



# EBEC Paris Saclay

**Matthieu Annequin,  
Hicham Bouanani  
Arnaud Petit  
Simon Tronchi**

# Our Team



**Arnaud Petit**



**Simon Tronchi**



**Matthieu Annequin**



**Hicham Bouanani**

# Describe a location in natural language

- 01 API
- 02 Objectives
- 03 Our algorithms
- 04 Optimization of the requests
- 05 Data visualization
- 06 Our solution



# API

We use the overpass API, which is an API of OverStreetMap

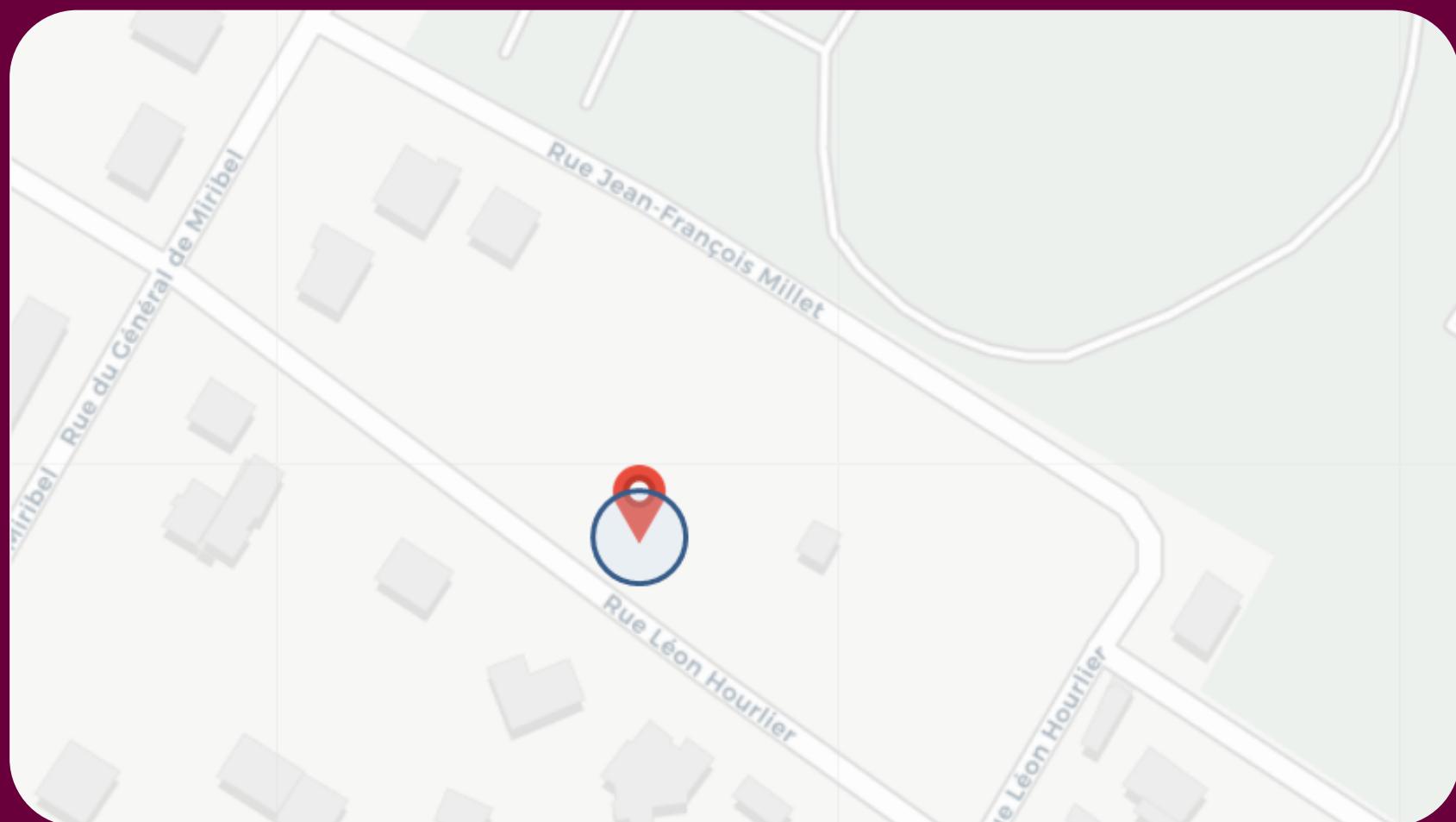
It takes a query and return a .json dataset

