

BoloFence

Activity-aware geofencing platform

Report Highlights

Salvatore Fiorilla (0000931332)

Igor Iurevici (0000937982)

Paola Persico (0000929044)

Introduction

Context-aware system

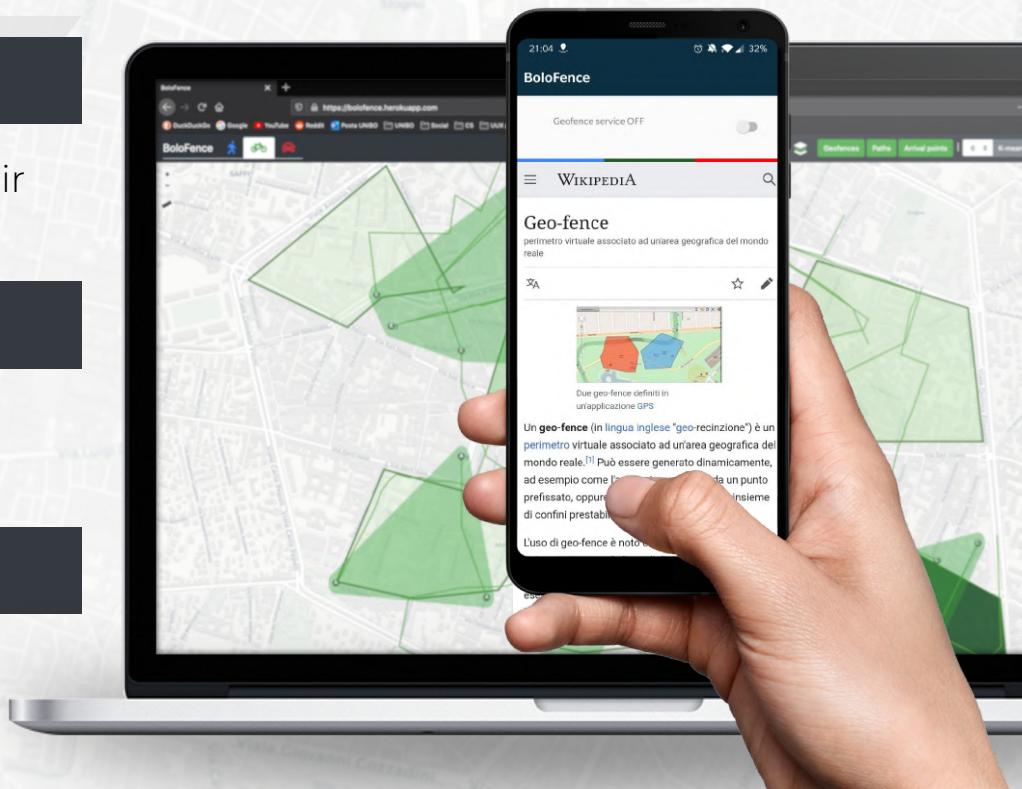
User's **location** is tracked according to their **mobility type** (walk, bike, car)

