Use Cases Team 11 : Memory Leak

1. **Create a Custom Plan**
    a. Summary: Client selects a starting city and a list of other cities to travel to
        i. Actor: Potential European Traveler
        ii. Precondition: List of cities are available for client to choose from
        iii. Description:
            1. Client clicks “File” in menu bar and selects “Load” to load the cities
            2. Client selects a starting city
            3. Client clicks “Create Plan” button which shows a list of cities to choose from
            4. Client selects from 0 - 11/13 cities, clicks “Generate” button
            5. The most efficient plan is displayed on the right.
            6. The Client clicks “Start Trip” and new window is displayed.
            7. On new window, the starting city’s name is displayed at the top and the Client is allowed to choose from a list of foods available at the city.
            8. Client clicks “Next” to visit next closest city and repeats step 3. And 4. until the trip is done which opens a new window.
            9. On the new window, receipt of all bought foods are displayed along with total cost from each city and total distance traveled.
        iv. Post Condition: The client has finished the custom trip
2. **Plan a Trip from Berlin to the rest of available cities**
    a. Summary: Client travels from “Berlin” and travels to the rest of cities in the most efficient order
        i. Actor: Musician Tour Manager
        ii. Precondition: Without specific user-selection of which cities, the program can organize the most efficient travel plan with all cities
        iii. Description:
            1. Client clicks “File” in menu bar and selects “Load” to load the cities
            2. Client clicks “Berlin Travel” button which displays the optimized travel plan and opens a new window.
            3. On new window, the starting city’s name is displayed at the top and the Client is allowed to choose from a list of foods available at the city.
            4. Client clicks “Next” to visit next closest city and repeats step 3. And 4. until the trip is done which opens a new window.
            5. On the new window, receipt of all bought foods are displayed along with total cost from each city and total distance traveled.
        iv. Post Condition: The client has finished a trip from “Berlin” to the rest of other cities.
3. **Plan a Trip from Paris to a selected number of cities**
    a. Summary: Client travels from “Paris” and travels to a desired number of cities in the most efficient order
        i. Actor: European Travel Agent

ii. Precondition: Without specific user-selection of which cities, the program can determine a plan a shortest trip based on the given number of cities

iii. Description:

1. Client clicks “File” in menu bar and selects “Load” to load the cities
2. Client clicks “Paris Travel” button which displays the optimized travel plan and opens a new window.
3. Client from 0 - total number of available cities, clicks “Generate” button
4. The most efficient plan is displayed on the right.
5. The Client clicks “Start Trip” and new window is displayed.
6. On new window, the starting city’s name is displayed at the top and the Client is allowed to choose from a list of foods available at the city.
7. Client clicks “Next” to visit next closest city and repeats step 3. And 4. until the trip is done which opens a new window.
8. On the new window, receipt of all bought foods are displayed along with total cost from each city and total distance traveled.

iv. Post Condition: The client has finished a trip from “Paris” to the rest of other cities.

4. Maintenance - Change prices of food items (in case Use Case 1,2,3 is too repetitive)

a. Summary: Allows the ability to make changes to the food data within the program

i. Actor: Software Administrator

ii. Precondition: Requires user to log-in in order to get access to maintenance functions

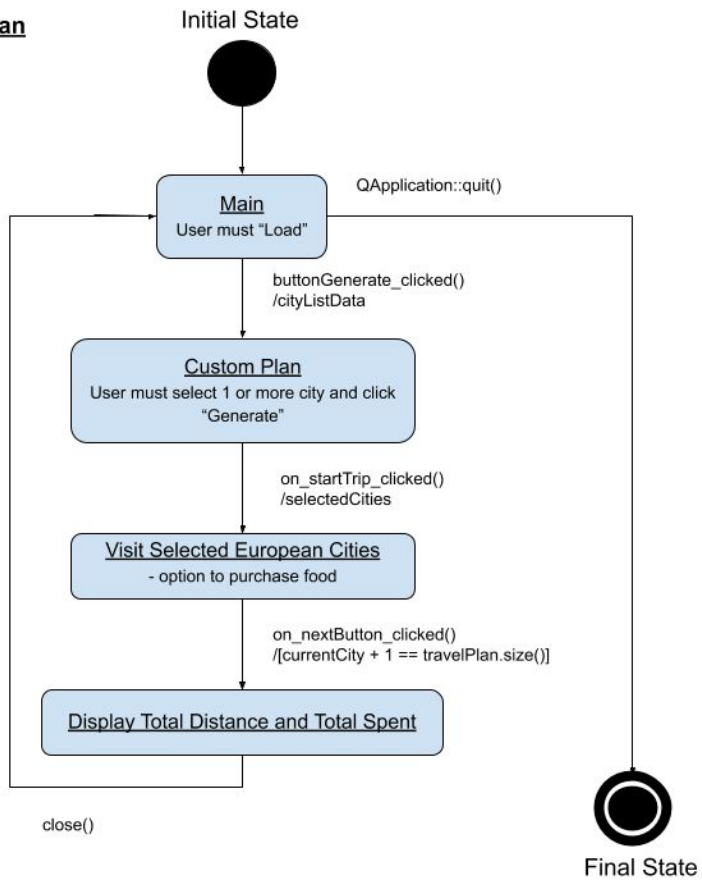
iii. Description:

