Report for the Web datamining and semantics project

by TRAN-HONG Nicolas (A4 DIA 1)

Objective : Make a Web application that integrates geospatial data from multiple sources, including dynamic data.

I. Choice of datasets for static data

My goal is to make a web application where users can enter a location (city, address) and display different buildings around them on a map.

I choose these 3 datasets, from open data source for my static data, which contains data of different buildings.

Higher Education Libraries

(https://www.data.gouv.fr/fr/datasets/bibliotheques-de-lenseignement-superieur/#_)

Postal Contact Points (Merchant Post Relays, Post Offices and Postal Agencies)

(https://data.iledefrance.fr/explore/dataset/les_bureaux_de_poste_et_agences_postales_en_idf)

Public Libraries

(https://data.culturecommunication.gouv.fr/explore/dataset/adresses-des-bibliotheques-publiques)

II. Ontology on Protégé

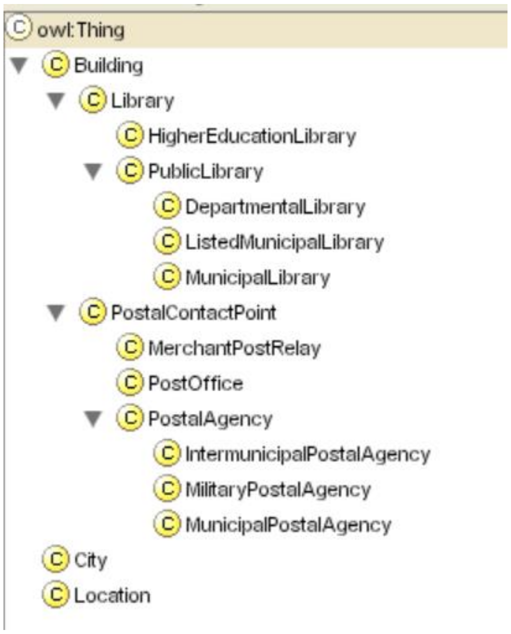
The data from previous datasets have to be stored on a RDF triple store which is a purpose-built database for the storage and retrieval of triples through semantic queries.

I then need to define an ontology, that will describes the types of entities that will be stored.

On Protégé, I will specify classes and properties to build the ontology.

- The first step is to design classes and subclasses of the ontology

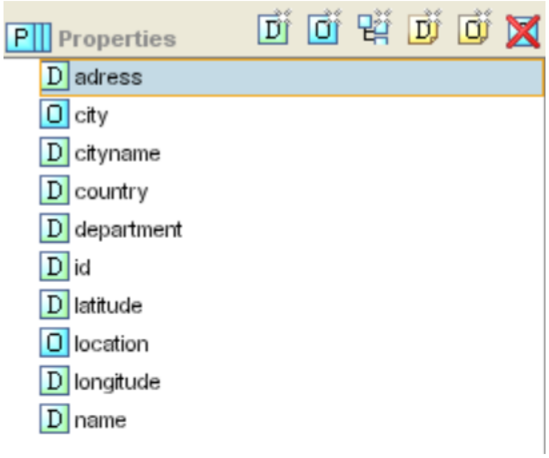
One of the functionality of my web application will be the possibility to choose a type of building we want to see on the map. That's why I need to design multiple sub-classes of building.



- owl:Thing
  - Building
    - Library
      - HigherEducationLibrary
      - PublicLibrary
        - DepartmentalLibrary
        - ListedMunicipalLibrary
        - MunicipalLibrary
    - PostalContactPoint
      - MerchantPostRelay
      - PostOffice
      - PostalAgency
        - IntermunicipalPostalAgency
        - MilitaryPostalAgency
        - MunicipalPostalAgency
  - City
  - Location

- Then we create datatype properties which are single properties (functional) and object properties with their domain and range.

These properties will be the useful buildings' information that we will get for our web application.

