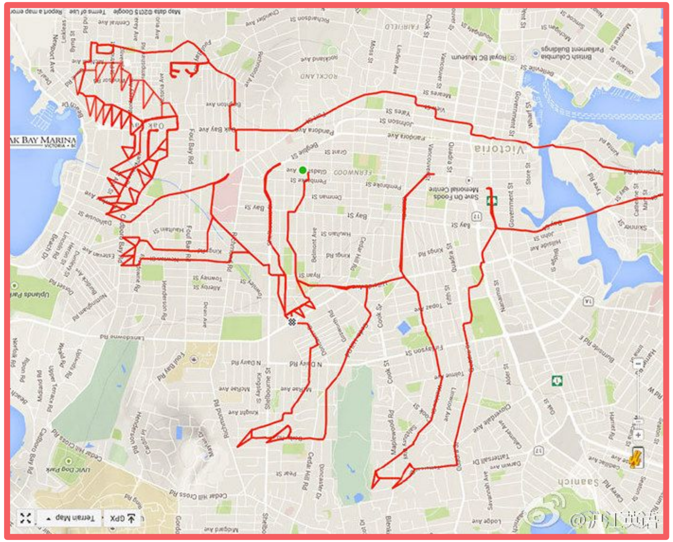
SCALABLE GEOFENCING

Elastic Search + Django

Kevin Sarmiento

GEOFENCING

- Virtual Perimeter (Geographically)
- Thrive Analytics
  - 60% of consumers use mobile devices to find info on local products/services.
  - 40% of them, on the go
- Examples
  - History Channel: Foursquare Check-ins
  - HoneyWell: Thermostat
  - Uber: LAX



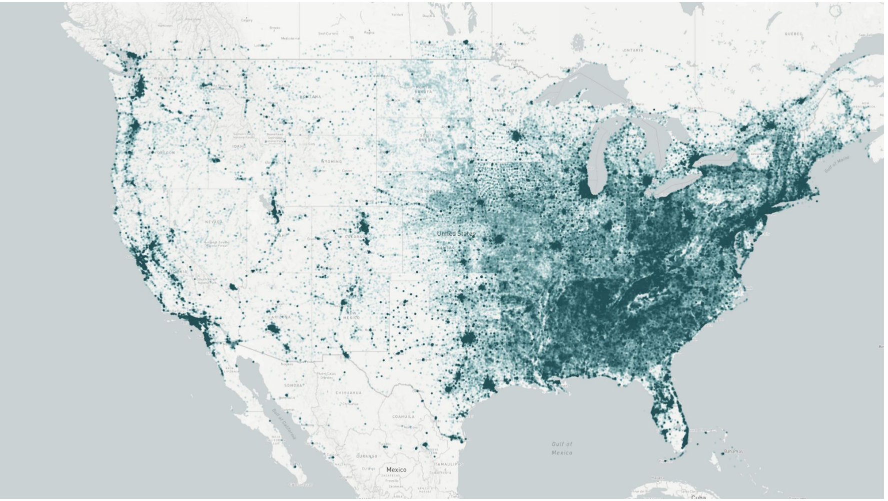
Computational Geometry

Point in Polygon (*PIP*)