1. User clicks “File” in menu bar and selects “Admin”.
2. A log-in window pops up and user enters correct username and password and selects “Login” button.
3. A Dialogue is popped-up notifying whether the log-in information is correct or incorrect.
    a. Alternative: If log-in information is correct, log-in window is still present for the user to re-type the information
4. User clicks “File” in menu bar and selects “Load” or “Load Extended”
5. On the right hand side of the main window, a new set of UI elements are displayed.
6. From the top of the new UI group, user selects a city and clicks “Load Foods”.
7. The name of the initial food in the selected city’s food list appears in the drop-down box right below “Load Foods”.
8. User selects a desired food.
9. In a text-entry box below the drop-down box, User enters a positive number either in whole numbers or decimal for the new price.
10. User selects “Change Price” to complete and confirm the change.

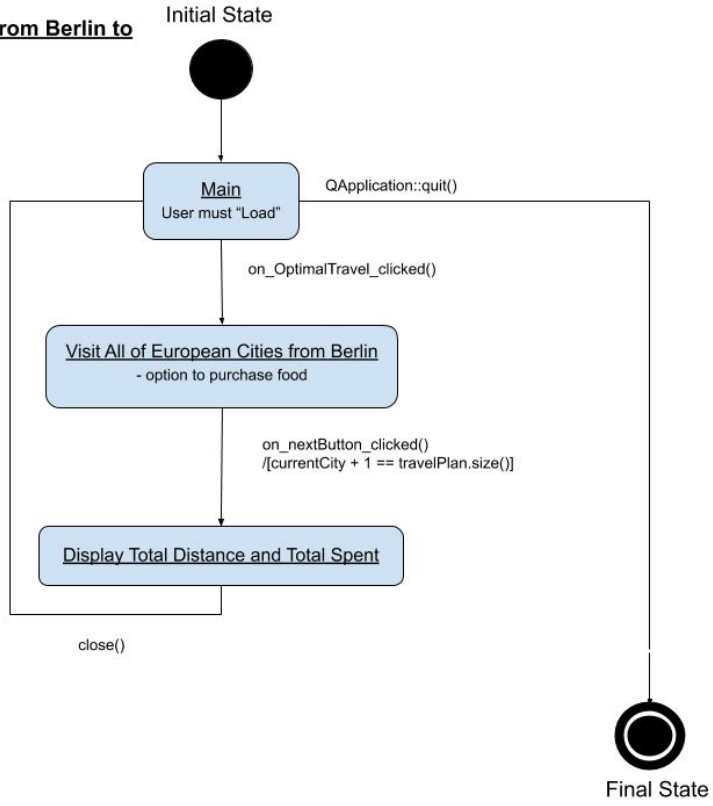
iv. mainPost Condition: The selected food item has an updated price.