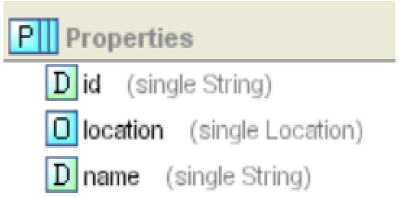


Properties

D adress
O city
D cityname
D country
D department
D id
D latitude
O location
D longitude
D name

Here are the properties for the different classes :

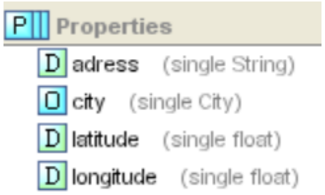
Building



Properties

D id (single String)
O location (single Location)
D name (single String)

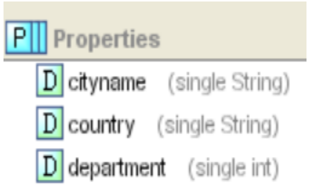
Location



Properties

D adress (single String)
O city (single City)
D latitude (single float)
D longitude (single float)

City



Properties

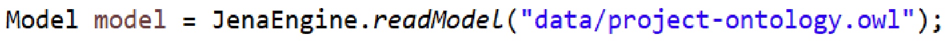
D cityname (single String)
D country (single String)
D department (single int)

III. Create the RDF file

Now, I need to convert static data into RDF so it can be loaded on the triple store.

I will generate RDF with **Apache Jena** in Java (with Eclipse). And I will generate all the data at once in a large Jena Model and serialize it as RDF at the end.

- The Jena model is built thanks to our previous ontology.



```
Model model = JenaEngine.readModel("data/project-ontology.owl");
```

- Then, I need to parse the 3 JSON files, that corresponds to the 3 datasets.
  For each row of the 3 datasets, I get the fields values that correspond to the ontology's properties and add them to the Jena Model.
  Some data cleaning is needed to obtain homogeneous data.

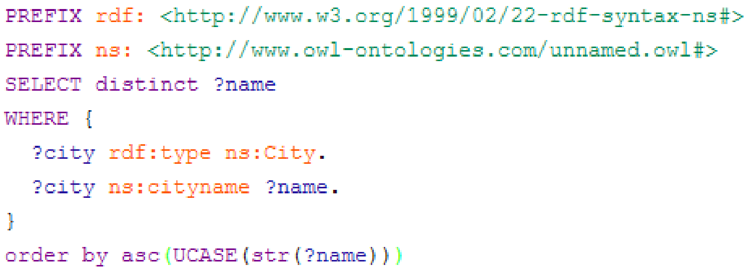
- After getting all the data, I can generate the **Turtle** file.

IV. Triple store

I load my ttl file manually to the triple store **Apache Jena Fuseki**.

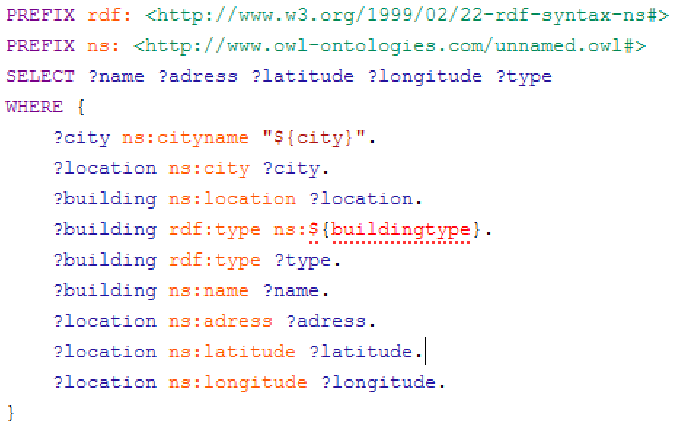
Below will be the SPARQL queries that I will use for the web application.

- Users will be able to select a city so I needed to get all the cities of my static database