GeoJSON: *points, lines, polygons*

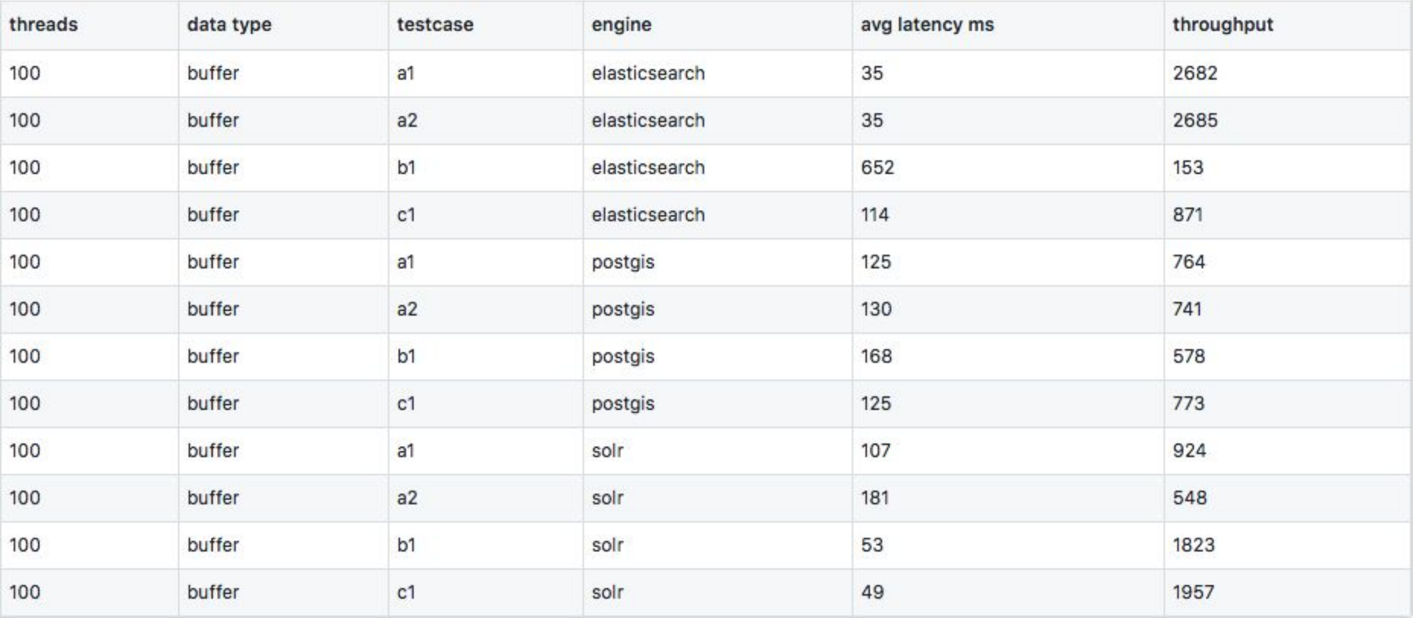
Geophy

“We are a real estate tech company, using data & machine learning to create instant and accurate valuations for commercial real estate”



- PostgreSQL, ElasticSearch, Solr
- ElasticSearch Results
  - Slow at indexing
  - Consumes considerable disk space with high precision polygons (<10m)
  - Worst performance for polygon in polygon
  - Best performance for point in simple polygon
  - SOLR and Elasticsearch are really good at handling tons of concurrent requests for simple spatial queries

GEOPHY



threads	data type	testcase	engine	avg latency ms	throughput
100	buffer	a1	elasticsearch	35	2682
100	buffer	a2	elasticsearch	35	2685
100	buffer	b1	elasticsearch	652	153
100	buffer	c1	elasticsearch	114	871
100	buffer	a1	postgis	125	764
100	buffer	a2	postgis	130	741
100	buffer	b1	postgis	168	578
100	buffer	c1	postgis	125	773
100	buffer	a1	solr	107	924
100	buffer	a2	solr	181	548
100	buffer	b1	solr	53	1823
100	buffer	c1	solr	49	1957

> *“Do the heavy lifting during data loading, so that retrieval and usage are fast and lightweight.”*

ElasticSearch

“Elasticsearch Is Fast.
Really, Really Fast.”

- HTTP native
- Schema-free JSON documents
- Supports basic geospatial searches
- Scalable
- Resilient
- Flexibility

