Slovenská technická univerzita
Fakulta informatiky a informačných technológii
Ilkovičova 2, 842 16 Bratislava

GIS webová aplikácia

Meno a priezvisko: Michal Knapík

Študijný program: Internetové technológie

Akademický rok: 2018/2019

Predmet: Pokročilé databázové technológie

1. O aplikácii

Túto aplikáciu sme vypracovali v rámci semestrálneho projektu z predmetu Pokročilé databázové technológie. Ide o webovú aplikáciu ktorá vizualizuje geodáta pomocou Mapbox GL JS API[1]. Aplikácia ponúka jednoduché používateľské rozhranie na prácu s vybranými prípadmi použitia. Jedná sa o zobrazenie staníc metra od liniek metra vo vzdialenosti jedného kilometra. Ďalej o zobrazenie trestných činov v podobe heatmapy a následne o filtrovanie dát cez typ trestného činu, ponúkame možnosť aj tz. heatmapy s priebehom času za jednotlivé mesiace. Posledným je zobrazenie a následné ofarbenie oblastí podľa podielu počtu trestných činov ku veľkosti danej oblasti.

2. Práca s databázou a datasety

V našom projekte sme použili databázu PostgreSQL[2] vo verzií 10.5 rozšírenú o PostGis[3] vo verzií 2.5. Pre zhromaždené datasety bolo potrebné vytvoriť v schéme databázy tabuľky a niektoré dáta bolo potrebné aj upraviť do nami požadovaného tvaru, aby sa s nimi dalo pracovať.

Datasety

V aplikácii pracujeme s niekoľkými datasetmy, prvým sú OSM dáta pre Washington DC ktoré sme importovali do databázy cez nástroj **osm2pqsql**[4], a to pomocou nasledujúceho príkazu.

osm2pgsql -U postgres -W -H localhost -d pdt -E 4326 district-of-columbia-latest.osm

Ďalším datasetom s ktorým pracujeme je zoznam všetkých zastávok metra vo Washingtonskej metropolitnej oblasti, nakoľko dáta z OSM neboli postačujúce. Tento dataset sme si upravil (zredukovali) do nami požadovanej formy pomocou nami vytvpreného jednoduchého python skriptu:

```
#!/usr/bin/python3
import json
from pprint import pprint

with open('blue.txt') as f:
    data = json.load(f)

for r in data['Stations']:
    print(r['Code'] + ',', "" + r['Name'] + '",', str(r['LineCode1']) + ',', str(r['LineCode2']) + ',',
str(r['LineCode3']) + ',', str(r['LineCode4']) + ',',
        str(r['Lat']) + ',', r['Lon'])
```

Samotné dáta sme stiahli pomocou príkazu:

[1] https://www.mapbox.com/mapbox-gl-js/api/
[2] https://www.postgresql.org
[3] https://postgis.net
[4] https://wiki.openstreetmap.org/wiki/Osm2pgsql

*curl -v -X GET "https://api.wmata.com/Rail.svc/json/jStations?LineCode=**SV**" -H "api_key: e13626d03d8e4c03ac07f95541b3091b"*

Červená časť v príkaze hovorí o farebnom označení linky metra. Ďalšie možnosti sú: RD, YL, GR, BL, OR. Po spracovaní a zredukovaní duplikátov sme si vytvorili tabuľku a importovali dáta do databázy.

```
CREATE TABLE metro_stations
(
   id text NOT NULL,
   name text,
   line1 text,
   line2 text,
   line3 text,
   line4 text,
   lat double precision,
   lon double precision,
   primary key(id)
)
```

Po importrovaní dat sme potrebovali pridať geocolum stĺpec do tabuľky:

```
SELECT AddGeometryColumn ('public','metro_stations','geom',4326,'POINT',2);
UPDATE metro_stations SET geom = ST_SetSRID(ST_MakePoint(lon, lat), 4326);
```

Posledným datasetom sú dáta o trestných činoch z roku 2016 pre Washington DC. Tieto nebolo potrebné upravovať, stačilo ich iba importovať do nami vytvorenej tabuľky.