# Objectives

- **OBJECTIVE 0: FIND THE STREET OF GPS COORDINATES**
- **OBJECTIVE 1: GET A TEXTUAL DESCRIPTION OF THE POSITION OF GEOGRAPHIC POINT USING THE NEAREST CROSSINGS**
- **OBJECTIVE 2: DESCRIPTION OF SEVERAL POINTS ON A SECTION**
- **OBJECTIVE 3: DESCRIPTION OF TWO POINTS ON TWO CONSECUTIVE SECTIONS**

# Our algorithms

Binary search of the nearest city and the nearest street

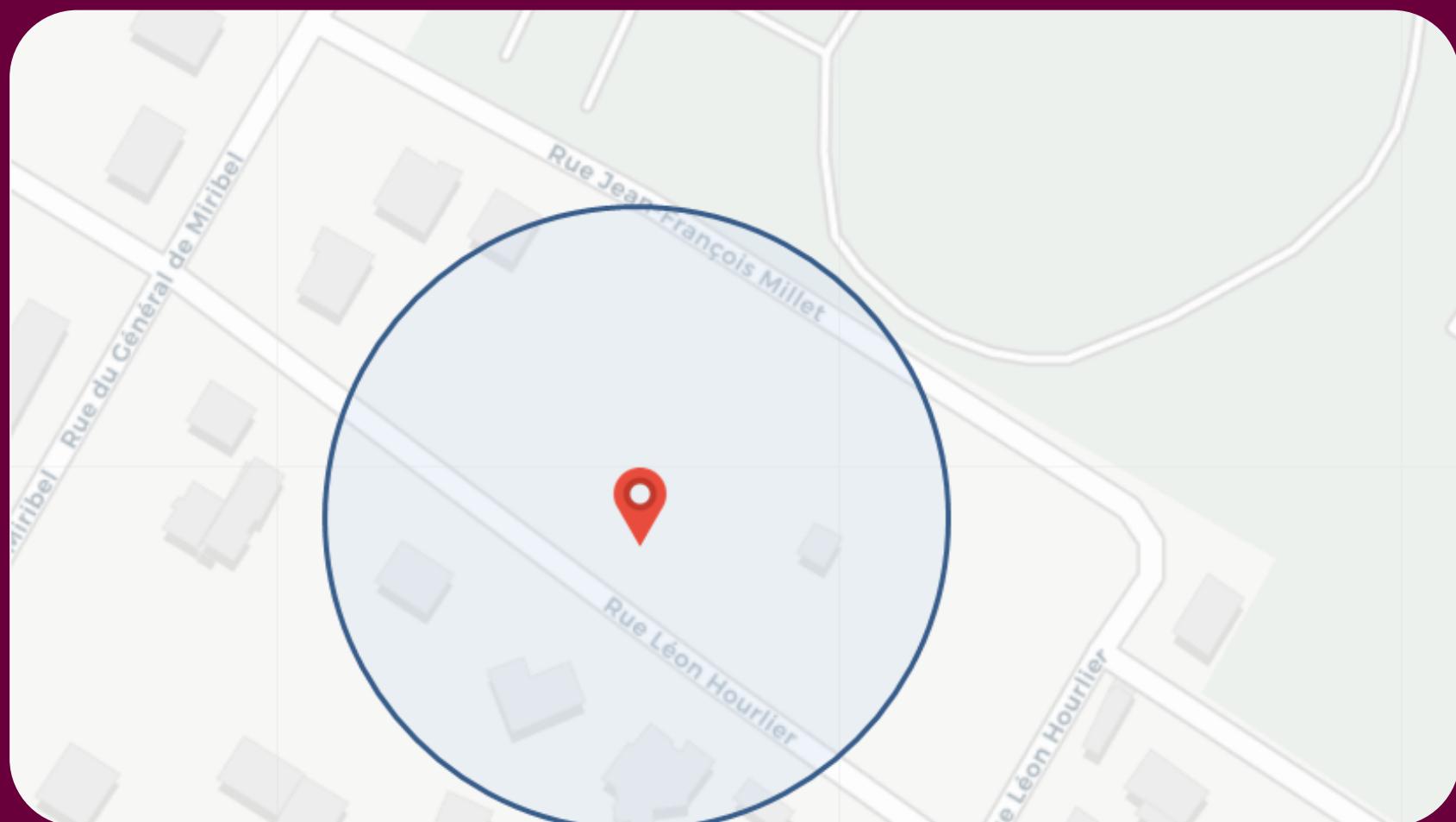


The algorithm uses concentric circles  