Curl

Java

C#

Python

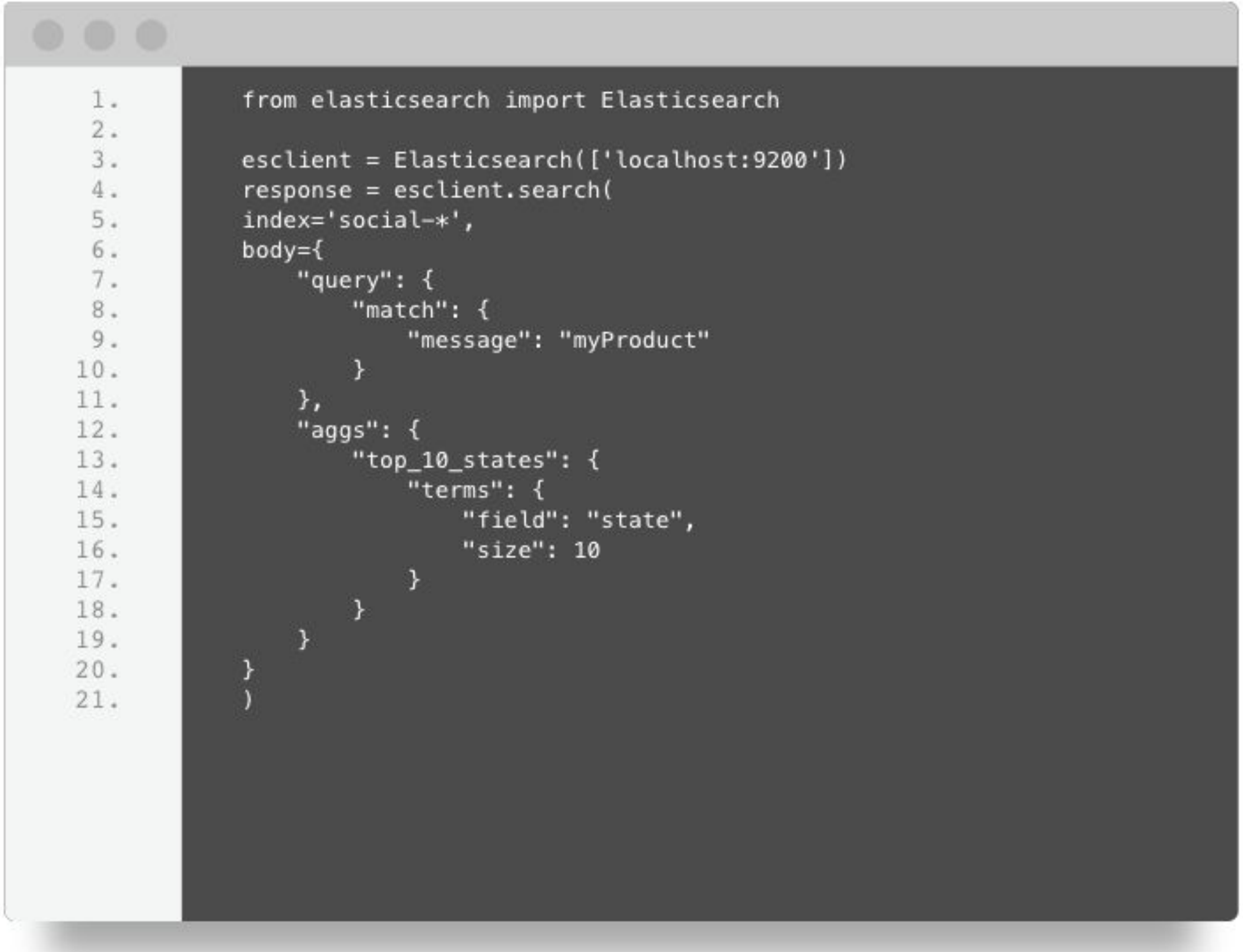
JavaScript

PHP

Perl

Ruby

SQL

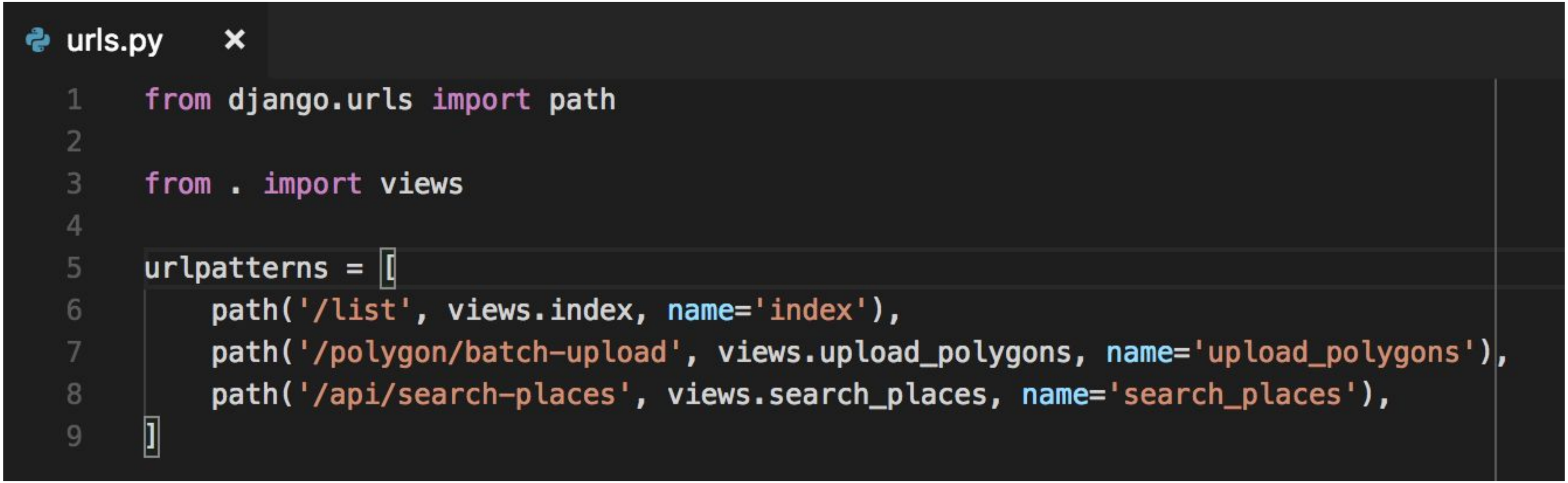


```
from elasticsearch import Elasticsearch

esclient = Elasticsearch(['localhost:9200'])
response = esclient.search(
index='social-*',
body={
    "query": {
        "match": {
            "message": "myProduct"
        }
    },
    "aggs": {
        "top_10_states": {
            "terms": {
                "field": "state",
                "size": 10
            }
        }
    }
}
)
```