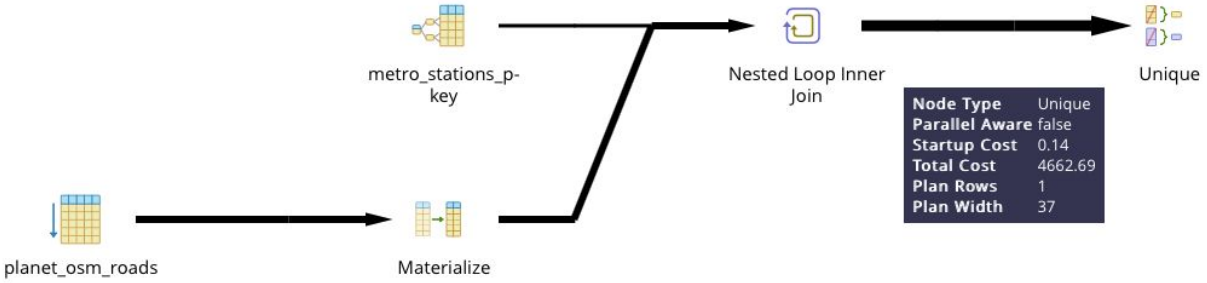
```
CREATE TABLE crime_incidents(
        X double precision,
        Y double precision,
        OBJECTID bigint,
        CCN bigint,
        REPORTDATETIME timestamp,SHIFT text,
        OFFENSE text,
        METHOD text,
        LASTMODIFIEDDATE timestamp,
        BLOCKSITEADDRESS text,
        BLOCKXCOORD double precision,
        BLOCKYCOORD double precision,
        WARD integer,
        ANC text,
        DISTRICT text,
        PSA integer,
        NEIGHBORHOODCLUSTER integer,
        BUSINESSIMPROVEMENTDISTRICT text,
        BLOCK_GROUP text,
        CENSUS_TRACT integer,
        VOTING_PRECINCT text,
        START_DATE timestamp,
        END_DATE timestamp,
        PRIMARY KEY(OBJECTID)
);
```

Queries

Pre získavanie dát z databázy používame nasledujúce dopyty.

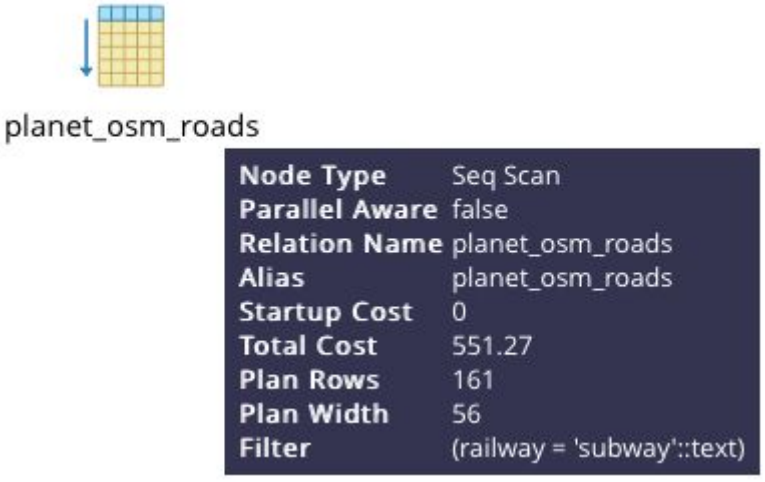
Prvý je určený na získanie všetkých zastávok metra vo vzdialenosti jedného kilometra od všetkých liniek metra:

```
SELECT DISTINCT ON (res.id) res.name, res.lat, res.lon FROM (
    SELECT poi.*, ST_DWithin(ST_Transform(road.way::geometry, 3857),
                             ST_Transform(poi.geom::geometry, 3857),
                             1000) AS in_dist
    FROM planet_osm_roads as road, metro_stations as poi
    WHERE road.railway LIKE '%subway%'
) AS res WHERE res.in_dist = true;
```



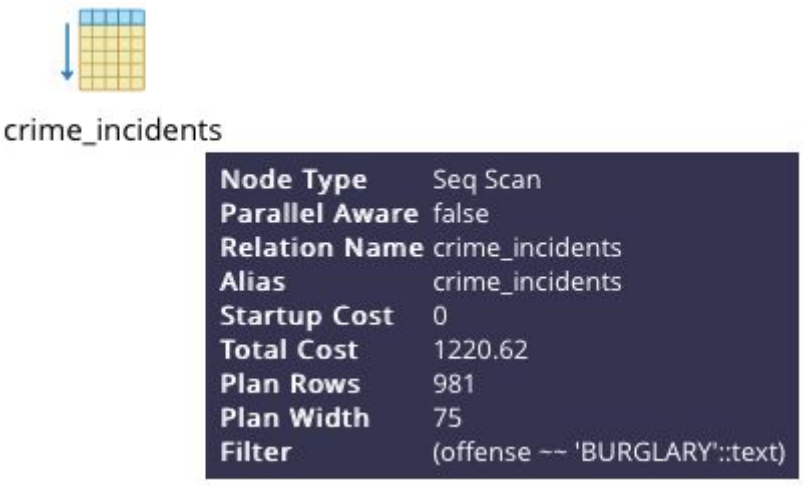
Pre ziskanie všetkých liniek (iba liniek) metra používame nasledujúci dopyt:

```
SELECT ST_AsText(way), name FROM planet_osm_roads WHERE railway = 'subway';
```