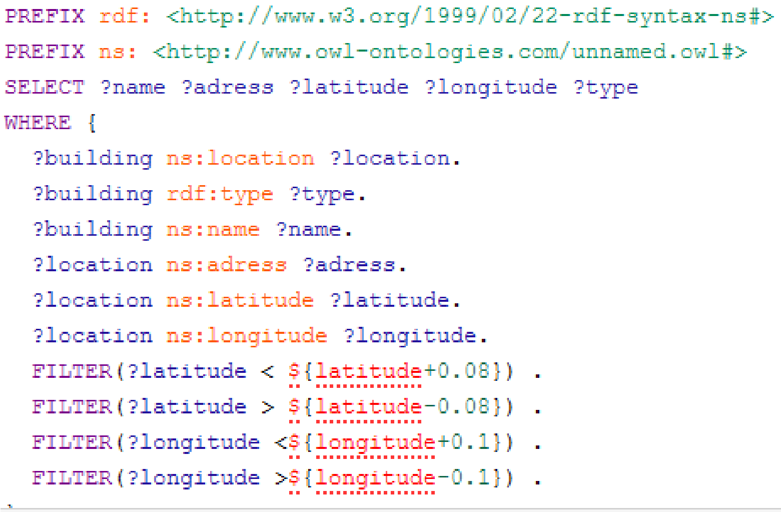
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ns: <http://www.owl-ontologies.com/unnamed.owl#>
SELECT distinct ?name
WHERE {
  ?city rdf:type ns:City.
  ?city ns:cityname ?name.
}
order by asc(UCASE(str(?name)))
```

- Users will be able to display all buildings (of a certain type) for a city



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ns: <http://www.owl-ontologies.com/unnamed.owl#>
SELECT ?name ?adress ?latitude ?longitude ?type
WHERE {
    ?city ns:cityname "${city}".
    ?location ns:city ?city.
    ?building ns:location ?location.
    ?building rdf:type ns:${buildingtype}.
    ?building rdf:type ?type.
    ?building ns:name ?name.
    ?location ns:adress ?adress.
    ?location ns:latitude ?latitude.
    ?location ns:longitude ?longitude.
}
```

- Users will be able to display all the buildings around (nearly 10 kms) a point on the map (defined by a latitude and a longitude)