Bates Hotel™

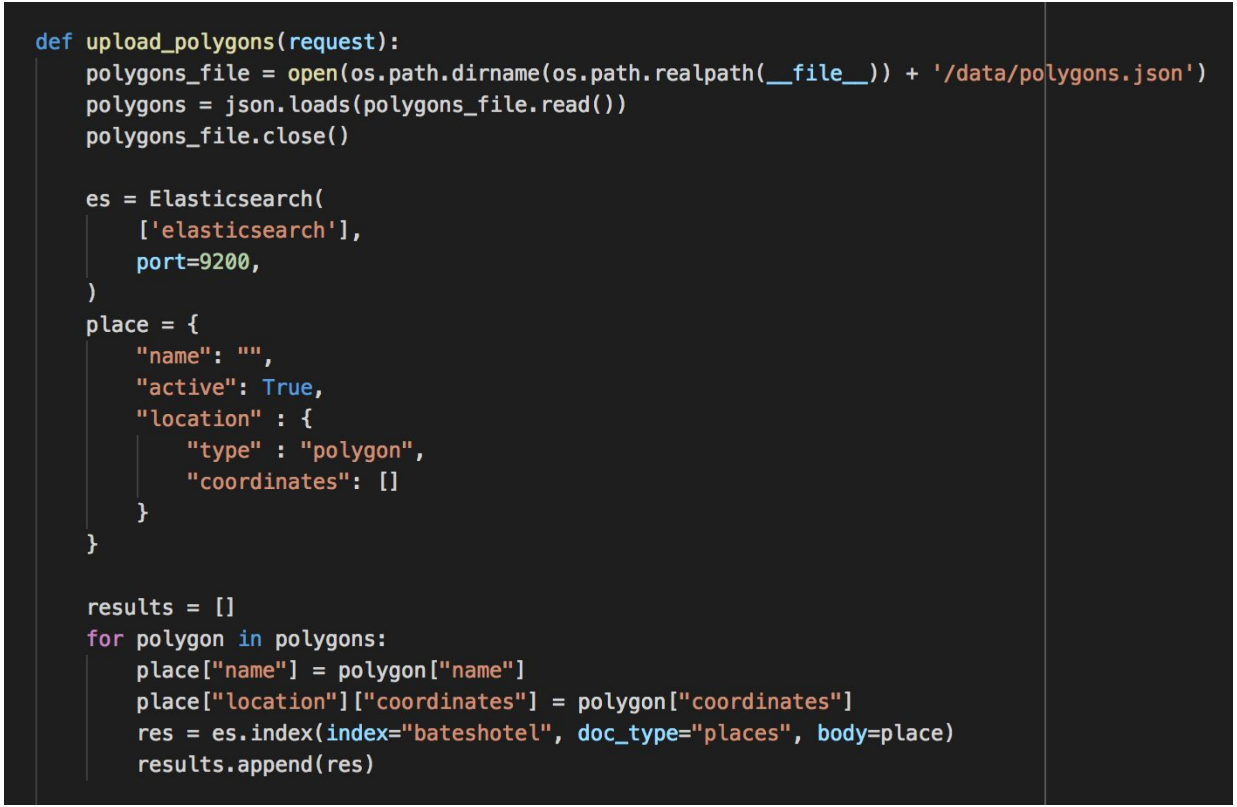
- Uses an app to display all the affiliated **stores, spas, bars and restaurants** which can be used by the Gold Members of the hotel with a fixed 20% discount.
- Each one of this places have an “area of influence”, which is a **geofence** defined by a **15 min** walk from the farthest point to the mentioned place.
- **Future Features**: Schedules, Open/Close.

IMPLEMENTATION

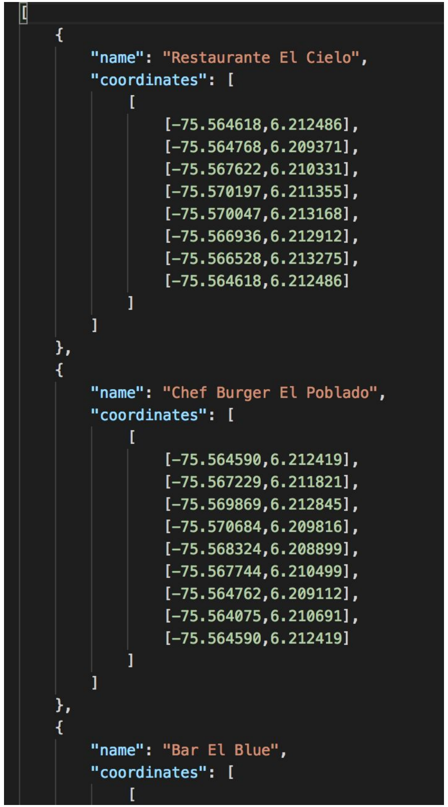


```python
from django.urls import path

from . import views

urlpatterns = [
    path('/list', views.index, name='index'),
    path('/polygon/batch-upload', views.upload_polygons, name='upload_polygons'),
    path('/api/search-places', views.search_places, name='search_places'),
]
```

IMPLEMENTATION

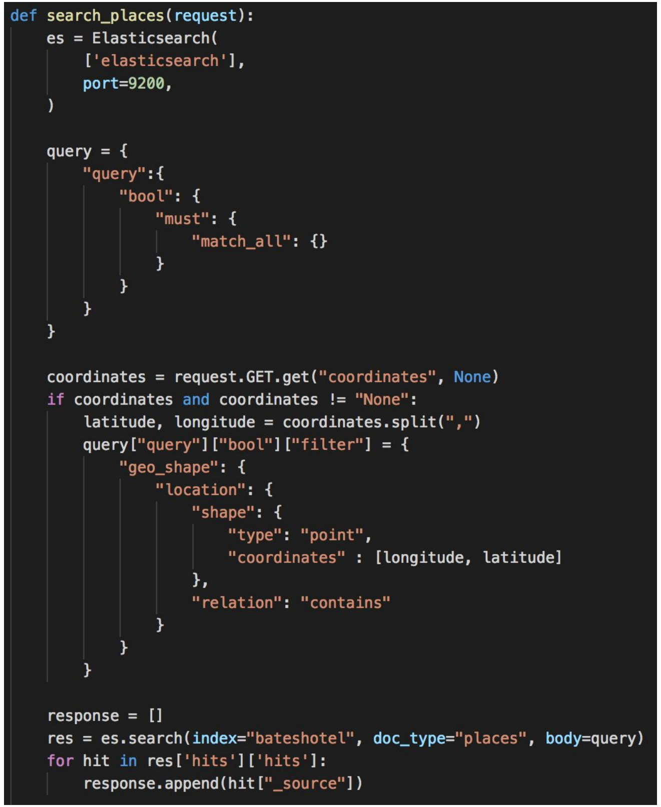


```python
def upload_polygons(request):
    polygons_file = open(os.path.dirname(os.path.realpath(__file__)) + '/data/polygons.json')
    polygons = json.loads(polygons_file.read())
    polygons_file.close()

    es = Elasticsearch(
        ['elasticsearch'],
        port=9200,
    )
    place = {
        "name": "",
        "active": True,
        "location" : {
            "type" : "polygon",
            "coordinates": []
        }
    }

    results = []
    for polygon in polygons:
        place["name"] = polygon["name"]
        place["location"]["coordinates"] = polygon["coordinates"]
        res = es.index(index="bateshotel", doc_type="places", body=place)
        results.append(res)
```