Geofence recognition

A **notification** is sent when a geofence is reached

Visualization and analysis

Web-app provides visualization and spatial data analysis tools



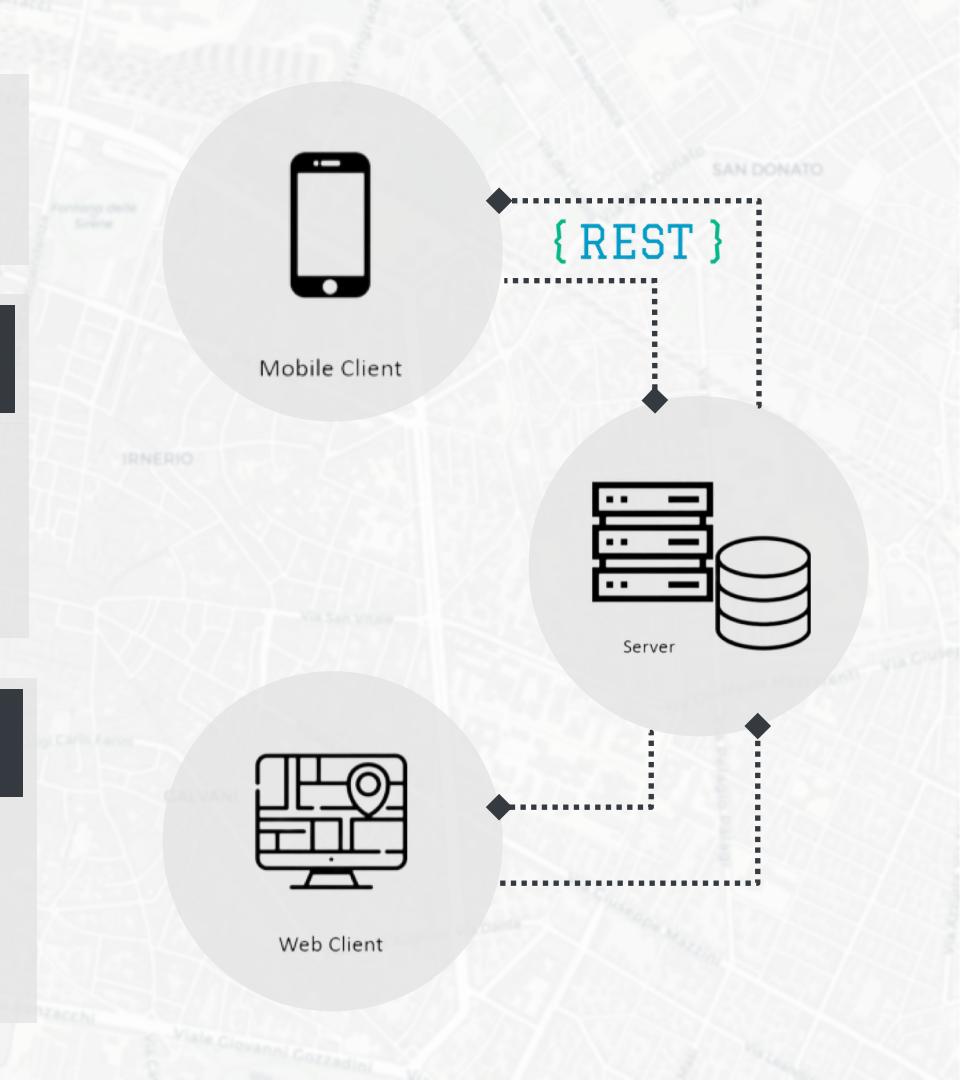
Architecture

Components

- Android client
- Server
- Web application

Client-server communication

- REST
- JSON POST request
- JSON response



Data privacy

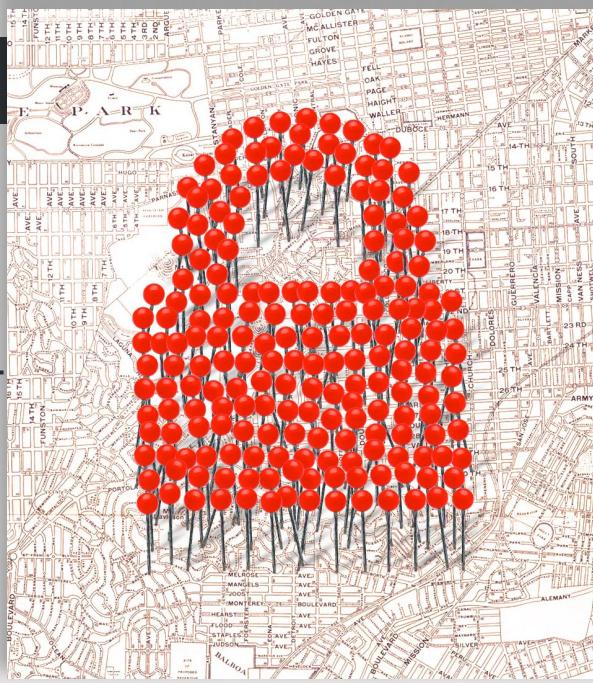
Perturbation

Coordinates are rounded
in order to grant **cloaking**

Smart precision:

