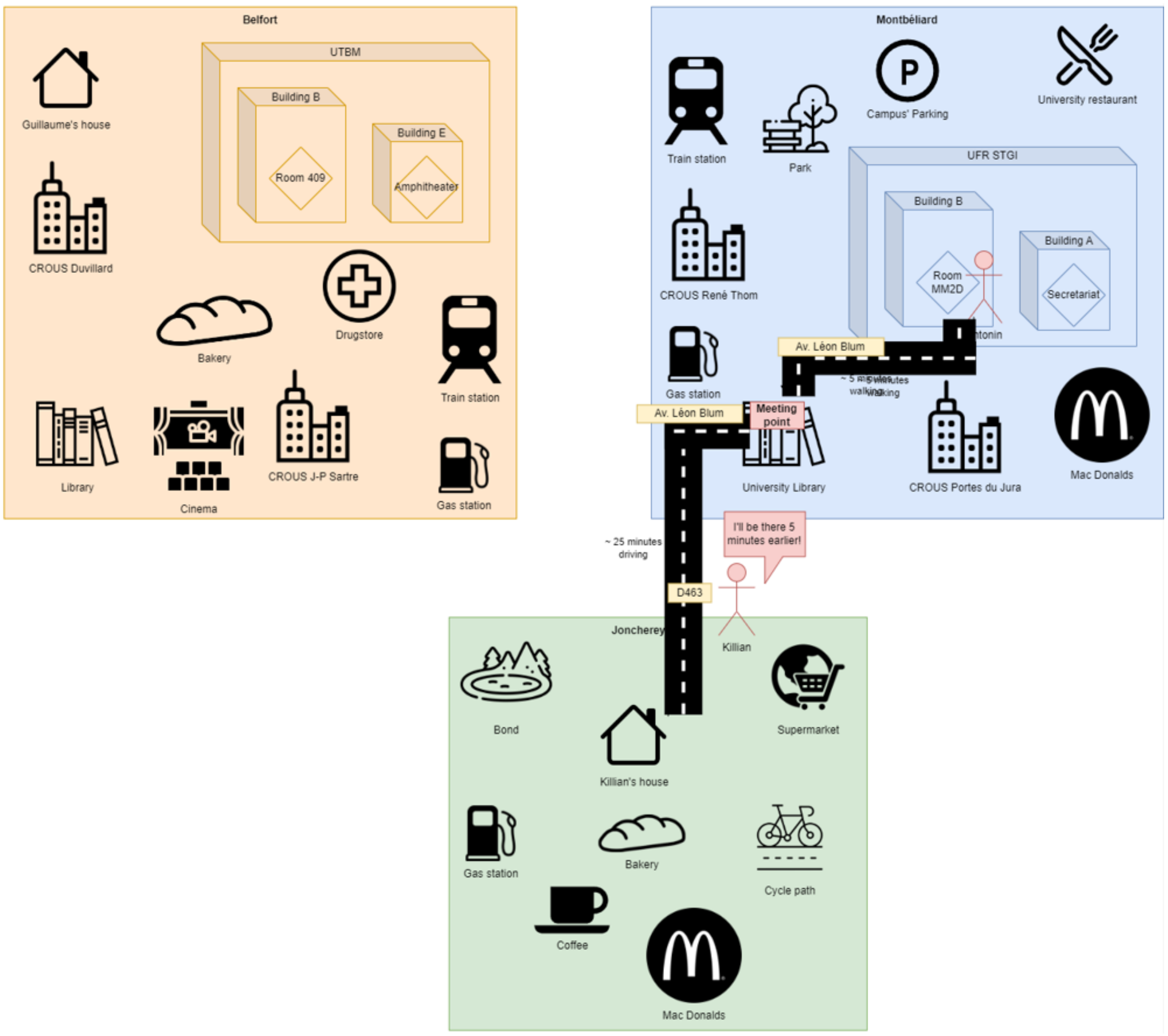
MINI-PROJECT MSC CONCEPTION



Antonin WINTERSTEIN
Killian MAXEL

Teacher: Mr. Philippe CANALDA

Introduction

The goal of the project is either to create a mobile application or a web-based application to map students' main points of interest (home, parking, university...), using an Origin-Destination (OD) matrix for these spaces, and facilitating group meetups among registered friends.

This application is designed to efficiently determine optimal meeting places and time windows for groups of friends. Additionally, it aims to generate individual routes for each friend and offer real-time tracking to signal any delays or advancements.

Specification of the subject

As specified in the subject, we must create an application that should have the capability to map various student locations. These locations can include homes, classrooms, places of daily activity, meeting points, car parks, and more.