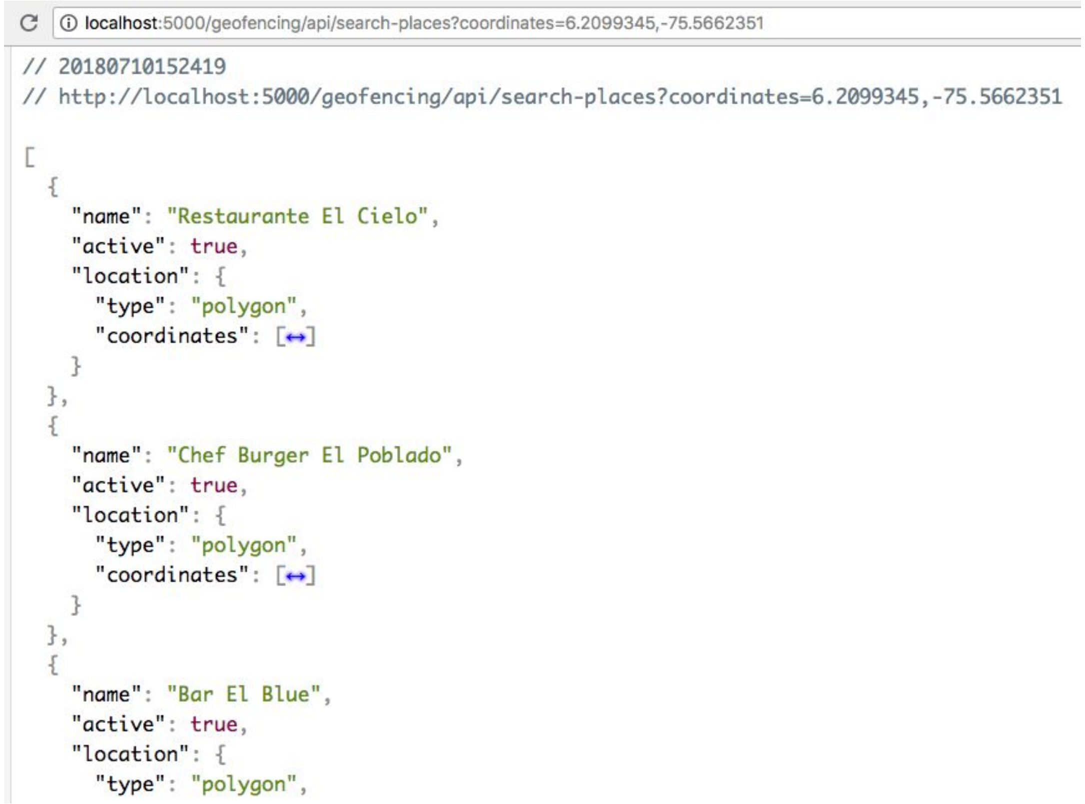


```json
[
    {
        "name": "Restaurante El Cielo",
        "coordinates": [
            [
                [-75.564618,6.212486],
                [-75.564768,6.209371],
                [-75.567622,6.210331],
                [-75.570197,6.211355],
                [-75.570047,6.213168],
                [-75.566936,6.212912],
                [-75.566528,6.213275],
                [-75.564618,6.212486]
            ]
        ]
    },
    {
        "name": "Chef Burger El Poblado",
        "coordinates": [
            [
                [-75.564590,6.212419],
                [-75.567229,6.211821],
                [-75.569869,6.212845],
                [-75.570684,6.209816],
                [-75.568324,6.208899],
                [-75.567744,6.210499],
                [-75.564762,6.209112],
                [-75.564075,6.210691],
                [-75.564590,6.212419]
            ]
        ]
    },
    {
        "name": "Bar El Blue",
        "coordinates": [
            [
```