Geofence proximity: ~8m
Else: ~40m

Noising is applied before
data is sent to server



Path updates

Each user session is
identified by a single-use
serial number **pseudonym**

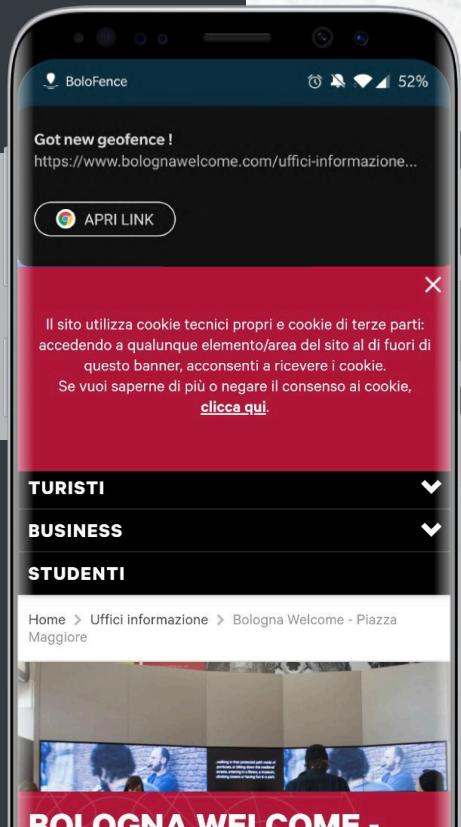
When the client dismisses
the service for a certain
amount of time, an **id reset**
occurs