We will also need to use an Origin-Destination matrix (with a size of 30 locations) for these locations. This matrix should provide details such as distances, time estimates, and route options between different places. This feature is vital for determining optimal meetup locations.

Three functions are to do using the map and the Origin-Destination matrix:

1. The main function is to determine a place X and a time-window where a group of friends can meet at the earliest taking into account each friend's location and schedule.
2. The second major function is to generate individual routes for each friend to reach the meetup location.
3. The third and last function is to follow friends as they make their way to the meetup location to provide alerts for delays or advancements.

Environment to consider

We should consider creating a map where we need to show some places frequented by students (in our case at Belfort-Montbéliard).

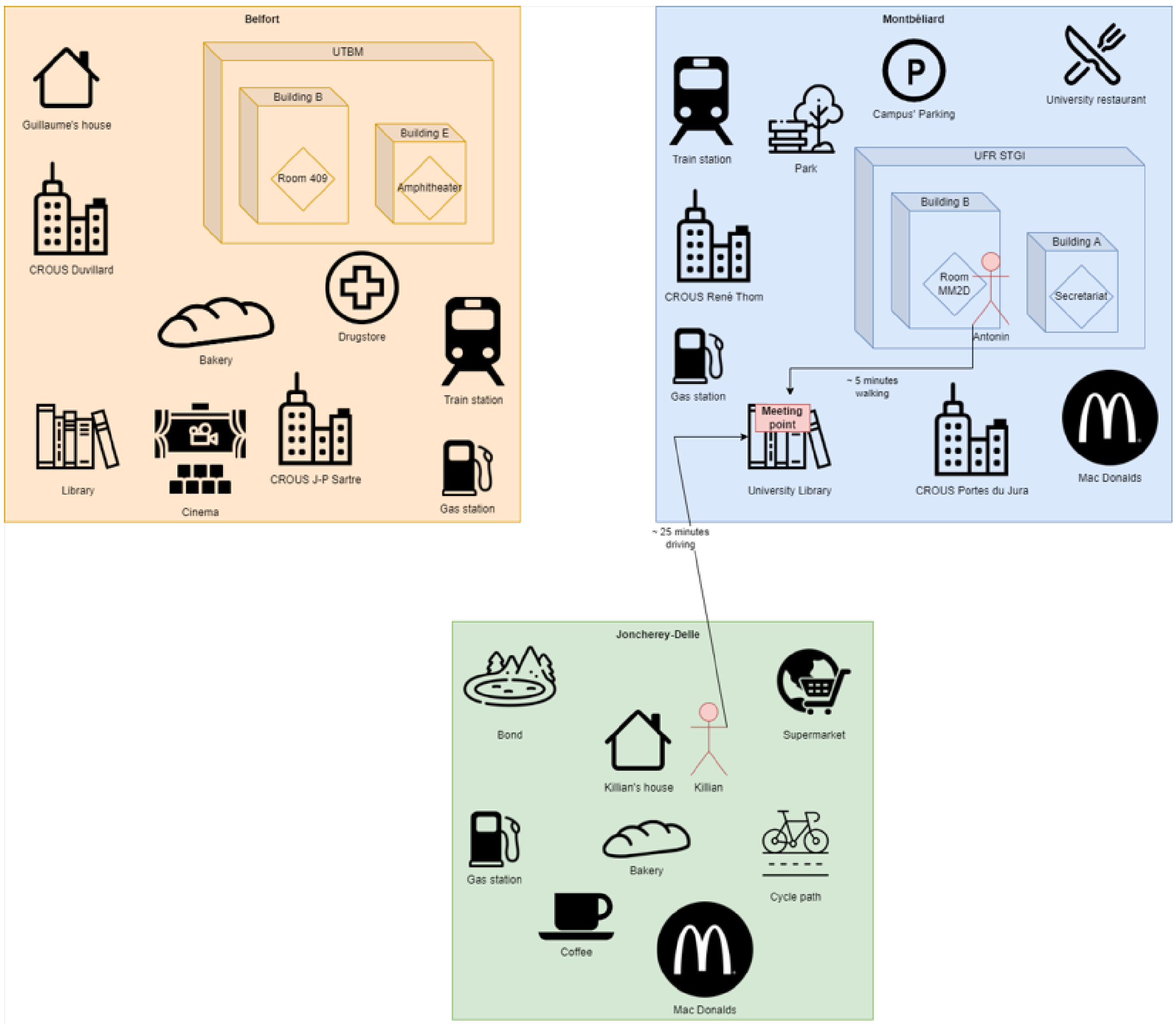
We will also need to register friends in order to place them on the points of interest depending on their actual position and also to allow the use of the Origin-Destination matrix where we need to determine the meeting point.