IMPLEMENTATION



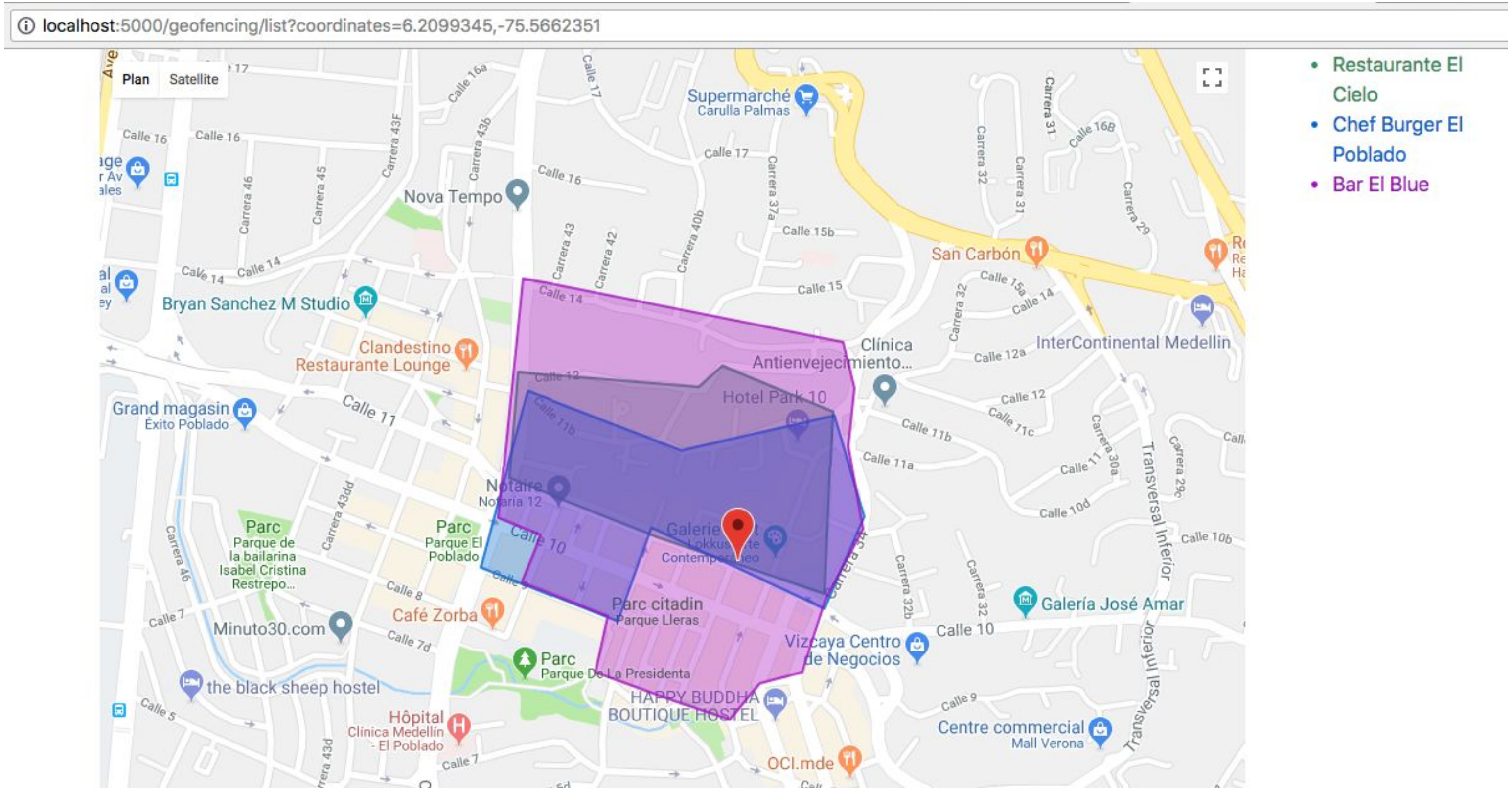
```python
def search_places(request):
    es = Elasticsearch(
        ['elasticsearch'],
        port=9200,
    )

    query = {
        "query":{
            "bool": {
                "must": {
                    "match_all": {}
                }
            }
        }
    }

    coordinates = request.GET.get("coordinates", None)
    if coordinates and coordinates != "None":
        latitude, longitude = coordinates.split(",")
        query["query"]["bool"]["filter"] = {
            "geo_shape": {
                "location": {
                    "shape": {
                        "type": "point",
                        "coordinates" : [longitude, latitude]
                    },
                    "relation": "contains"
                }
            }
        }

    response = []
    res = es.search(index="bateshotel", doc_type="places", body=query)
    for hit in res['hits']['hits']:
        response.append(hit["_source"])
```



localhost:5000/geofencing/api/search-places?coordinates=6.2099345,-75.5662351

```
// 20180710152419
// http://localhost:5000/geofencing/api/search-places?coordinates=6.2099345,-75.5662351

[
  {
    "name": "Restaurante El Cielo",
    "active": true,
    "location": {
      "type": "polygon",
      "coordinates": [↔]
    }
  },
  {
    "name": "Chef Burger El Poblado",
    "active": true,
    "location": {
      "type": "polygon",
      "coordinates": [↔]
    }
  },
  {
    "name": "Bar El Blue",
    "active": true,
    "location": {
      "type": "polygon",
```