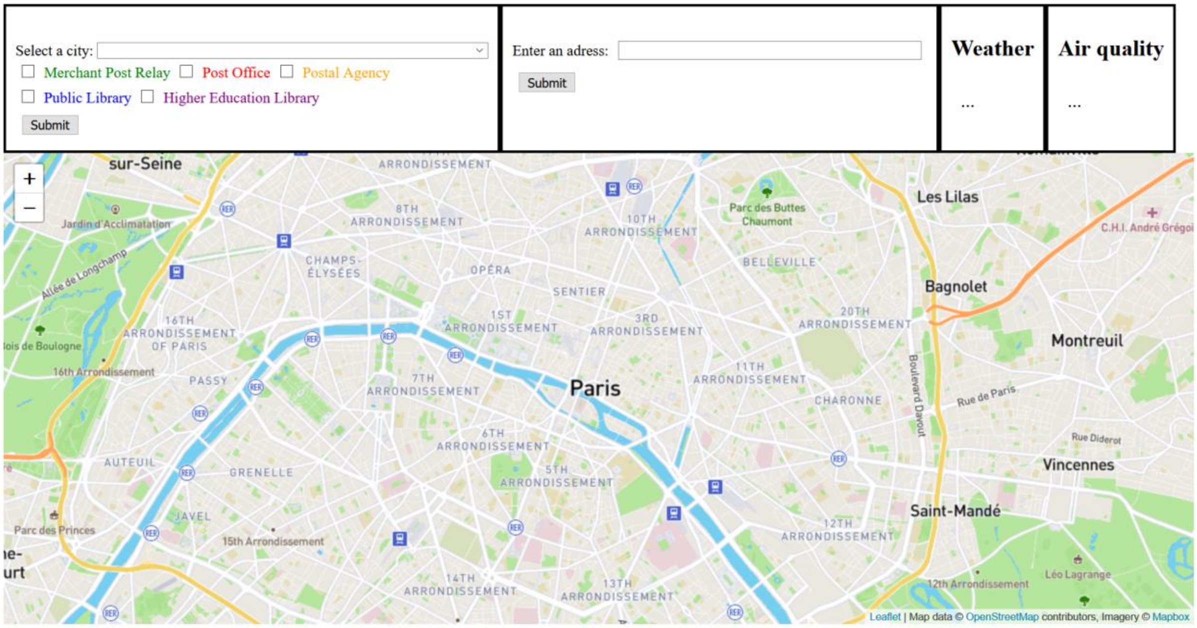
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ns: <http://www.owl-ontologies.com/unnamed.owl#>
SELECT ?name ?adress ?latitude ?longitude ?type
WHERE {
  ?building ns:location ?location.
  ?building rdf:type ?type.
  ?building ns:name ?name.
  ?location ns:adress ?adress.
  ?location ns:latitude ?latitude.
  ?location ns:longitude ?longitude.
  FILTER(?latitude < ${latitude+0.08}) .
  FILTER(?latitude > ${latitude-0.08}) .
  FILTER(?longitude <${longitude+0.1}) .
  FILTER(?longitude >${longitude-0.1}) .
```

V. Web Application

In this part, I will present the different functionalities of our web application, made via HTML, CSS and Javascript files.

To display a map, I use Leaflet (https://leafletjs.com/).

Here’s the interface of our web application.



Select a city:
□ Merchant Post Relay □ Post Office □ Postal Agency
