widget in self.root.winfo\_children():  
 widget.destroy()  
  
 self.cart\_label = Label(self.root, text="Cart Items", font=("Arial", 18), fg=NETFLIX\_WHITE, bg=NETFLIX\_BLACK)  
 self.cart\_label.pack(pady=10)  
  
 scrollbar = Scrollbar(self.root)  
 scrollbar.pack(side="right", fill="y")  
  
 self.cart\_frame = Frame(self.root, bg=NETFLIX\_BLACK)  
 self.cart\_frame.pack(fill="both", expand=True)  
  
 self.cart\_canvas = Canvas(self.cart\_frame, bg=NETFLIX\_BLACK, yscrollcommand=scrollbar.set)  
 self.cart\_canvas.pack(side="left", fill="both", expand=True)  
 scrollbar.config(command=self.cart\_canvas.yview)  
  
 self.cart\_frame\_inner = Frame(self.cart\_canvas, bg=NETFLIX\_BLACK)  
 self.cart\_canvas.create\_window((0, 0), window=self.cart\_frame\_inner, anchor="nw")  
  
 cursor.execute("SELECT \* FROM cart")  
 cart\_items = cursor.fetchall()  
  
 row\_index = 0  
 for item in cart\_items:  
 flavor\_name = item[0]  
 allergens = item[1]  
 image\_path = os.path.join("C:/Users/admin/PycharmProjects/pythonProject3",  
 f"{flavor\_name}.jpg")  
 if os.path.exists(image\_path):  
 image = Image.open(image\_path)  
 image = image.resize((100, 100), Image.LANCZOS)  
 photo = ImageTk.PhotoImage(image)  
 button = Button(self.cart\_frame\_inner, text=flavor\_name, image=photo, compound="top",  
 bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 12),  
 command=lambda name=flavor\_name: self.remove\_from\_cart(name))  
 button.image = photo # Keep a reference to avoid garbage collection  
 button.grid(row=row\_index, column=0, padx=10, pady=10)  
  
 if allergens:  
 allergens\_label = Label(self.cart\_frame\_inner, text=f"Allergens: {allergens}", bg=NETFLIX\_BLACK,  
 fg=NETFLIX\_WHITE, font=("Arial", 10))  
 allergens\_label.grid(row=row\_index + 1, column=0, padx=10, pady=5)  
 else:  
 label = Label(self.cart\_frame\_inner, text=flavor\_name, bg=NETFLIX\_BLACK, fg=NETFLIX\_WHITE,  
 font=("Arial", 12))  
 label.grid(row=row\_index, column=0, padx=10, pady=10)  
  
 if allergens:  
 allergens\_label = Label(self.cart\_frame\_inner, text=f"Allergens: {allergens}", bg=NETFLIX\_BLACK,  
 fg=NETFLIX\_WHITE, font=("Arial", 10))  
 allergens\_label.grid(row=row\_index + 1, column=0, padx=10, pady=5)  
  
 row\_index += 2 # Increase by 2 to leave space for allergens if present  
  
 self.cart\_frame\_inner.update\_idletasks()  
 self.cart\_canvas.config(scrollregion=self.cart\_canvas.bbox("all"))  
  
 def remove\_from\_cart(self, flavor\_name):  
 cursor.execute("DELETE FROM cart WHERE name=?", (flavor\_name,))  
 conn.commit()  
 self.load\_cart()  
  
 def open\_suggest\_window(self):  
 self.clear\_window()  
 self.flavor\_name\_label = Label(self.root, text="Enter Flavor Name:", font=("Arial", 14), fg=NETFLIX\_WHITE, bg=NETFLIX\_BLACK)  
 self.flavor\_name\_label.pack(pady=5)  
 self.flavor\_name\_entry = Entry(self.root)  
 self.flavor\_name\_entry.pack(pady=5)  
 self.ingredients\_label = Label(self.root, text="Enter Ingredients (comma separated):", font=("Arial", 14), fg=NETFLIX\_WHITE, bg=NETFLIX\_BLACK)  
 self.ingredients\_label.pack(pady=5)  
 self.ingredients\_entry = Entry(self.root)  
 self.ingredients\_entry.pack(pady=5)  
 self.allergens\_label = Label(self.root, text="Enter Allergens (comma separated):", font=("Arial", 14), fg=NETFLIX\_WHITE, bg=NETFLIX\_BLACK)  
 self.allergens\_label.pack(pady=5)  
 self.allergens\_entry = Entry(self.root)  
 self.allergens\_entry.pack(pady=5)  
 self.submit\_suggest\_button = Button(self.root, text="Submit", command=self.suggest\_flavor, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.submit\_suggest\_button.pack(pady=10)  
 self.back\_button = Button(self.root, text="Back", command=self.create\_main\_window, bg=NETFLIX\_RED, fg=NETFLIX\_WHITE, font=("Arial", 14))  
 self.back\_button.pack(pady=10)  
  
 def suggest\_flavor(self):  
 flavor\_name = self.flavor\_name\_entry.get()  
 ingredients = self.ingredients\_entry.get().split(", ")  
 allergens = self.allergens\_entry.get().split(", ")  
 if self.allergens\_entry.get().strip() != "":  
 pass  
 else:  
 allergens = None  
 if flavor\_name and ingredients:  
  
 cursor.execute("SELECT COUNT(\*) FROM suggested\_flavors WHERE name=?", (flavor\_name,))  
 if cursor.fetchone()[0] == 0:  
 cursor.execute("INSERT INTO suggested\_flavors (name, seasonal, ingredients, allergens) VALUES (?, ?, ?, ?)",  
 (flavor\_name, 0, ", ".join(ingredients), ", ".join(allergens) if allergens else None))  
 conn.commit()  
 messagebox.showinfo("Suggestion", f"Suggested {flavor\_name} has been added.")  
 else:  
 messagebox.showwarning("Warning", f"Flavor {flavor\_name} is already suggested.")  
 else:  
 messagebox.showerror("Error", "Flavor name and ingredients are required.")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 root.title("Ice Cream Parlor")  
 root.geometry("800x600")  
 root.resizable(False, False) # Disable resizing  
 app = IceCreamParlorApp(root)  
 root.mainloop()