RESULTS

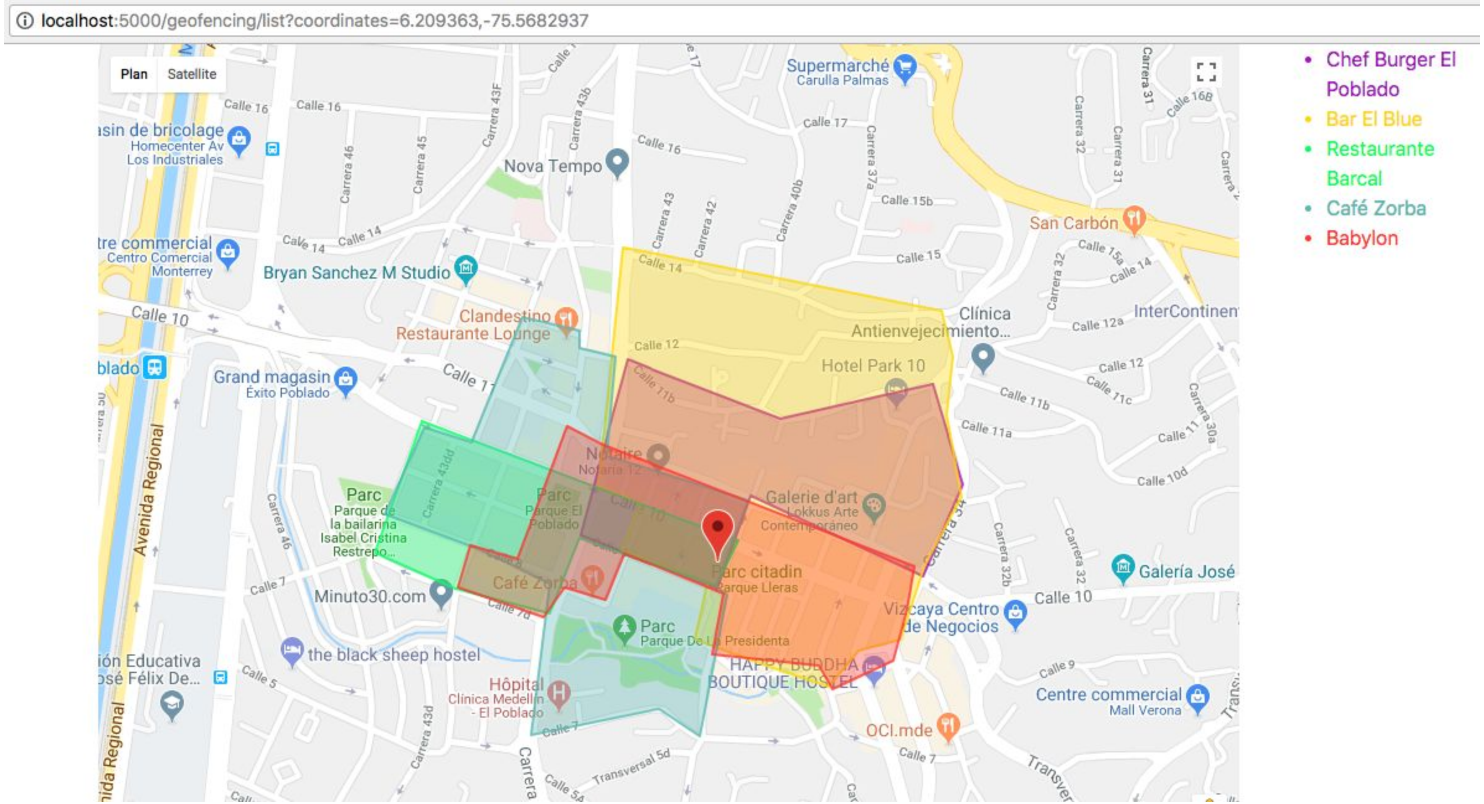


localhost:5000/geofencing/list?coordinates=6.2099345,-75.5662351

- Restaurante El Cielo
- Chef Burger El Poblado
- Bar El Blue

Plan
Satellite
Supermarché Carulla Palmas
Nova Tempo
Bryan Sanchez M Studio
Clandestino Restaurante Lounge
Grand magasin Éxito Poblado
San Carbón
Clínica Antienvejecimiento...
Hotel Park 10
InterContinental Medellin
Notaire Notaria 12
Galerie Lokkus Arte Contemporáneo
Parc Parque de la bailarina Isabel Cristina Restrepo...
Parc Parque El Poblado
Café Zorba
Minuto30.com
Parc citadin Parque Lleras
Parc Parque De La Presidenta
Vizcaya Centro de Negocios
Galería José Amar
the black sheep hostel
Hôpital Clínica Medellín - El Poblado
HAPPY BUDDHA BOUTIQUE HOSTEL
Centre commercial Mall Verona
OCI.mde
Calle 10
Calle 11
Calle 14
Calle 16
Calle 17
Transversal Inferior