□ Public Library □ Higher Education Library
Submit

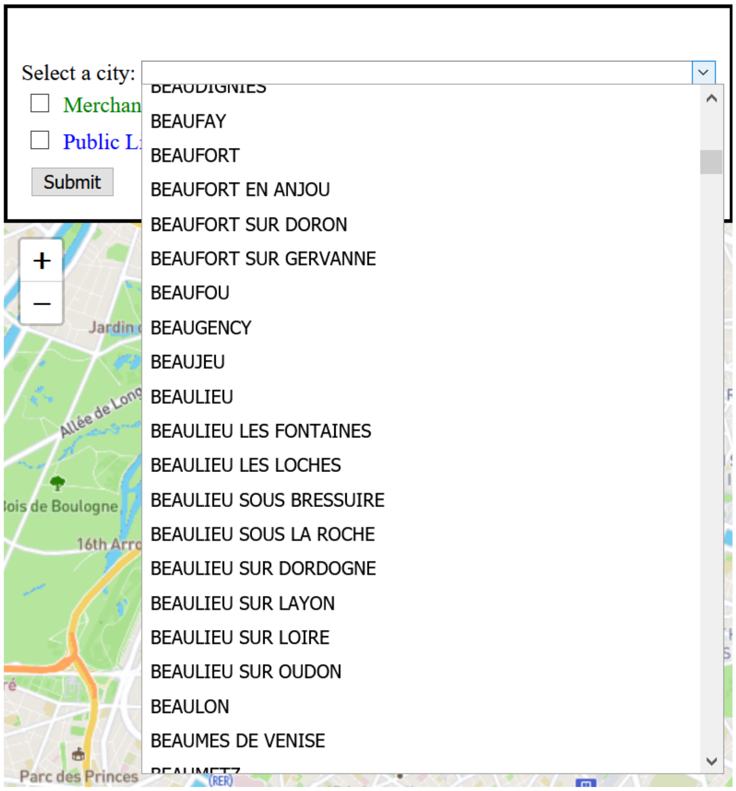
Enter an adress:
Submit

Weather
...

Air quality
...

- First Functionality : Select a city and the type of buildings you want to see on the map

Thanks to the a SPARQL query, Users can select a city among all cities available in our database.

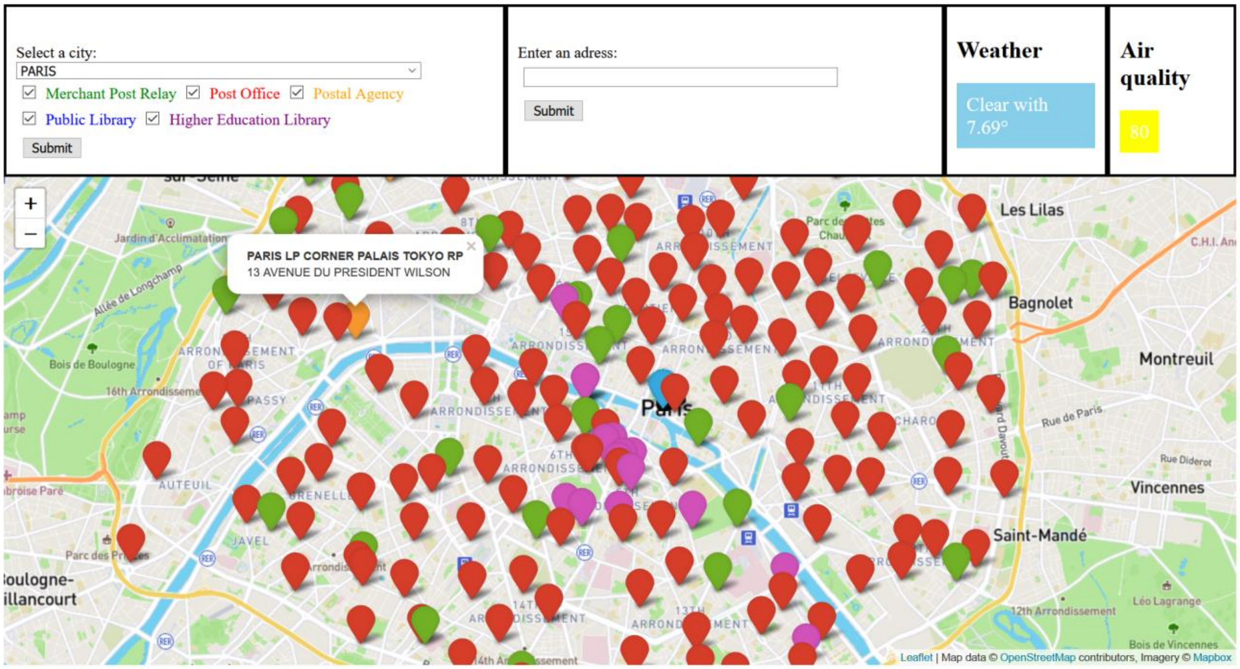


Select a city:

☐ Merchan

☐ Public L

Below, I select Paris and check all type of building :



Select a city: PARIS

☑ Merchant Post Relay ☑ Post Office ☑ Postal Agency

☑ Public Library ☑ Higher Education Library

Enter an adress:

Weather

Clear with 7.69°

Air quality

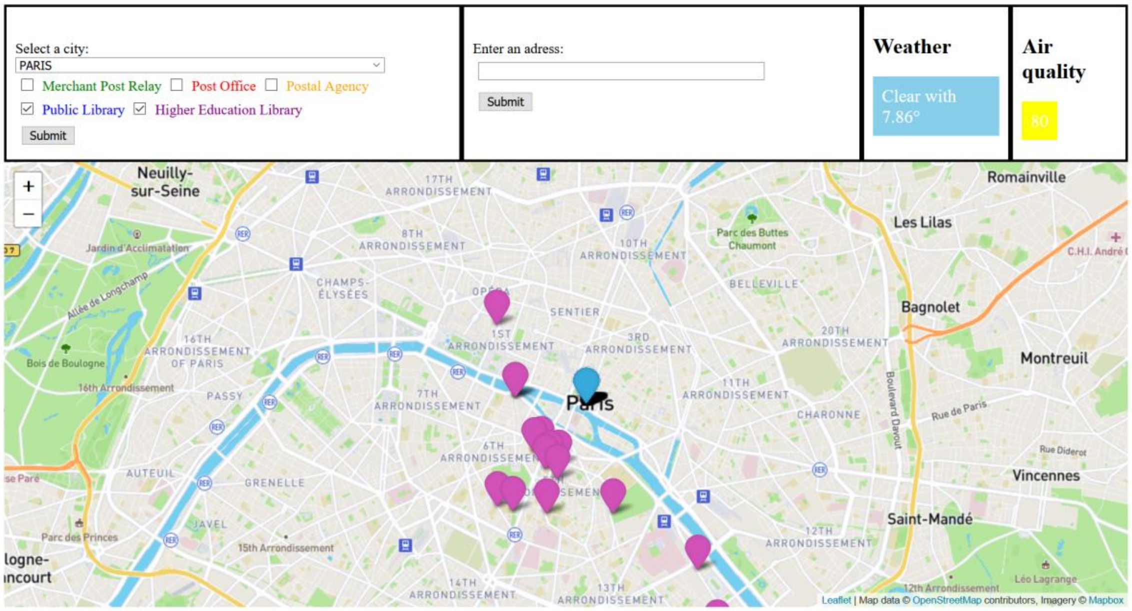
80

PARIS LP CORNER PALAIS TOKYO RP
13 AVENUE DU PRESIDENT WILSON