Next we need to include roads on the map in order to allow the generation of customized routes for each friend and also to determine, by tracking the friend going to the meeting point, if he is late or early.

Graphical resolution

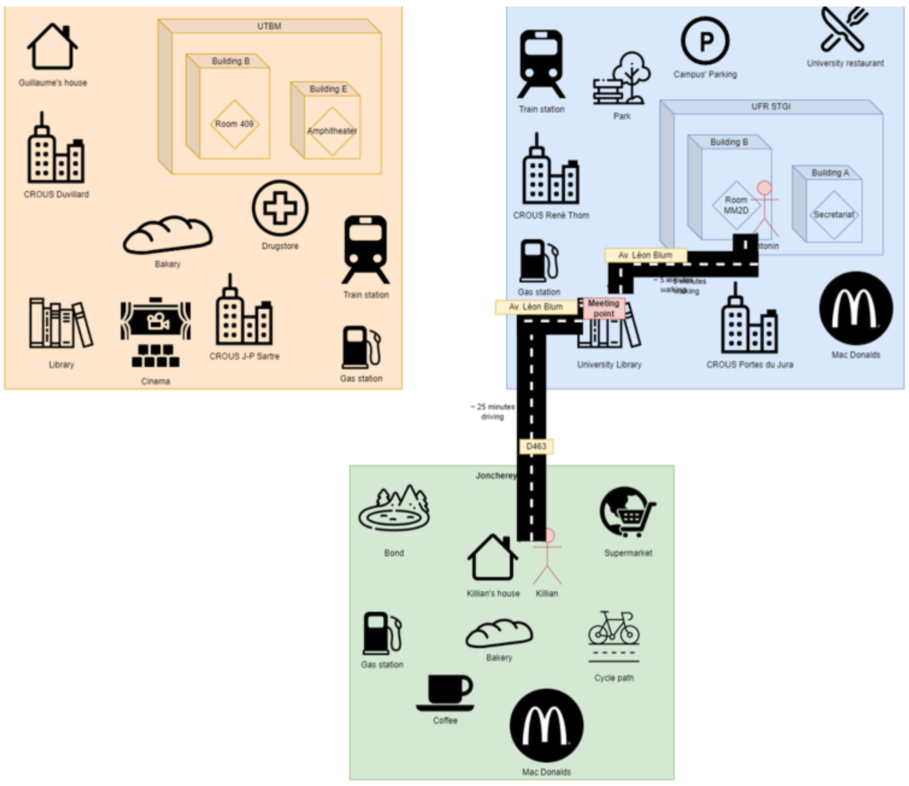
To try resolving the initial problem. We started by creating graphics for each part of the subject.

We first represented our 30 locations in three zones with two friends located at two of these locations (in this graphic, Antonin is in the MM2D room and Killian in Killian's house). From the positions of these friends, a meeting point must be determined in order to allow them to meet at the earliest. In this case it seems like the university library is the optimal candidate.



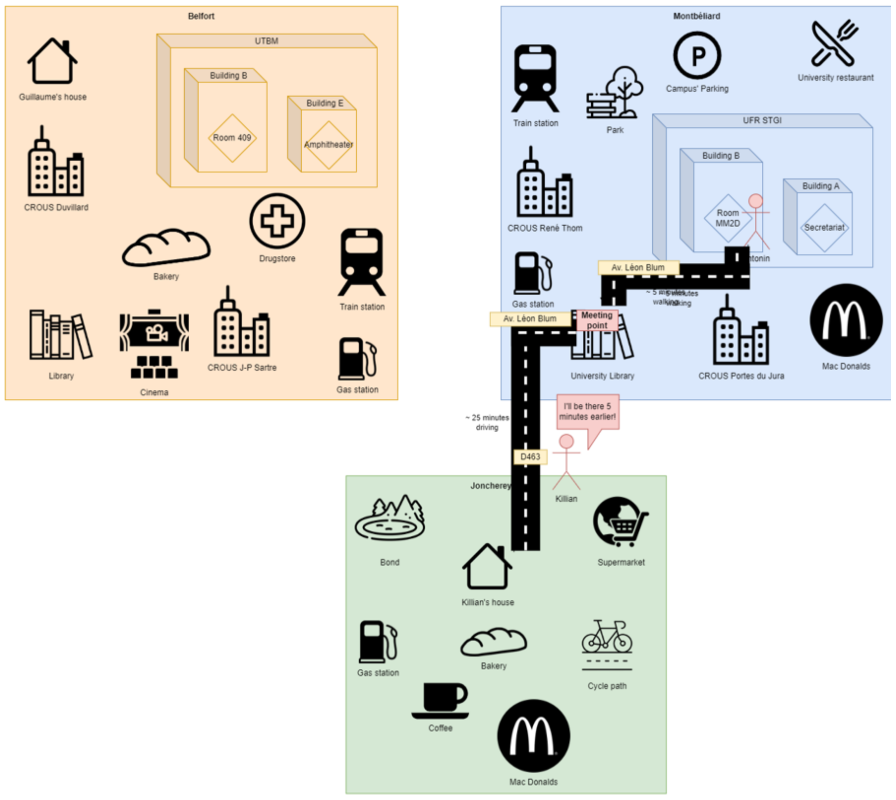
First step/function of the project representation with the election of the meeting point