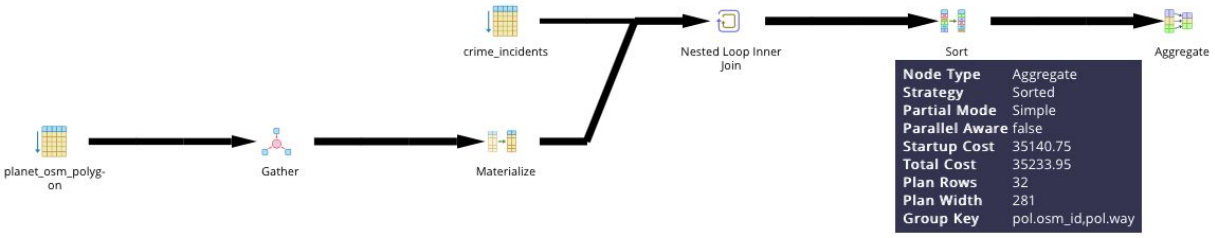
Pre získanie všetkých trestných činov podľa typu trestného činu používame nasledujúci dopyt, pričom %s predstavuje vstupný reťazec s typom zločinu:

```
SELECT y, x, reportdatetime, offense, method, to_char(start_date, 'MM') FROM
crime_incidents WHERE offense LIKE %s;
```



Posledným dopytom získame priemerný počet trestných činov v oblastiach, v pomere k veľkosti danej oblasti:

```
SELECT ST_AsText(res.way), count(res.*),
        ST_Area(geography(res.way)),
        count(res.*) / ST_Area(geography(res.way)) AS crime_per_size FROM (
                SELECT pol.osm_id, pol.way, ST_Contains(pol.way, ci.way) AS is_in
                FROM planet_osm_polygon as pol, crime_incidents as ci
                WHERE ci.offense LIKE %s AND (
                        pol.boundary = 'neighborhood' OR
                        pol.boundary = 'protected_area' OR
                        pol.boundary = 'suburb' OR pol.boundary = 'borough')
) AS res WHERE is_in = true GROUP BY res.osm_id, res.way;
```



