Geospatial Databases - Project

Prof. Dr. Agnès Voisard
Prof. Nicolas Lehmann

Schematic Public Transportation Maps

Gleb Shcheletskiy – 4921000
João Pedrosa – 5301331

Task 1

To start off the project, we set up the “GSDBOMM” database using PostgreSQL and extending it with PostGIS. Afterwards we imported all the data from Portugal into the database, as we are only going to be working with subway stations in Lisbon.

```
postgres=# \l
                          List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |
-----------+----------+----------+-------------+-------------+
 GSDBOMM   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
```

```
                   List of relations
  Schema  |          Name           | Type  |  Owner
----------+-------------------------+-------+----------
 public   | planet_osm_line         | table | postgres
 public   | planet_osm_line_tmp     | table | postgres
 public   | planet_osm_nodes        | table | postgres
                          ---
 topology | layer                   | table | postgres
 topology | topology                | table | postgres
(49 rows)
```

Task 2

In order to program our web service we decided to use the *JavaScript* programming language. To automatically configure and install the project, we used the *Node.js package manager* (*npm*).

Our first step to build the schematic map is to acquire all the data related to the subway stations and lines in Lisbon. To achieve this, we use the following query utilizing the *Overpass API*:

```
[out:json];
(
  node["station"="subway"](38.699223, -9.226499, 38.797711, -9.084250);
  relation["route"="subway"](38.699223, -9.226499, 38.797711, -9.084250);
);
out body;
>;
out skel qt;
```

The coordinates presented inside the parenthesis refer to the limits of Lisbon's metropolitan area. The "nodes" requested correspond to information about the subway stations, while the "relations" correspond information about the lines and how the stations are linked. All the data outputted by this query was stored in the file ***lisbon-metro-original.json***.

After obtaining this file, the *npm* script ***npm run process-data*** formats the data and stores it into ***lisbon-metro-formatted.json***. This creates a file that, for each station, stores its **id**, **name**, **label** and **position**, and for each line stores its **name**, **color** and **nodes**. These nodes hold the station's **names** and new **coordinates** that will allow us to represent them in a schematic map. These coordinates have to then be manually calculated and inserted. They are based on the original latitude and longitude of each station and set up in a way that every two consecutive stations can only form a 0º, 45º or 90º angle between them. After modifying the obtained file, we save the information into ***lisbon-metro.json***.

```
},
"Alvalade": {
  "id": 256971869,
  "name": "Alvalade",
  "label": "Alvalade",
  "position": {
    "lat": 38.753034,
    "lon": -9.1439777
  }
},
```

station

```
"name": "Verde",
"color": "#2F9B9C",
"shiftCoords": [0, 0],
"nodes": [
```

line

```
{
  "coords": [39, 21],
  "labelPos": "E",
  "name": "Alvalade"
},
```

node

The way in which we formatted and stored our data in the new file allows us to use **d3-tube-map** to obtain a schematic map in the visual style of London's Underground Map. With all the data and dependencies ready, we call our ***initMap()*** method to draw the schematic map in the webpage.

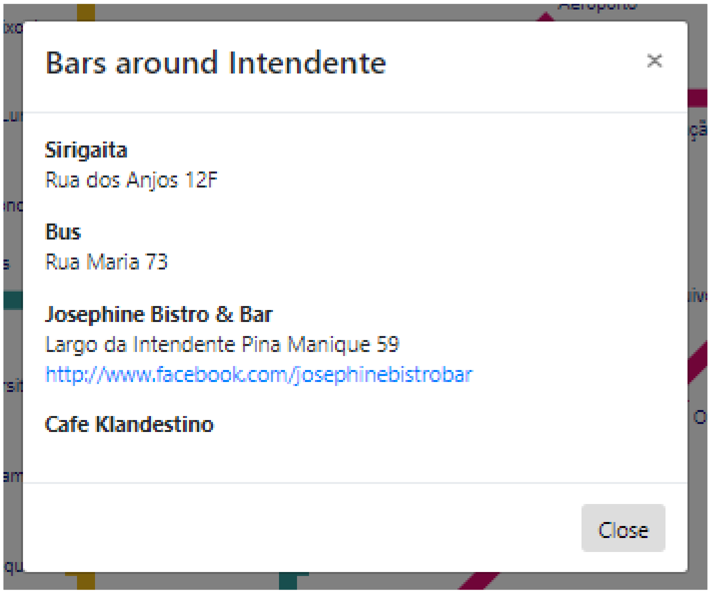
Gleb Shcheletskiy
João Pedrosa

Task 3

To enrich our schematic map we decided to add the functionality to list pubs/bars around each subway station. Every time one of the subway stations is selected, the method ***getBarsAround()*** is called. This method, takes the data for the selected station and uses it to execute the following query (again, using *Overpass API*):

```
[out:json];
node["amenity"="bar"](around:500,${lat},${lon});
out;
```

Here **lat** and **lon** refer to the latitude and longitude of the selected station and these values are fetched from the ***lisbon-metro.json*** file.
To display the information that we obtained, a pop-up appears in the webpage with a list of all the bars queried. For every bar obtained by this query, we list its **name**, **street address** and **website**, as long as they have this information available in *OSM*.



This information is fetched in real-time, so it should always be up-to-date, as long as *OSM* also is.

All files and documentation can be found in: https://github.com/JCanhao/GSDB.git