Next, using the meeting point location and the locations of each friend, we must generate individual routes for each friend to reach the meetup location with which road to take, the distance, time of travel and so on.



Second step/function of the project representation with the route of each friend towards the meeting point

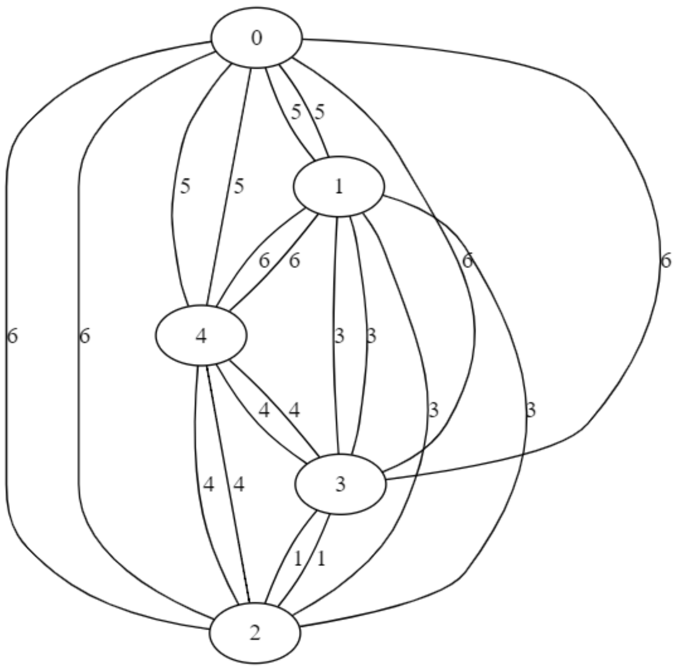
Finally, we should provide for the last function a way to alert if the friend will have delay or advance regarding its way to the meeting point.



Third step/function of the project representation with Killian alerting that he will be there earlier

Graph representation of the OD Matrix

In order to validate the values assigned to the OD matrix and ensure that the properties of reflexivity, symmetry, and transitivity are correctly followed, we opted to create a second graphical representation using the dot format and the graphviz library.

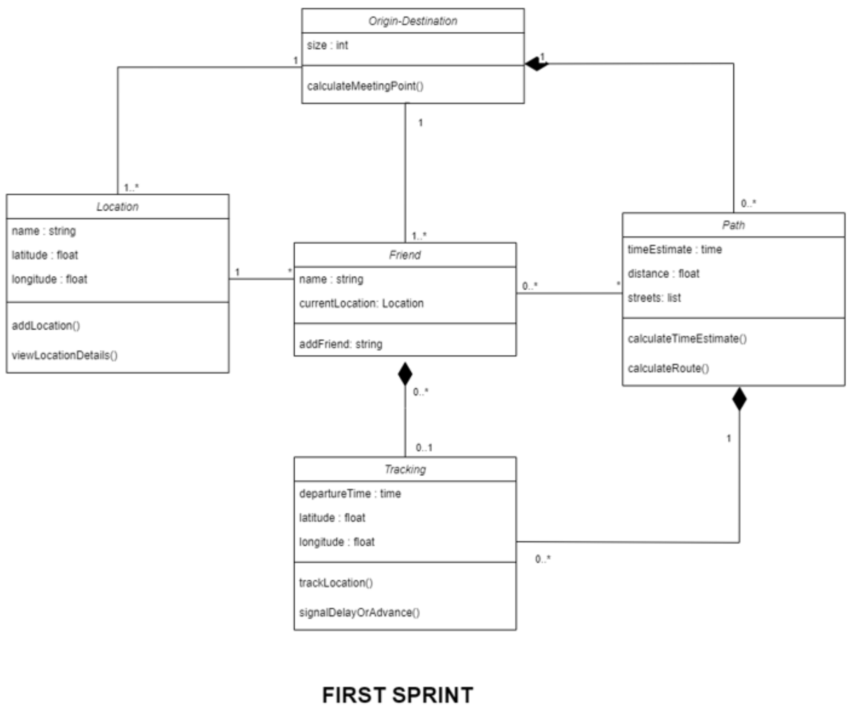


Simplified Graph representation of our matrix
(only 5 locations for visibility)
(complete version here)