The map is zoomed to the city selected and all buildings in the city are then displayed on the map via a markers when you click on the "Submit" button. Each color represent a type of building : Blue for Public Libraries, Red for Post offices, Green for Merchant Post Relays, Yellow for Postal Agencies and Purple for Higher Education Libraries.

User can click on a marker to see the name and the address of a building.

I then keep "Paris" for the city and check only the libraries :



Select a city: PARIS

☐ Merchant Post Relay ☐ Post Office ☐ Postal Agency

☑ Public Library ☑ Higher Education Library

Enter an adress:

Weather

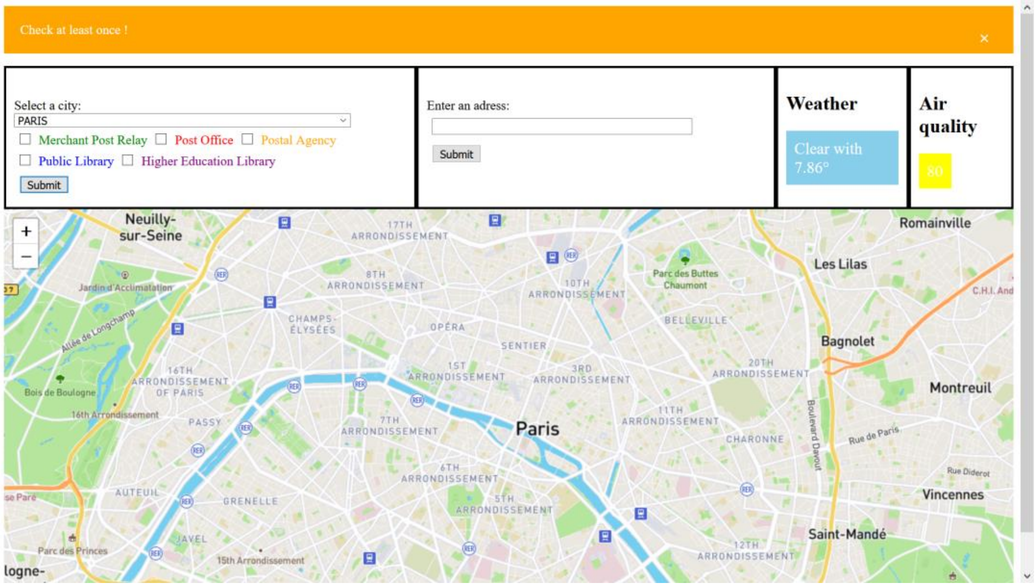
Clear with 7.86°

Air quality

80

Previous markers are removed when you click again on the submit button.

I you don't check at least one type of building, an alert appear at the top :



Check at least once !

Select a city: PARIS

☐ Merchant Post Relay ☐ Post Office ☐ Postal Agency