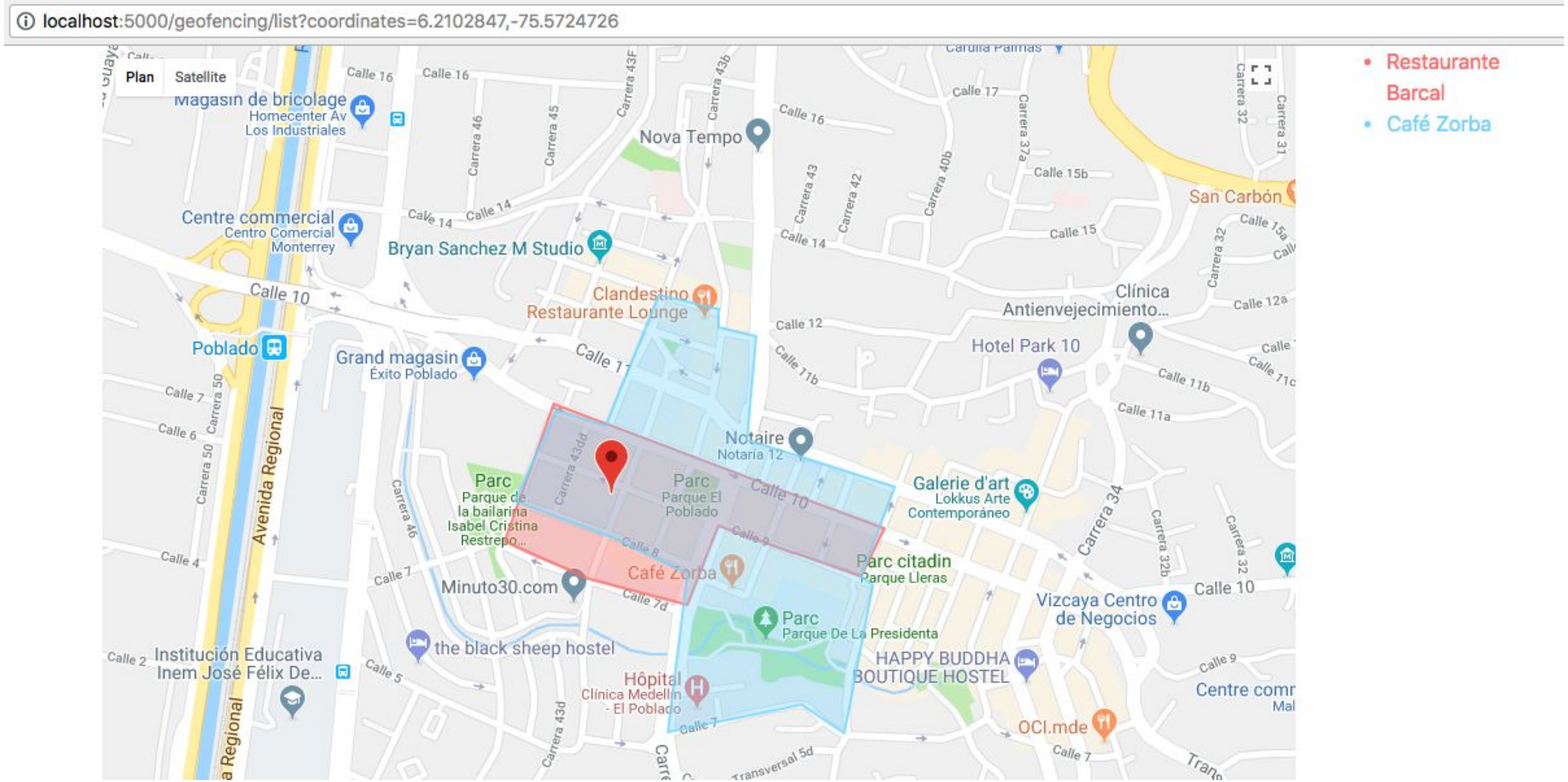
RESULTS



localhost:5000/geofencing/list?coordinates=6.209363,-75.5682937

- Chef Burger El Poblado
- Bar El Blue
- Restaurante Barcal
- Café Zorba
- Babylon

RESULTS

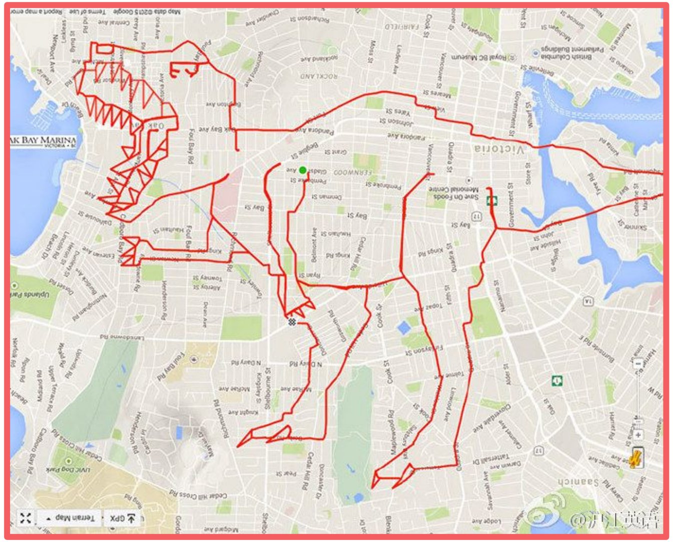


localhost:5000/geofencing/list?coordinates=6.2102847,-75.5724726

- Restaurante Barcal
- Café Zorba

IN SUMMARY

- ElasticSearch can handle thousand request per minute
- Upload of polygons is expensive, but you gain retrieval speed
- Python integration == Piece of cake
- Save polygon data in DB, migrate to ES
- It’s possible to add more fields to make complex queries
  - Open/Closed
  - Schedule
  - Active
- https://github.com/krsarmiento/bateshotel



SCALABLE GEOFENCING

Elastic Search + Django

¡Thanks!