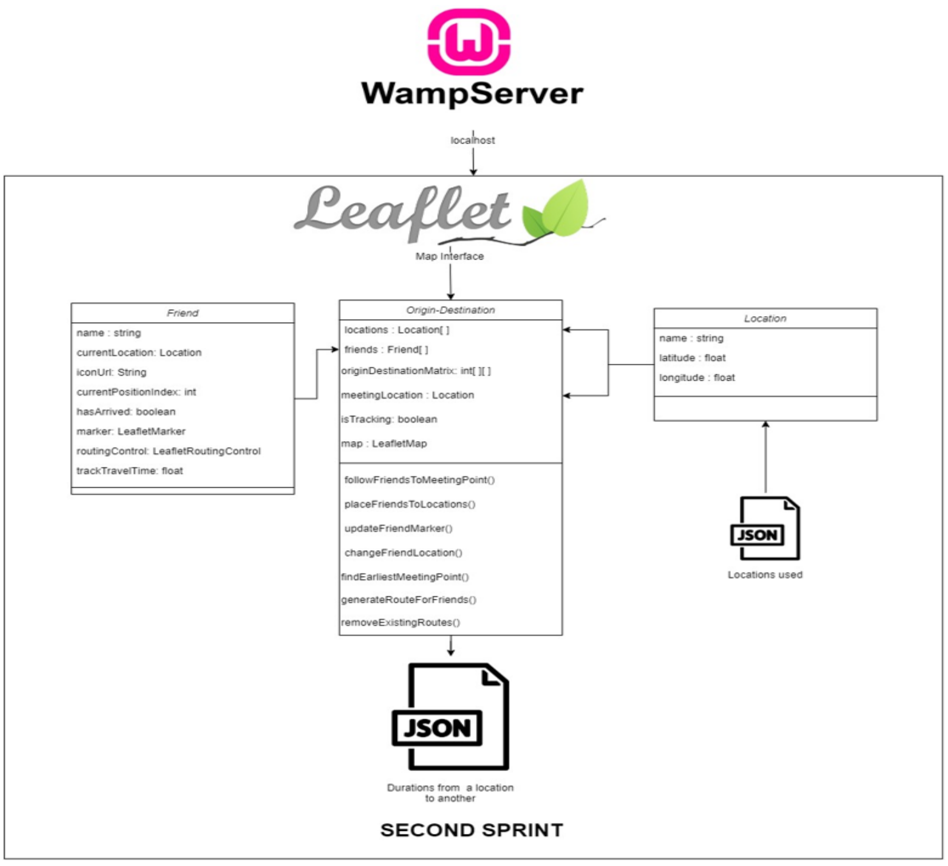
But because we have a 30x30 matrix, it is difficult to notice anything on it.

Class diagrams

After having represented the problem, we decided to work on a class diagram. For that, we first had a global representation but in practice, we changed it while developing to ease our implementation.