☐ Public Library ☐ Higher Education Library

Enter an adress:

Weather

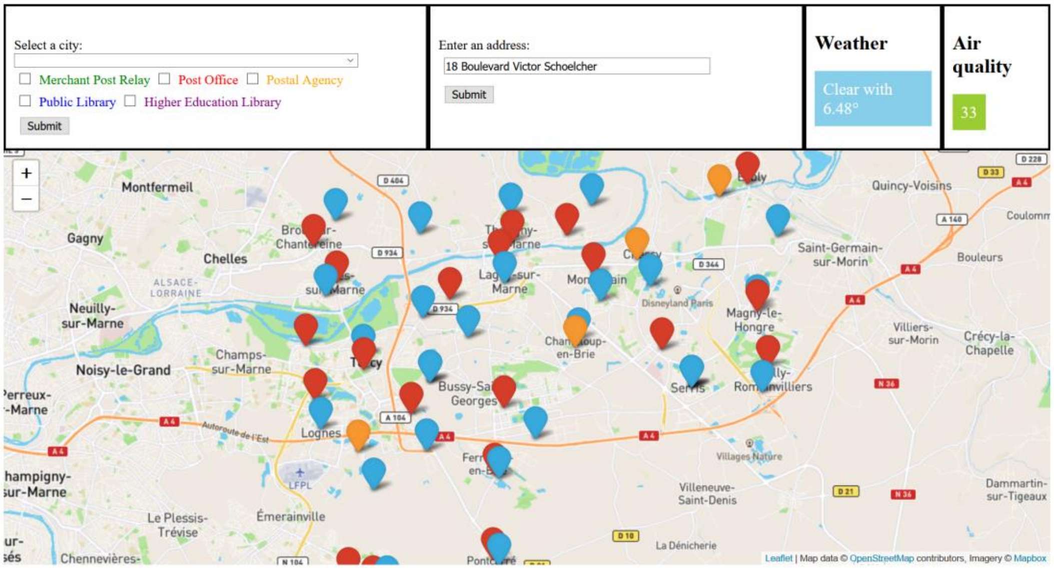
Clear with 7.86°

Air quality

80

- Second Functionality : Enter an address to display all buildings in a area of nearly 10 kms

Here I enter “18 Boulevard Victor Schoelcher” (which is in the city Busy-Saint-Georges) :



18 Boulevard Victor Schoelcher

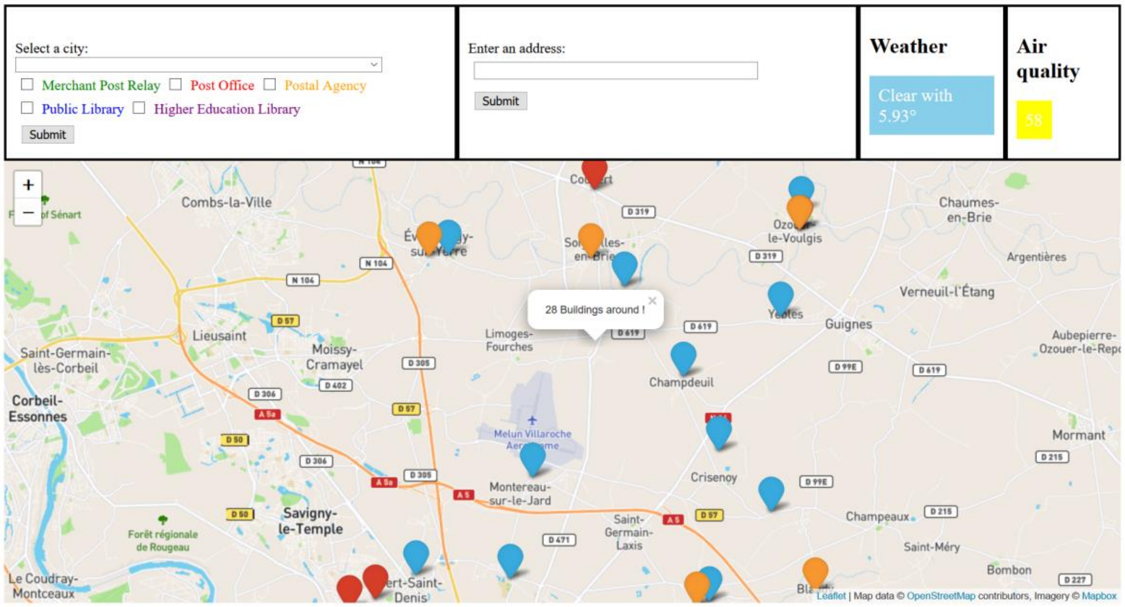
Clear with 6.48°

33

To get the coordinates of an address, I use the following API : https://geo.api.gouv.fr/adresse

- Third Functionality : Click on the map to display all buildings in a area of nearly 10 kms

Here I clicked randomly on the map.



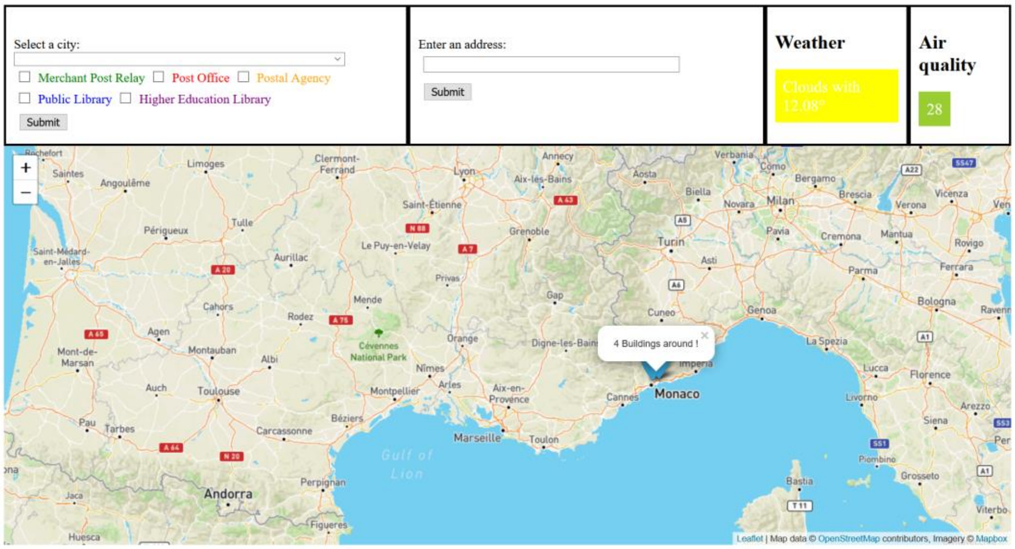
Clear with 5.93°

58

28 Buildings around !

A pop-up appears where you clicked and displays the number of buildings around.

- <u>Other Functionalities :</u> The weather and the air quality of the location is displayed for all other functionalities.