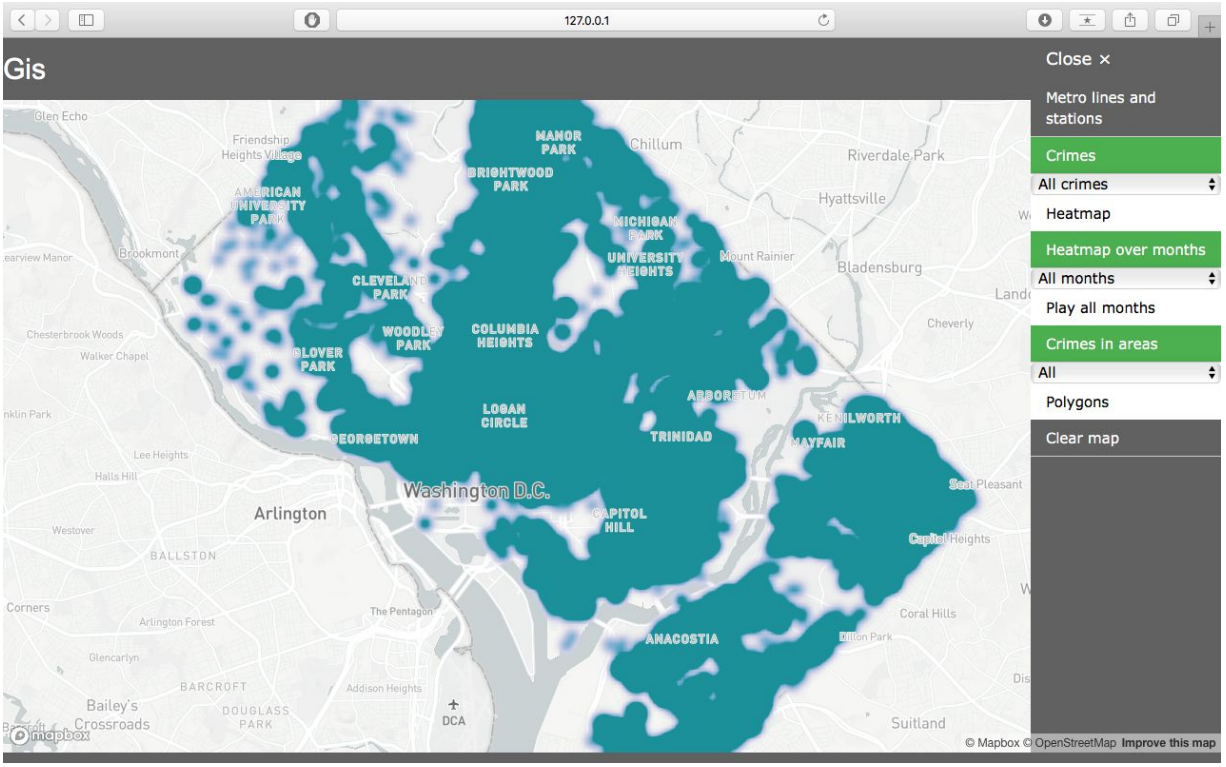
3. Frontend a backend

Aplákácia zodpovedá klasickej architektúre typu klient/server, kde serverová časť rieši prácu s databázou a klientská časť umožňuje prácu s používateľským rozhraním.

Frontend

Frontend-ová časť aplikácie je postavená klasicky na HTML5 (index.html) obohatená o W3schools CSS prvky. Celá klientska funkcionalita je naprogramovaná v JavaScripte, pomocou Mapbox GL JS API a jQuery.

V pravej časti sa nachádza menu pre prácu so stránkou. Jednoduché tlačidlá ponúkajú používateľovi možnosť interagovať s prípadmi použitia a pracovať tak so samotnou vizualizáciou dát.



Backend

Pre vytvorenie webového servera sme použili framework CherryPy[5] pre programovací jazyk Python, táto časť nášho riešenia pracuje aj so samotnou databázou a čiastočne predpripravuje dáta pre vizualizáciu v Mapbox API, tz. že pripravuje dáta vo formáte GeoJSON[6].

- webserver pristupuje k dátam pomocou ***psycopg2*** knižnice
- pomocou knižnice ***json*** pripravuje dáta do GeoJSON podoby
- predpripravené dáta sa posielajú na frontend