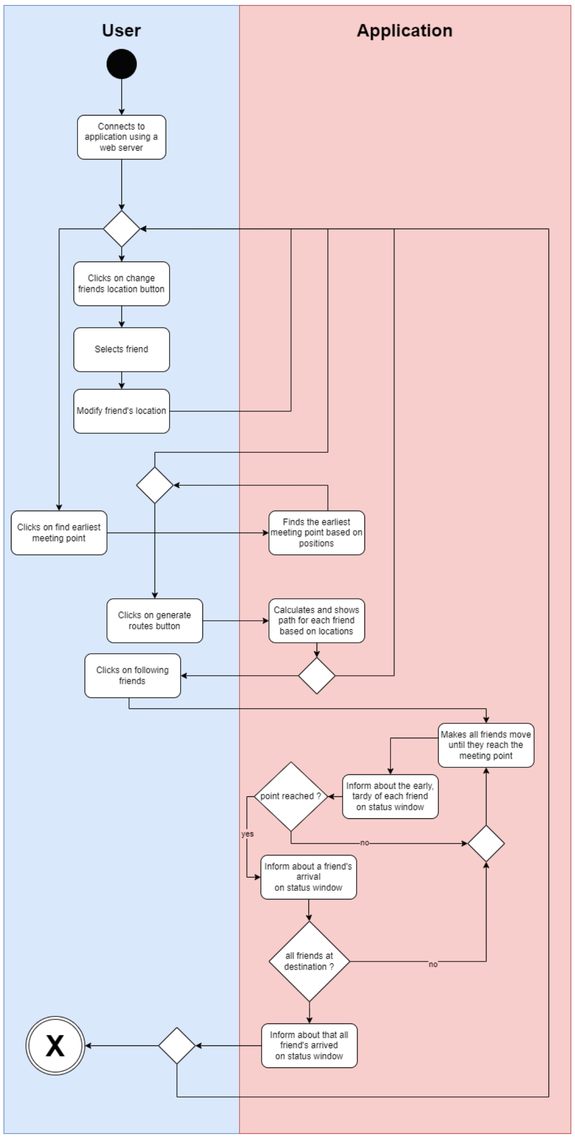
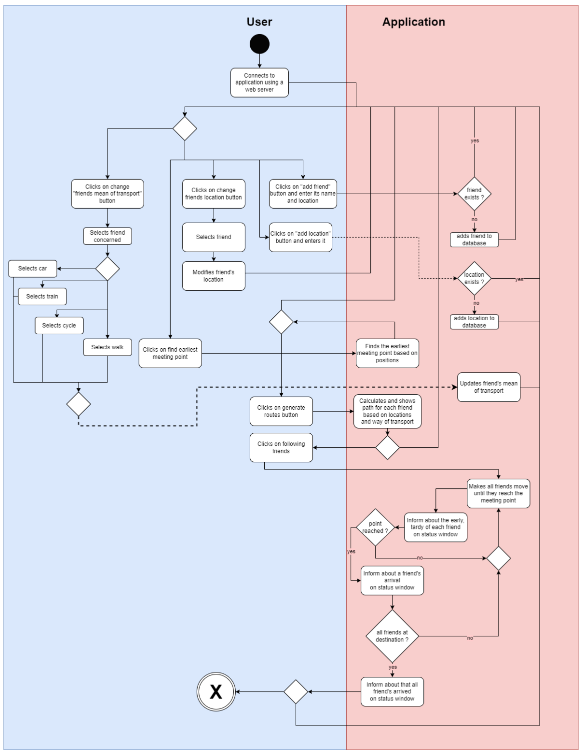
First class diagram



Final class diagram

Activity diagrams

After making a representation of our classes we created activity diagrams in order to have a clear view of how our application will work.

 User Application Connects to application using a web server Clicks on change friends location button Selects friend Modify friend's location Clicks on find earliest meeting point Finds the earliest meeting point based on positions Clicks on generate routes button Calculates and shows path for each friend based on locations Clicks on following friends Makes all friends move until they reach the meeting point point reached ? Inform about the early, tardy of each friend on status window yes no Inform about a friend's arrival on status window all friends at destination ? no Inform about that all friend's arrived on status window X	 User Application Connects to application using a web server yes Clicks on change "friends mean of transport" button Clicks on change friends location button Clicks on "add friend" button and enter its name and location friend exists ? no adds friend to database Selects friend concerned Selects friend Clicks on "add location" button and enters it Selects car Selects train Selects cycle Selects walk Modifies friend's location location exists ? yes no adds location to database Clicks on find earliest meeting point Finds the earliest meeting point based on positions Updates friend's mean of transport Clicks on generate routes button Calculates and shows path for each friend based on locations and way of transport Clicks on following friends Makes all friends move until they reach the meeting point point reached ? Inform about the early, tardy of each friend on status window yes no Inform about a friend's arrival on status window all friends at destination ? no yes Inform about that all friend's arrived on status window X
Activity diagram showing basics features	Updated activity diagram for final application's features
Link to full sized image	Link to full sized image

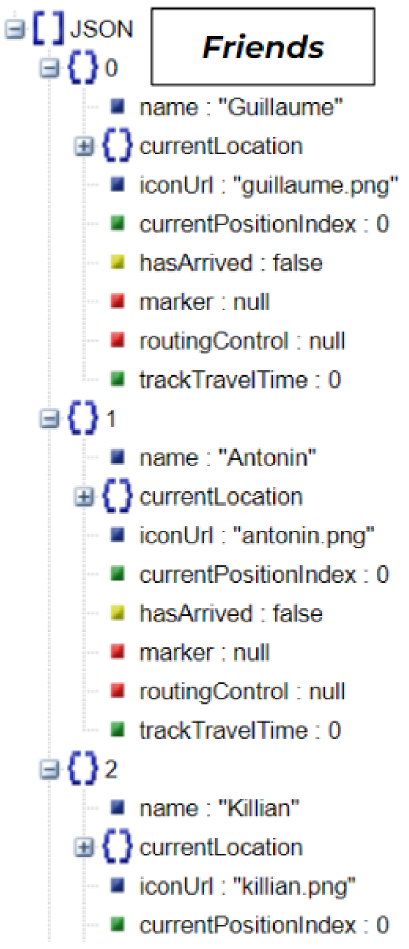
Dataset

In order to be able to develop the application with its different constraints, we created a static dataset manually.