to find the nearest street

It adapts the radius using binary  
search

# Our algorithms

**Binary search of the nearest city and the nearest street**

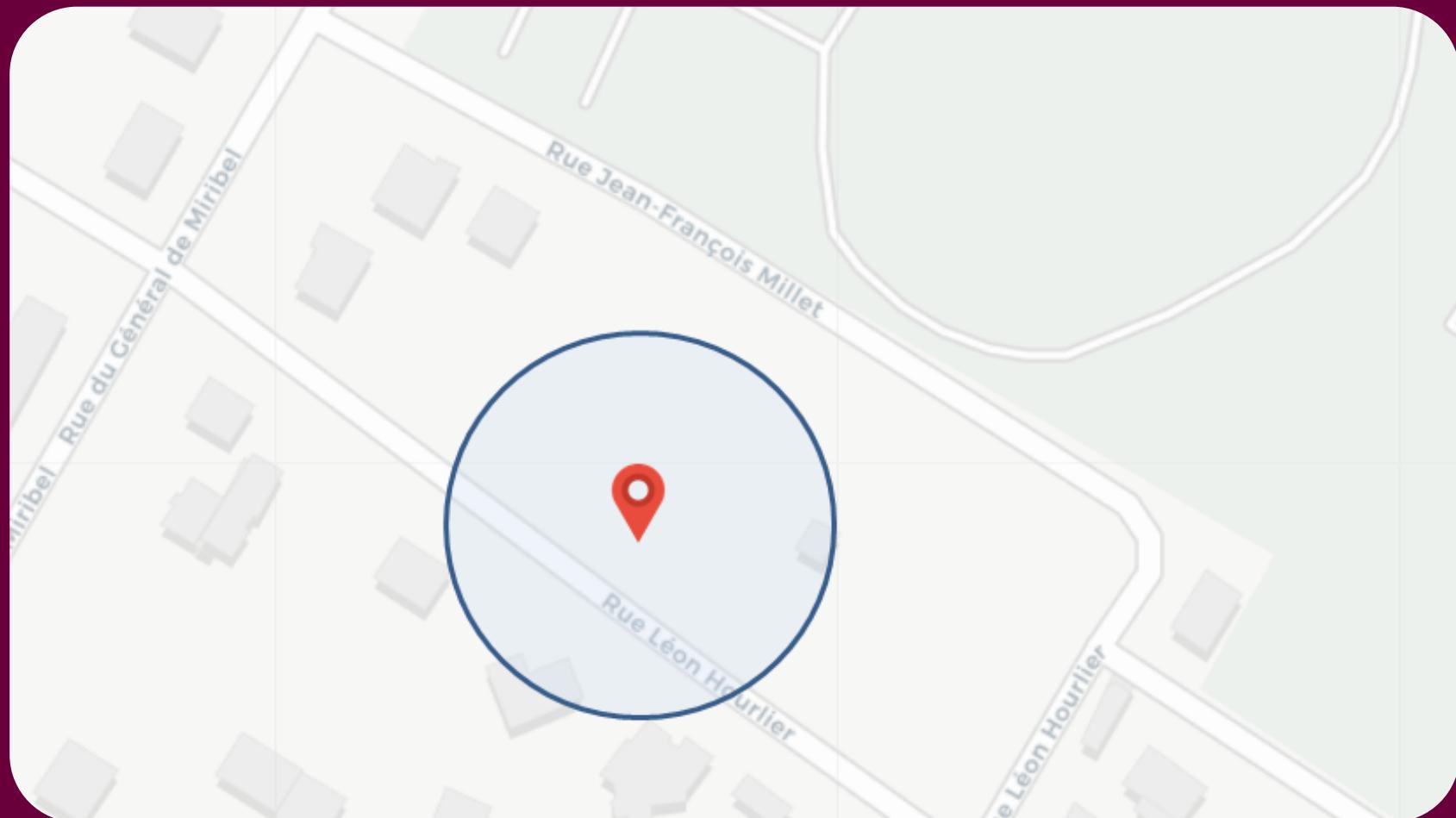


**The algorithm uses concentric circles  
to find the nearest street**