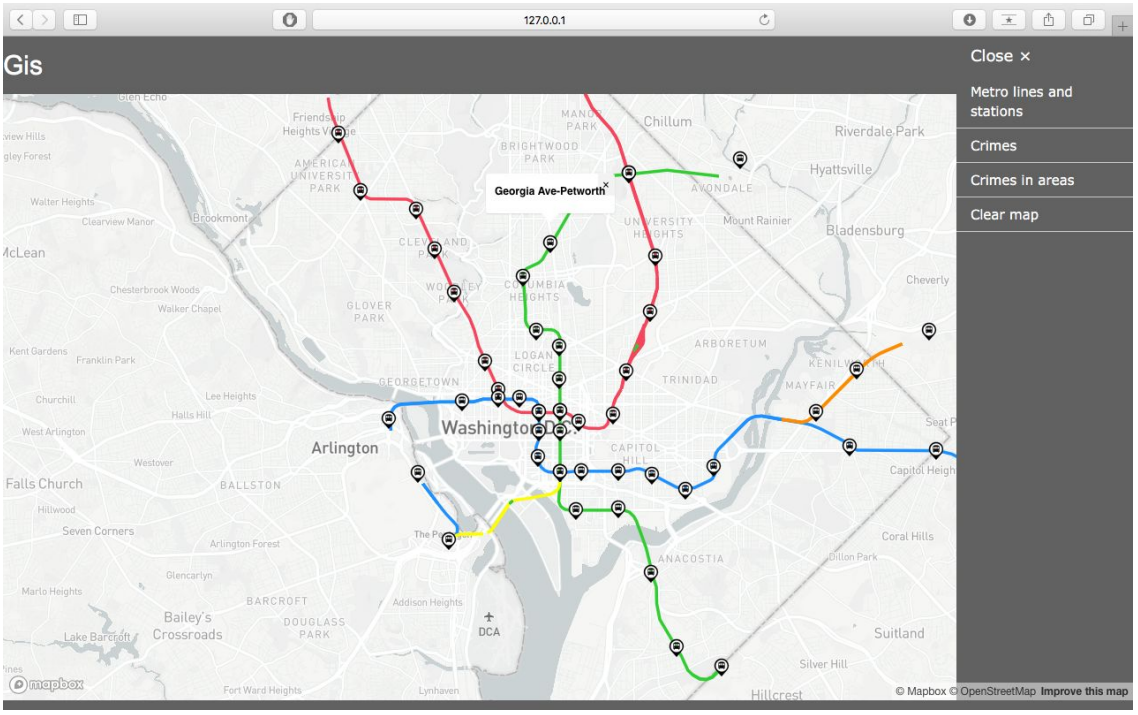
4. Služby

Dáta, ktoré vrátia služby sú v podobe GeoJSON-u, avšak klientská strana si ich musí ešte trochu upraviť do podoby použiteľnej pre Mapbox GL JS API.

Linky metra a jeho zástavky

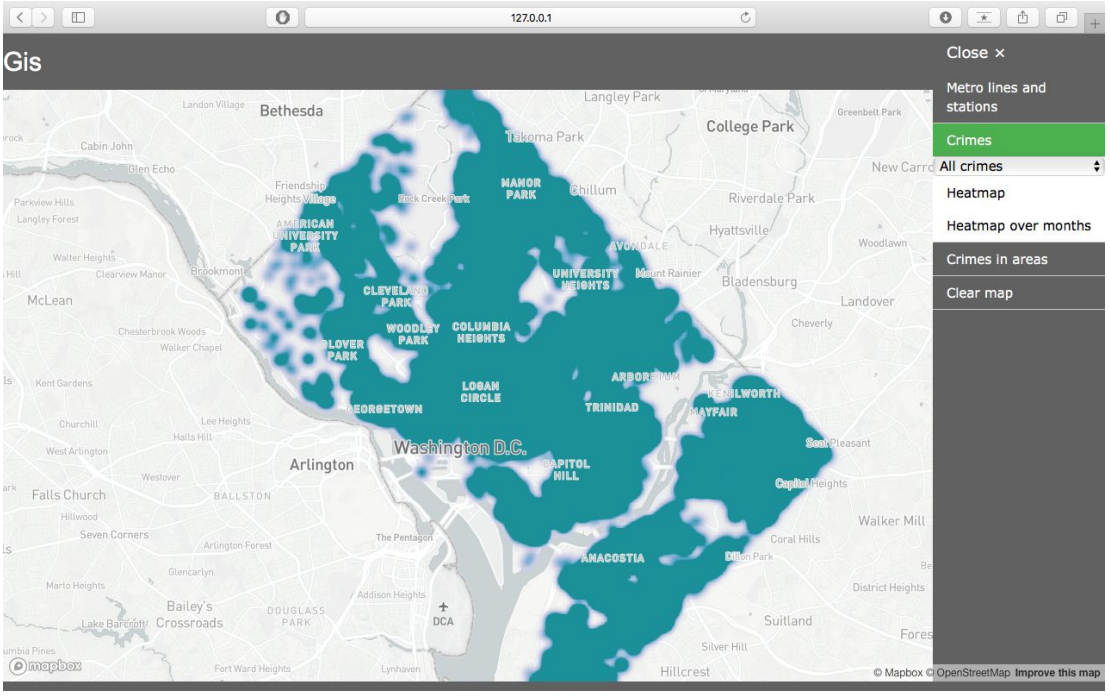
Táto služba vráti všetky linky metra a všetky zástavky metra vo vzdialenosti jedného kilometra od všetkých liniek, nakoľko naše datasety nie sú úplne rovnaké a linky metra nepokrývajú tak veľkú oblasť ako zastávky metra.