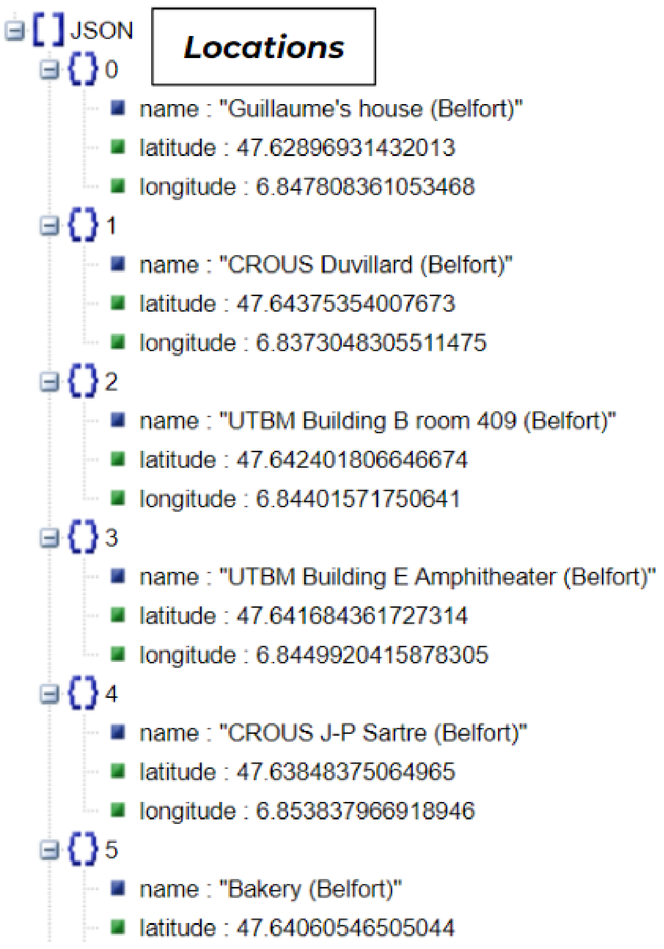
This dataset must have information about locations, friends and time of travel between locations (i.e the Origin-Destination matrix).

To tackle this problem, we created three JSON files which are:

- ***locationsData***: contains information of 30 locations, each holding three attributes (the name of the location, its latitude and its longitude).
- ***originDestinationData***: contains the time of travel in minutes between each location in a 2D array.
- ***friendsData***: contains information of 4 friends, each holding several attributes (the name of the friend, the current location, the custom iconUrl, the current position index of the route, a boolean to know if he arrived at the destination, the marker needed to show him, the information of its route and the travel time to know if he's in advance or has delay).