**It adapts the radius using binary  
search**

# Our algorithms

**Binary search of the nearest city and the nearest street**

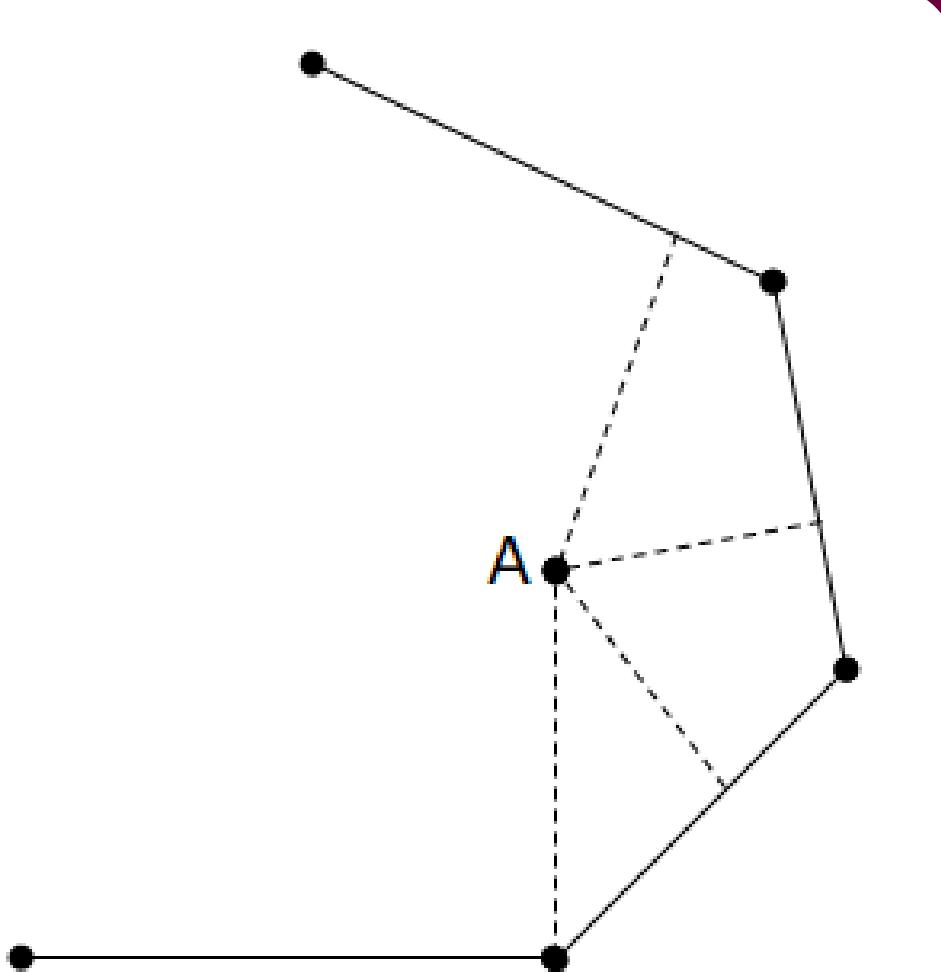
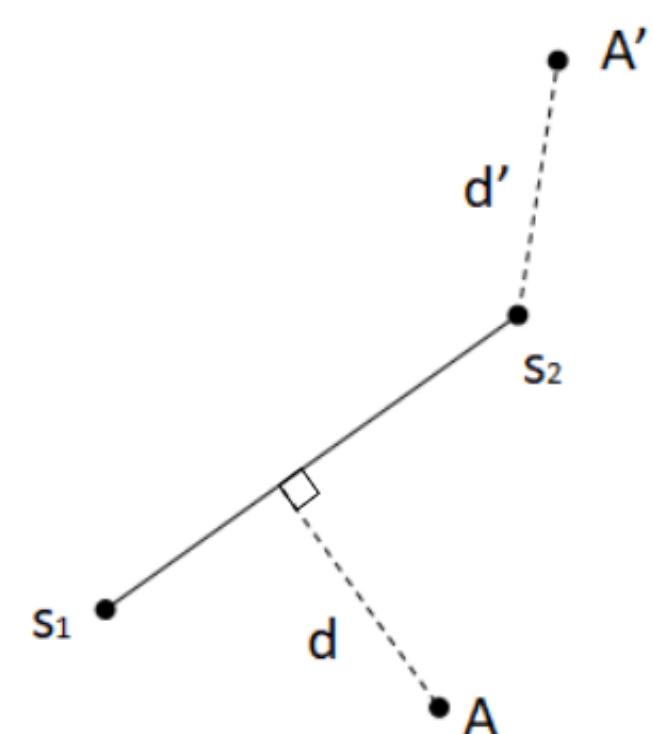