[5] https://cherrypy.org
[6] http://geojson.org

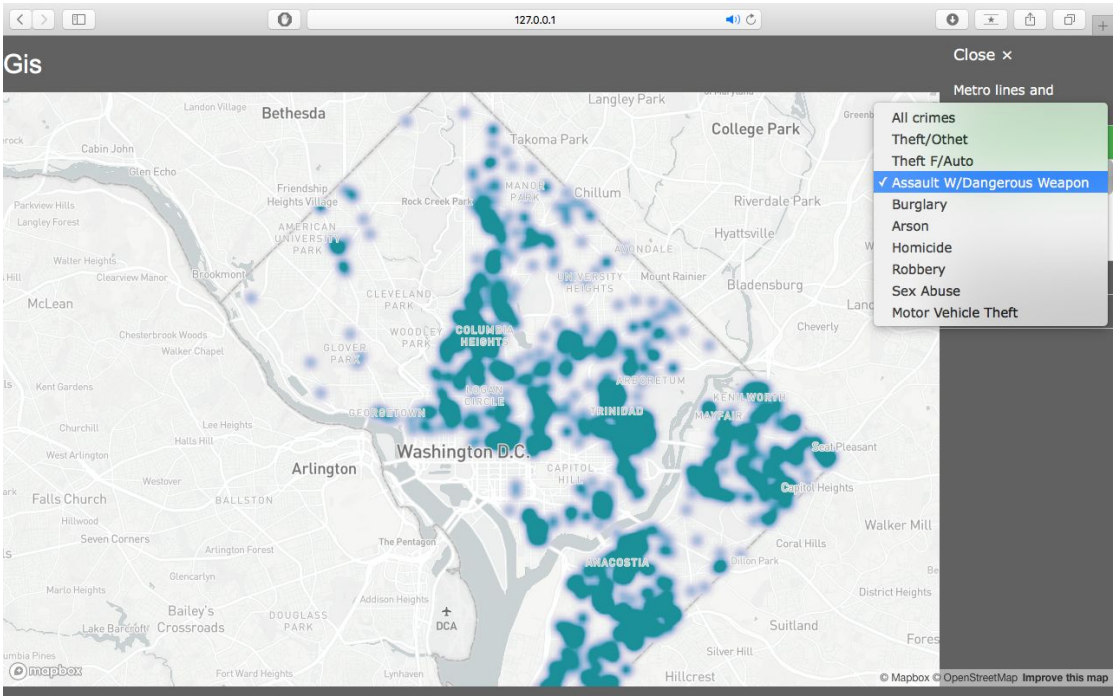


Trestné činy

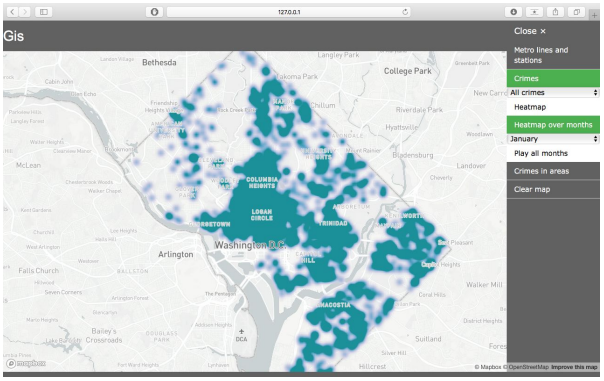
Táto služba je rozdelená na dve časti, prvá ponúka možnosť zobrazenia trestných činov v klasickej heatmape.

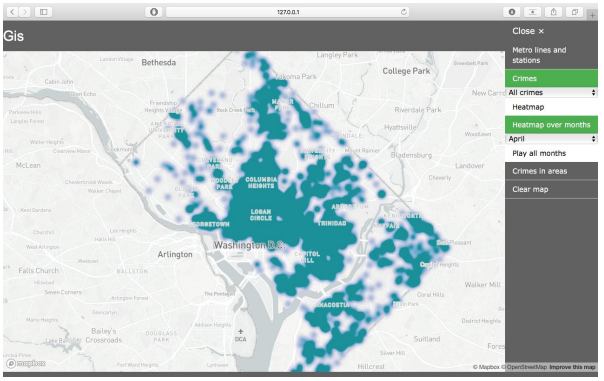


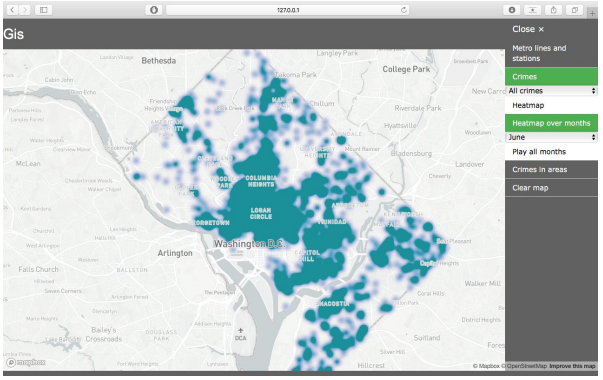
Prvá časť tiež umožňuje filtrovať vizualizáciu výsledku pomocou typu trestného činu.