Select a city:

□ Merchant Post Relay □ Post Office □ Postal Agency

□ Public Library □ Higher Education Library

Submit

Enter an address:

Submit

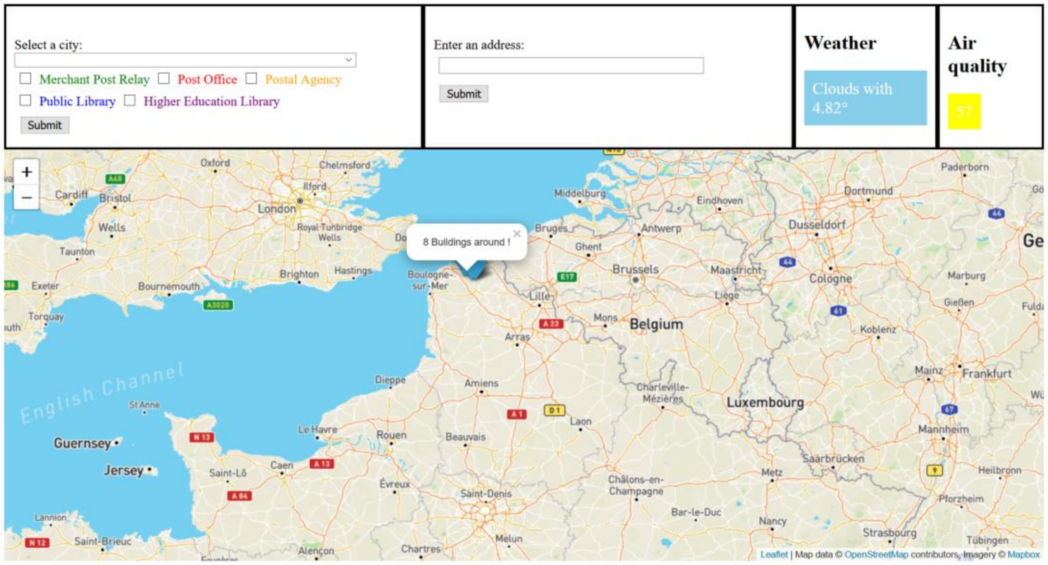
Weather

Clouds with 12.08°

Air quality

28

4 Buildings around !



Select a city:

□ Merchant Post Relay □ Post Office □ Postal Agency

□ Public Library □ Higher Education Library

Submit

Enter an address:

Submit

Weather

Clouds with 4.82°

Air quality

57

8 Buildings around !

2 color codes are used to represent the different levels of air quality and temperature.

I used two API to collect these information:

- https://aqicn.org/api/ for the Air quality
- https://openweathermap.org/current#other for the weather

Finally, here a video showcasing this web application :

https://www.youtube.com/watch?v=95uwnvcAIEU