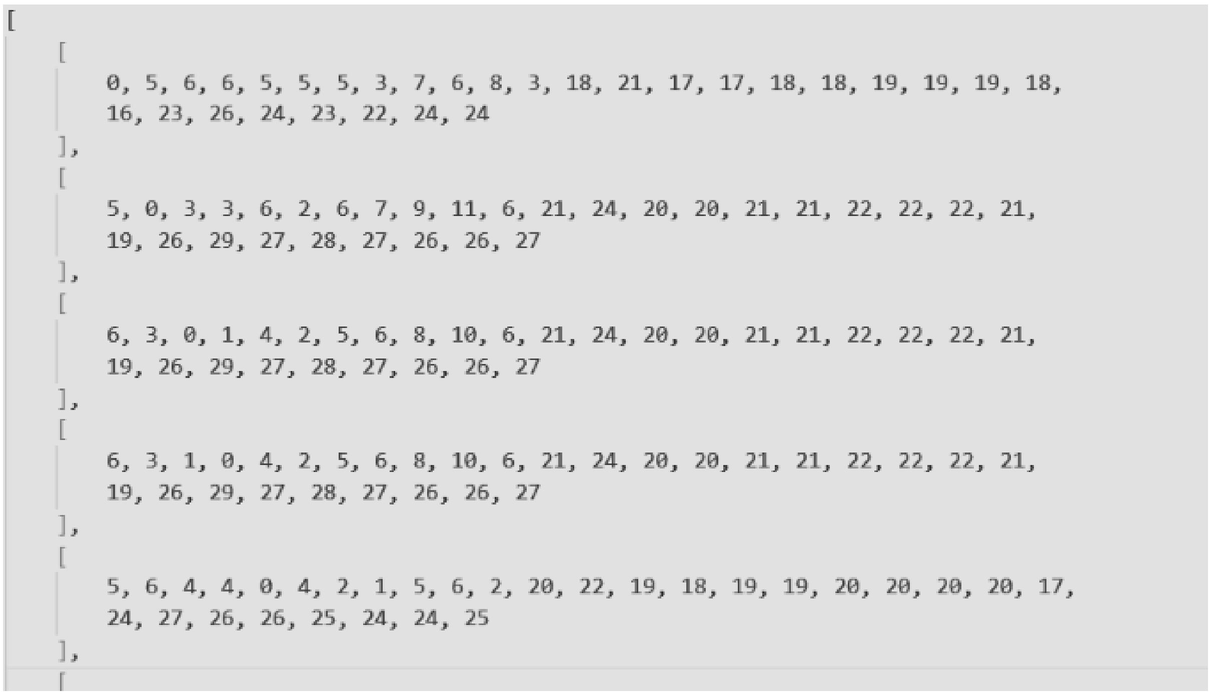


```
[ ] JSON   Friends
  { } 0
      name : "Guillaume"
    { } currentLocation
      iconUrl : "guillaume.png"
      currentPositionIndex : 0
      hasArrived : false
      marker : null
      routingControl : null
      trackTravelTime : 0
  { } 1
      name : "Antonin"
    { } currentLocation
      iconUrl : "antonin.png"
      currentPositionIndex : 0
      hasArrived : false
      marker : null
      routingControl : null
      trackTravelTime : 0
  { } 2
      name : "Killian"
    { } currentLocation
      iconUrl : "killian.png"
      currentPositionIndex : 0
```



```
[ ] JSON   Locations
  { } 0
      name : "Guillaume's house (Belfort)"
      latitude : 47.62896931432013
      longitude : 6.847808361053468
  { } 1
      name : "CROUS Duvillard (Belfort)"
      latitude : 47.64375354007673
      longitude : 6.8373048305511475
  { } 2
      name : "UTBM Building B room 409 (Belfort)"
      latitude : 47.642401806646674
      longitude : 6.84401571750641
  { } 3
      name : "UTBM Building E Amphitheater (Belfort)"
      latitude : 47.641684361727314
      longitude : 6.8449920415878305
  { } 4
      name : "CROUS J-P Sartre (Belfort)"
      latitude : 47.63848375064965
      longitude : 6.853837966918946
  { } 5
      name : "Bakery (Belfort)"
      latitude : 47.64060546505044
```

Preview of our friendsData and locationsData



```
[
    [
        0, 5, 6, 6, 5, 5, 5, 3, 7, 6, 8, 3, 18, 21, 17, 17, 18, 18, 19, 19, 19, 18,
        16, 23, 26, 24, 23, 22, 24, 24
    ],
    [
        5, 0, 3, 3, 6, 2, 6, 7, 9, 11, 6, 21, 24, 20, 20, 21, 21, 22, 22, 22, 21,
        19, 26, 29, 27, 28, 27, 26, 26, 27
    ],
    [
        6, 3, 0, 1, 4, 2, 5, 6, 8, 10, 6, 21, 24, 20, 20, 21, 21, 22, 22, 22, 21,
        19, 26, 29, 27, 28, 27, 26, 26, 27
    ],
    [
        6, 3, 1, 0, 4, 2, 5, 6, 8, 10, 6, 21, 24, 20, 20, 21, 21, 22, 22, 22, 21,
        19, 26, 29, 27, 28, 27, 26, 26, 27
    ],
    [
        5, 6, 4, 4, 0, 4, 2, 1, 5, 6, 2, 20, 22, 19, 18, 19, 19, 20, 20, 20, 20, 17,
        24, 27, 26, 26, 25, 24, 24, 25
    ],
    [
```

Preview of our originDestinationData

Technologies considered

Since we are storing our dataset in JSON files, we decided to create a web-based application using *HTML-CSS-JS*. Because we want to define specific locations with the possibility to determine a meeting point and a route for each location, we thought about displaying a map using the Leaflet API. It also allows us to easily add markers for these locations but also for the friends located on the map. For the routing part, we also used the Leaflet Routing Machine extension.
Note that you should use a server in order to be able to run the application.