



Backend developer technical test

Context

papernest provides multiple eligibility APIs internally and to its partner to better choose which contracts the client needs.

You want to provide an API to real estate agencies that will allow them to get the network coverage for all the homes they are renting.

Goal

Build a small api project that can request addresses and retrieve 2G/3G/4G network coverage for each available operator on each address.

A position is considered covered if there is a tower with the available technology in a specific radius.

This radius depends on the technology :

2G : 30km 3G : 5km 4G : 10km

Instructions

- Implement a route that takes multiples locations and returns appropriated results for them.
- Use Python and the framework of your choice

- You must use async
- How you manage the data sources is up to you. Do as you want (you can use other data sources if you want).
 - If you transform the CSV file with some offline processing, please provide the source file.
- The api interface (payload format) can be changed if you want.

Bonus

- Add logs to allow for monitoring of error rates
- Add unit tests

Submission Guidelines:

- Apart from the instructions above, you have the choice to use **what you want**, but everything you do should be logical and documented. It is the moment to show us your best practices :)
- Provide a link to a GitHub repository with your completed application.
- Include a README file with instructions on how to set up and run the application.
- Ensure all dependencies are listed and the app can run smoothly without errors

Example


POST :

```
{ "id1" : "157 boulevard Mac Donald 75019 Paris", "id4" : "5 avenue  
Anatole France 75007 Paris", "id5" : "1 Bd de Parc, 77700 Coupvray",  
"id6" : "Place d'Armes, 78000 Versailles", "id7" : "17 Rue René  
Cassin, 51430 Bezannes", "id8" : "78 Le Poujol, 30125 L'Estréchure"  
}
```

Response example:

```
{ "id1" : { "orange": {"2G": true, "3G": true, "4G": false}, "SFR": {"2G": true, "3G": true, "4G": true}, "bouygues": {"2G": true, "3G": true, "4G": false} }, "id4" : { "orange": {"2G": true, "3G": true, "4G": false}, "bouygues": {"2G": true, "3G": false, "4G": false}, "SFR": {"2G": true, "3G": true, "4G": false} } ... }
```

Data that you can use

 2018_01_Sites_mobiles_2G_3G_4G_France_metropolitaine_L93_ver... 2139.4...

The file above provides a list of network coverage measure. Each line have the provider, Lambert93 geographic coordinate (X, Y) and network coverage for 2G, 3G and 4G

<https://adresse.data.gouv.fr/api> This API allow you to retrieve :

- address detail from a query address (the insee code, geographic coordinates, etc.)
- Do reverse geographic search (from longitude and latitude, retrieve an address).

Appendix

Convert from Lambert 93 to GPS coordinates can be a pain. The following example code converts from the two coordinates system in Python using [pyproj](#) library. You can use another library to do the conversion if you wish.

```
import pyproj
def lamber93_to_gps(x, y):
    lambert = pyproj.Proj('+proj=lcc +lat_1=49 +lat_2=44 +lat_0=46.5 +lon_0=3 +x_0=700000 +y_0=6600000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs')
    wgs84 = pyproj.Proj('+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs')
    x = 102980
    y = 6847973
    long, lat = pyproj.transform(lambert, wgs84, x, y)
    return long, lat
```