**The algorithm uses concentric circles  
to find the nearest street**

**It adapts the radius using binary  
search**

# Our algorithms

Computation of the distance between a point and a segment

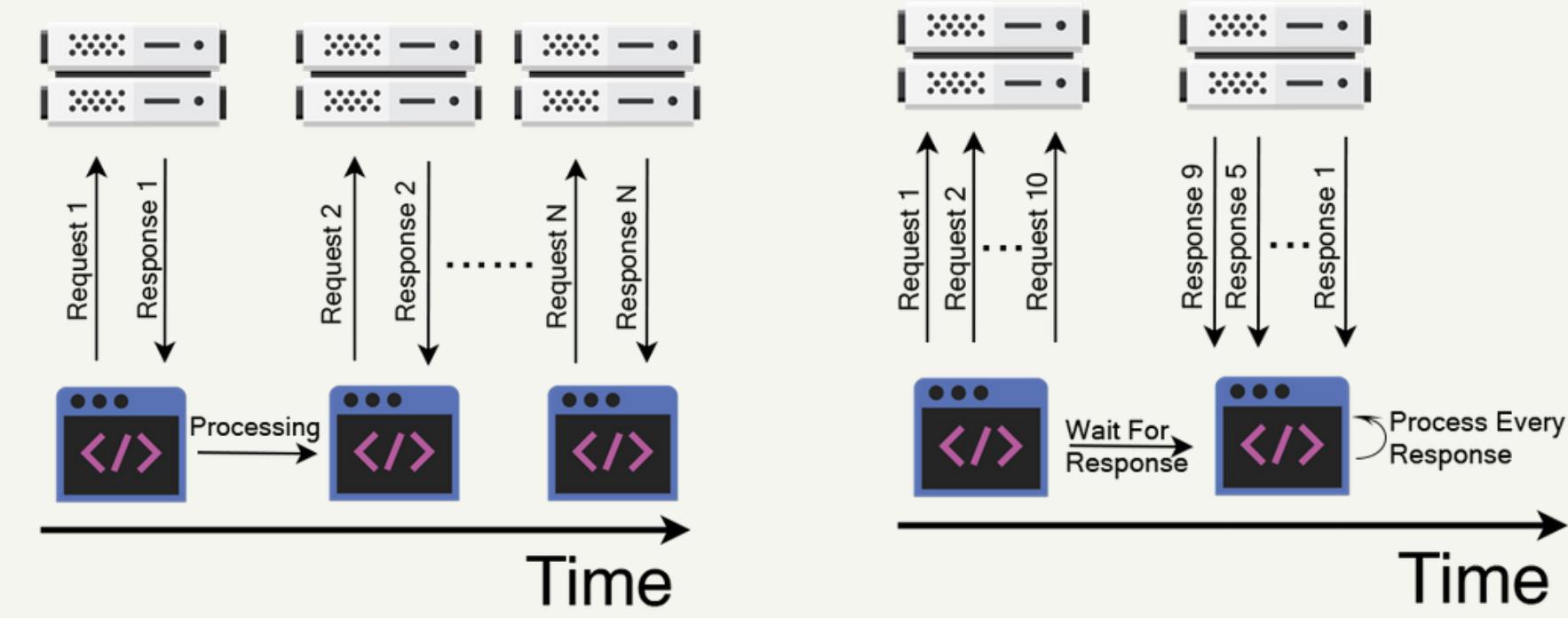


The algorithm finds the minimum distance between a point and a segment, using orthogonal projection if possible

Let  $n$  be the number of points,

$$\operatorname{argmin}_{i \in \{1, \dots, n-1\}} \left( \frac{\|(s_{i+1} - s_i) \wedge (s_i - A')\|}{\|s_{i+1} - s_i\|} \right)$$