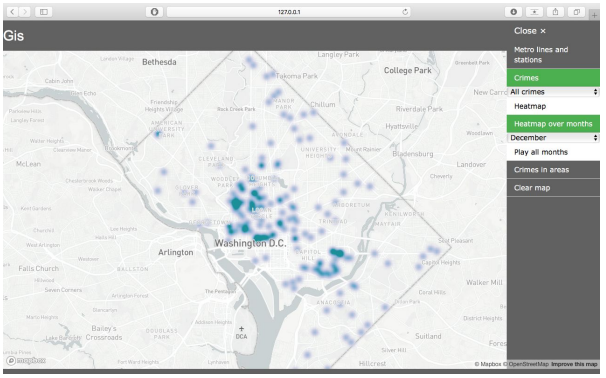


Druhá časť tejto služby ponúka vizualizáciu trestných činov v tz. heatmap over time, konkrétne za jednotlivé mesiace v roku. Umožňuje taktiež “prehrať” všetky mesiace vo vizualuzácií ako keby išlo o video, tz. od začiatku roka až po jeho koniec.



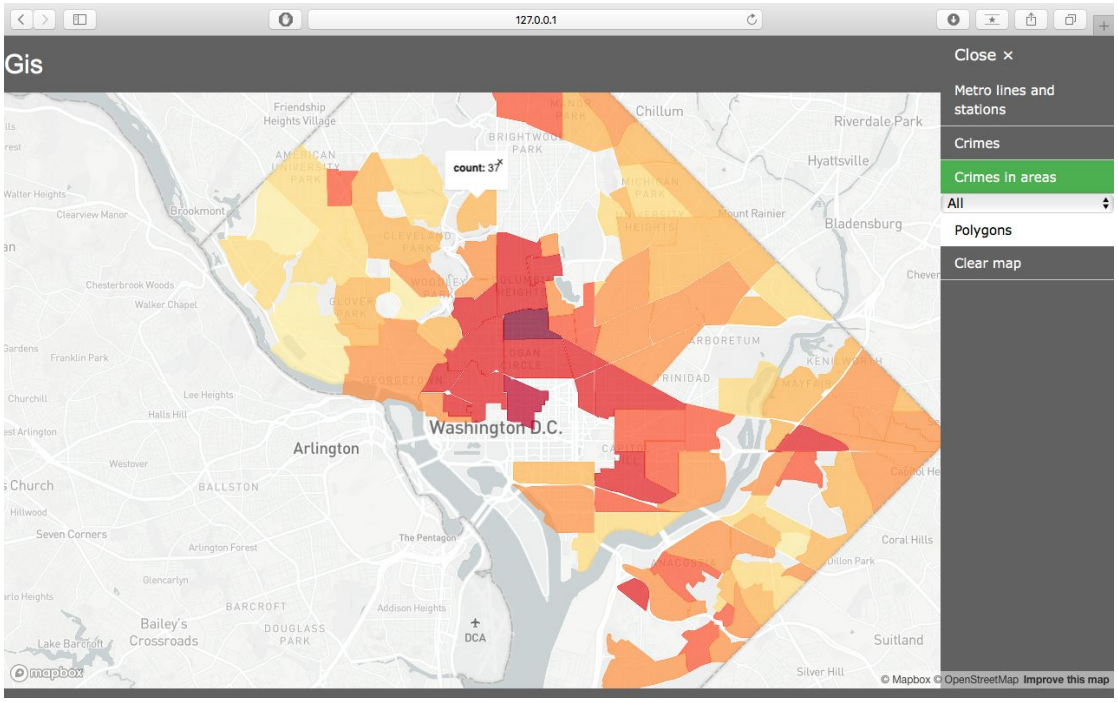






Trestné činy v oblastiach

Poslednou službou v našej aplikácii je možnosť zobrazenia početnosti trestných činov v závislosti od veľkosti oblasti v ktorej sa trestné činy uskutočnili.



Táto služba taktiež umožňuje filtrovať výsledok na základe typu trestných činov.