State Diagrams

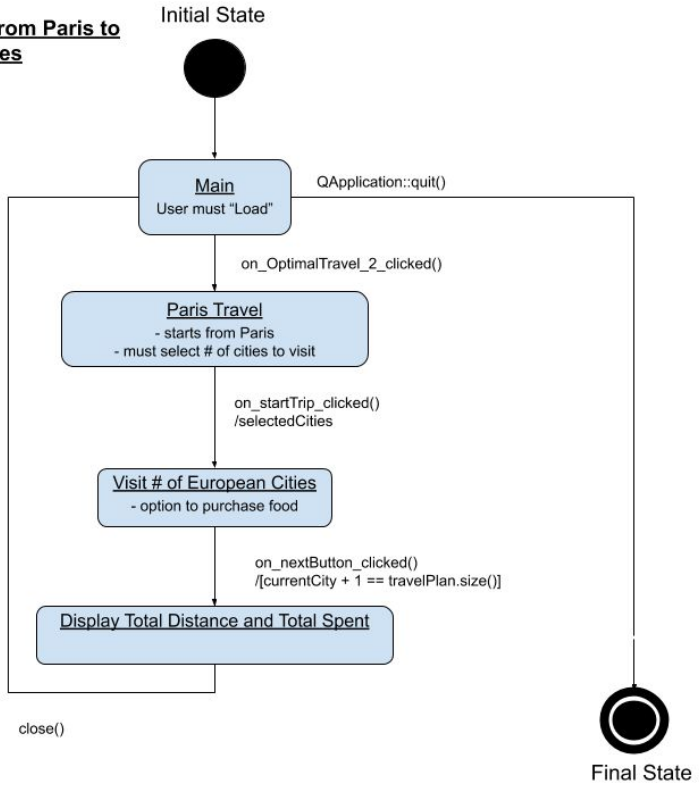
Use Case 1: Create a Custom Plan



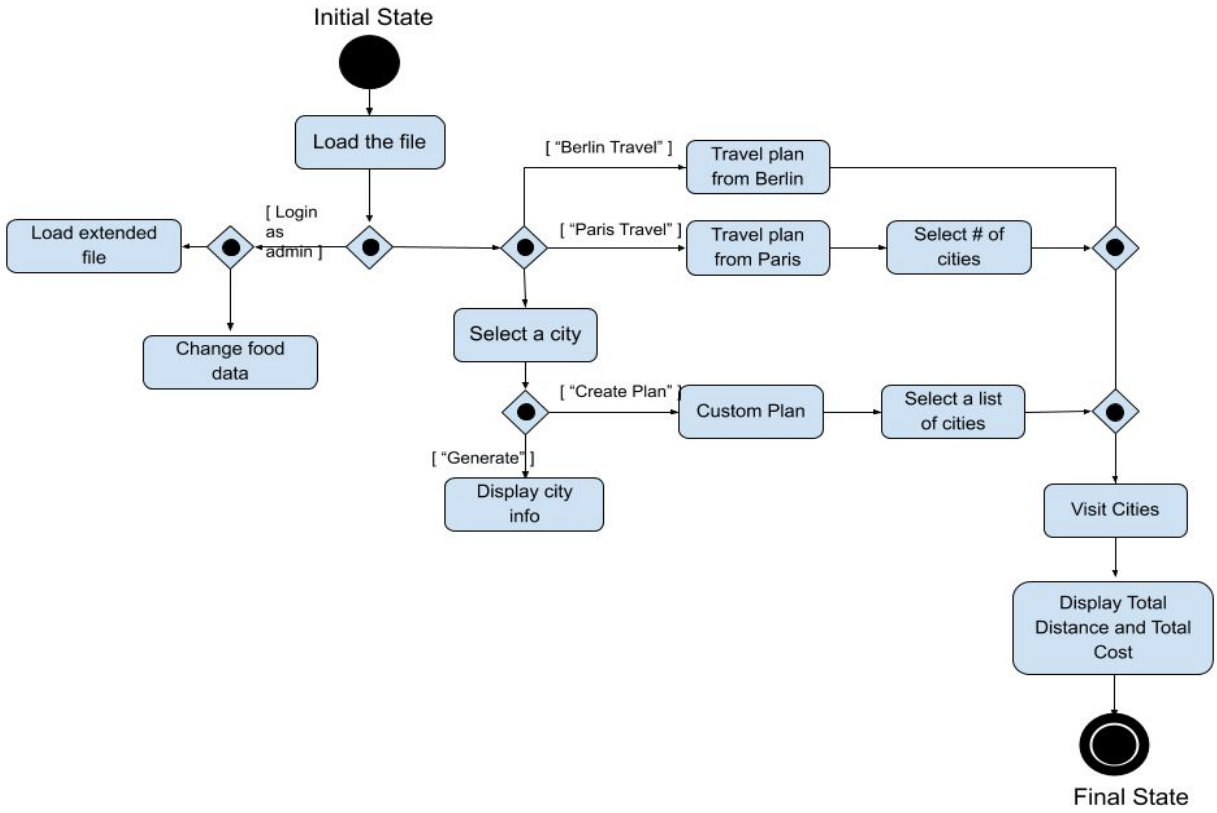
Use Case 2: Plan a Trip from Berlin to all available of cities



Use Case 3: Plan a Trip from Paris to a selected number of cities



Activity Diagram



UML Class Diagrams