# Optimization of the requests: supercharge requests python library with custom methods

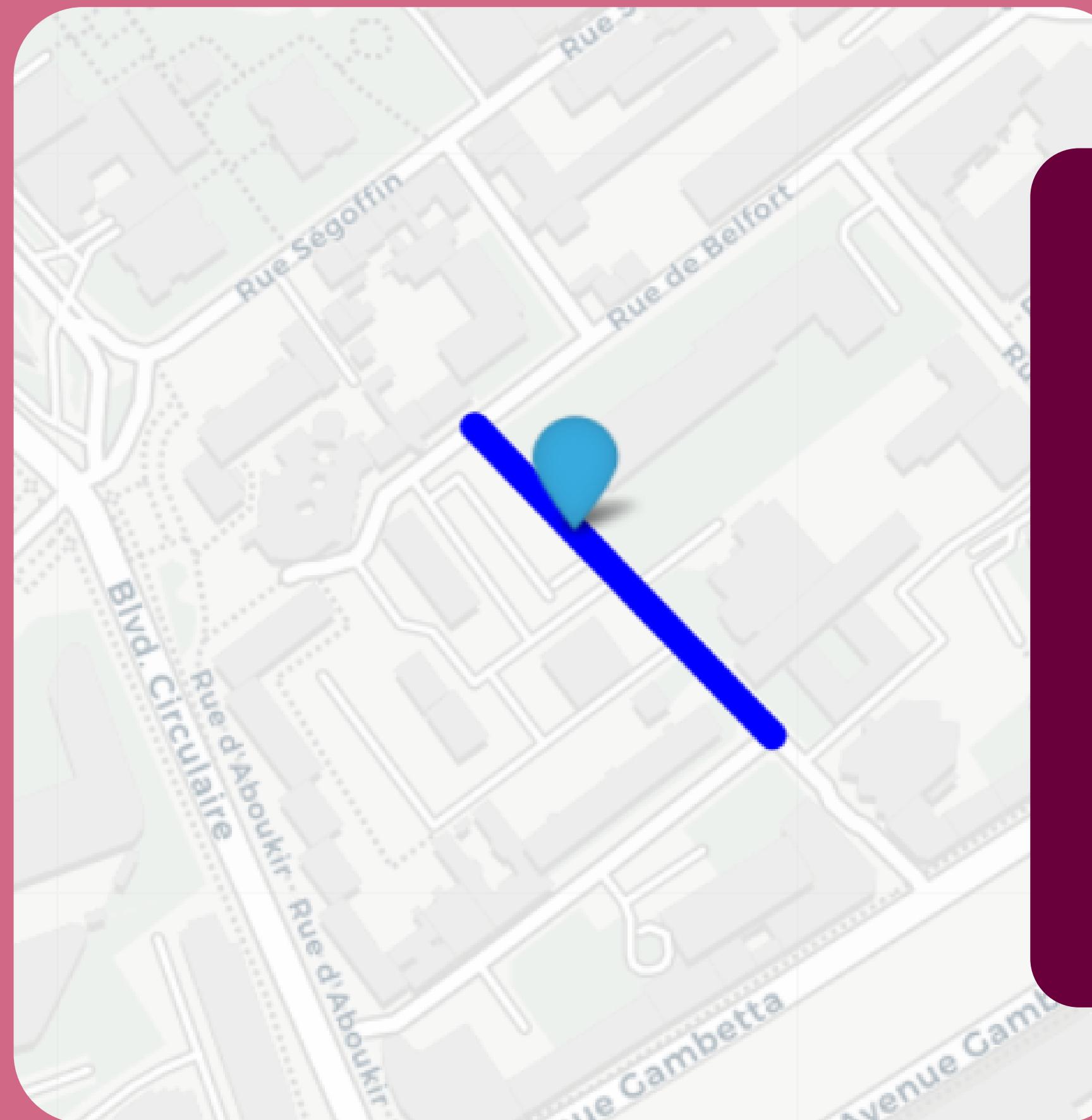


## Async

Several API requests at once, continue the algorithm when possible while waiting for the responses.

## Cache system

Store previous requests in a python dictionary (saved afterward in a raw file) to faster process.



# Data Visualization

We created an algorithm to have a efficient visualization of our result susing the Folium library

It marks the given point and highlights the related road section.

# Our solution

Algorithm : python package.  
Web server with python backend.  
Deployable on any app platform such as Heroku,  
debian server...

# Our interface

We create an interface that enables the user to easily enter coordinates by hand or using a .csv file

It returns the visualization, a table with the result and a sentence in French that gives the location of the tree

EPSILON TECHNICAL SOLUTIONS

## One Object Solution

Enter the coordinates of your object to see the description of the road section which include your object. You can enter multiple objects to see the description of the road section for each. You can also upload a csv file with all of your objects coordinates in it.

The interface includes the following elements:

- A red circular button with a white plus sign (+) to add new objects.
- Input fields for "latitude" and "longitude".
- A red "RUN" button with a white play icon.
- A red "CSV FILE" button with a white file icon.
- Icons below the buttons: a double arrow for comparison, a chart for data visualization, and a checkmark for industry standard code.
- Text at the bottom left: "We built an API on python which relies on the Overpass API. The Basemap used is Openstreetmap. The".
- Text at the bottom center: "Built with Python".
- Text at the bottom right: "Data visualisation included".
- Text at the bottom far right: "Industry standard code".

# Thank you!



EPSILON TECHNICAL SOLUTIONS

Check our Github  
repository