System components

Client



Features



Path tracking

Coordinates are requested with **dynamic frequency**

Context awareness

Android activity recognition client provides a **mobility transition** awareness

Message notification

Notification is sent if, and only if a geofence is entered

Web content navigation

The UI includes a webview in order to allow **link browsing**

Design

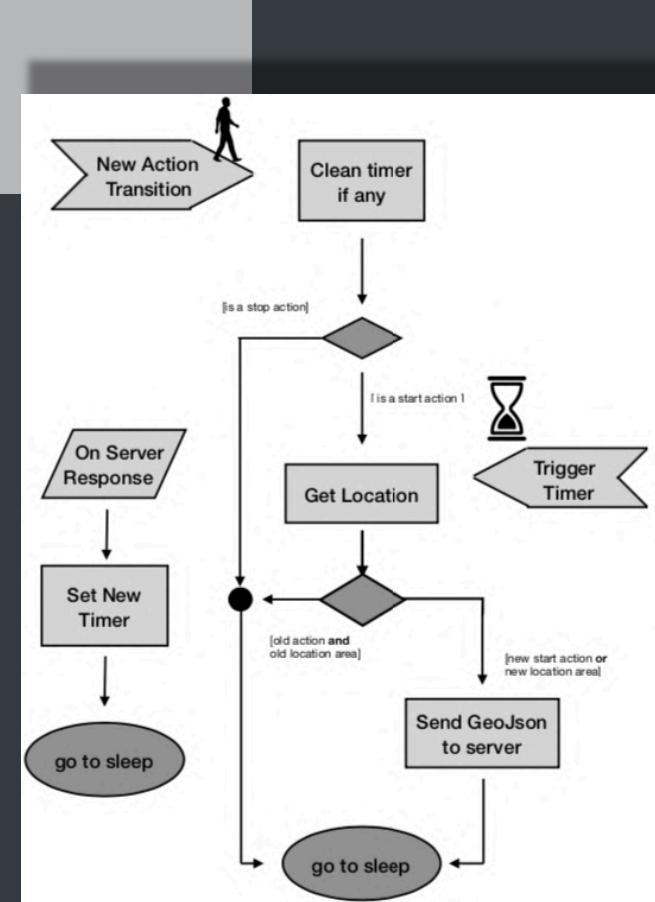
Path identifier

Tracking Service

- New action
- Timer trigger

Client-Server
geoJSON exchange

Dynamic Time-To-Sleep



Implementation

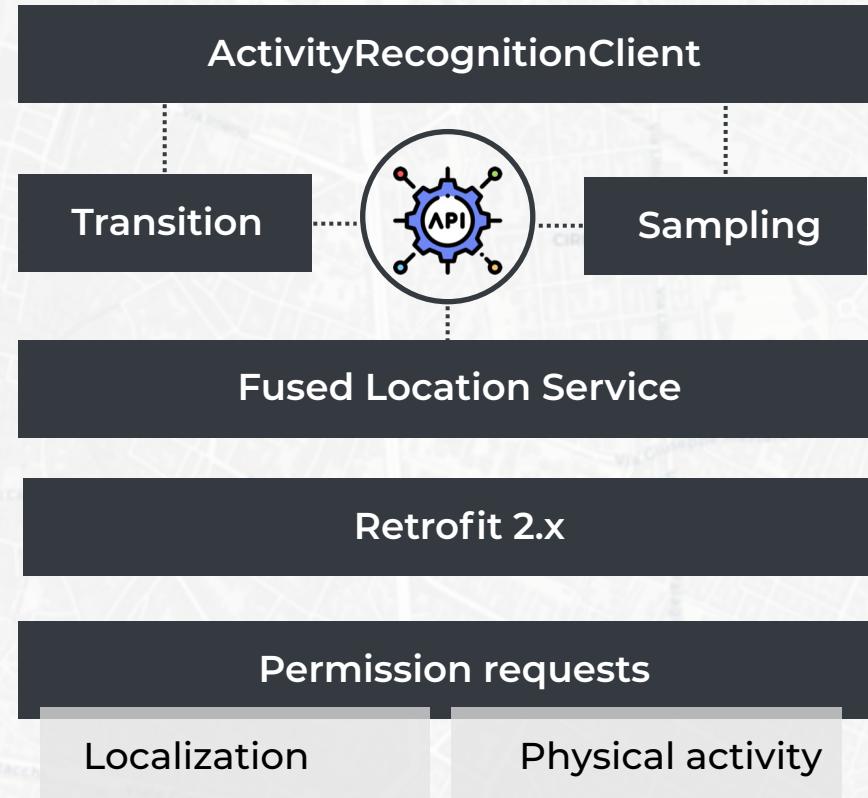
Here's an example of our geoJSON Format:

Request

```
1 {"type": "Feature",
2  "geometry": {
3    "type": "Point",
4    "coordinates": [ 11.355, 44.485 ]
5    },
6  "properties": {
7    "activity": "walk",
8    "pathId": 32,
9    "currentGeofence": "Giardini
10   Margherita"
11 }
```

Response

```
1 { "pathId": 32,
2  "timeToSleep" : "60.8346"
3  "message" : "No new Geofence"
4 }
```



System components

Server



Design

CodeIgniter Framework

- MVC template
- PHP language
- Ease in setup and usage
- Lightweight

REST Library

- CodeIgniter RestServer
- JSON POST support



CodeIgniter

Database structure

Spatial data is stored in a **SQL-based** DBMS and exploits **PostGIS** spatial queries both for requests and pre-computation.

Indexes and materialized views are used for a better performance.

GEOFENCE

- id
- geom
- message
- intensity

PATH

- id
- geom
- last_update



PostgreSQL

REST Service



Request Optimization

Time-To-Sleep

A request occurs when a time-to-sleep expires. It consists in an **underestimation** of time to reach the closest geofence. A **minimum threshold** is set up in order to avoid too low timers.

How?

Time-to-sleep is computed by considering the **distance** to the closest geofence, the **average speed** of the related activity and driving limit in the center of Bologna.

5 km/h



15.5 km/h

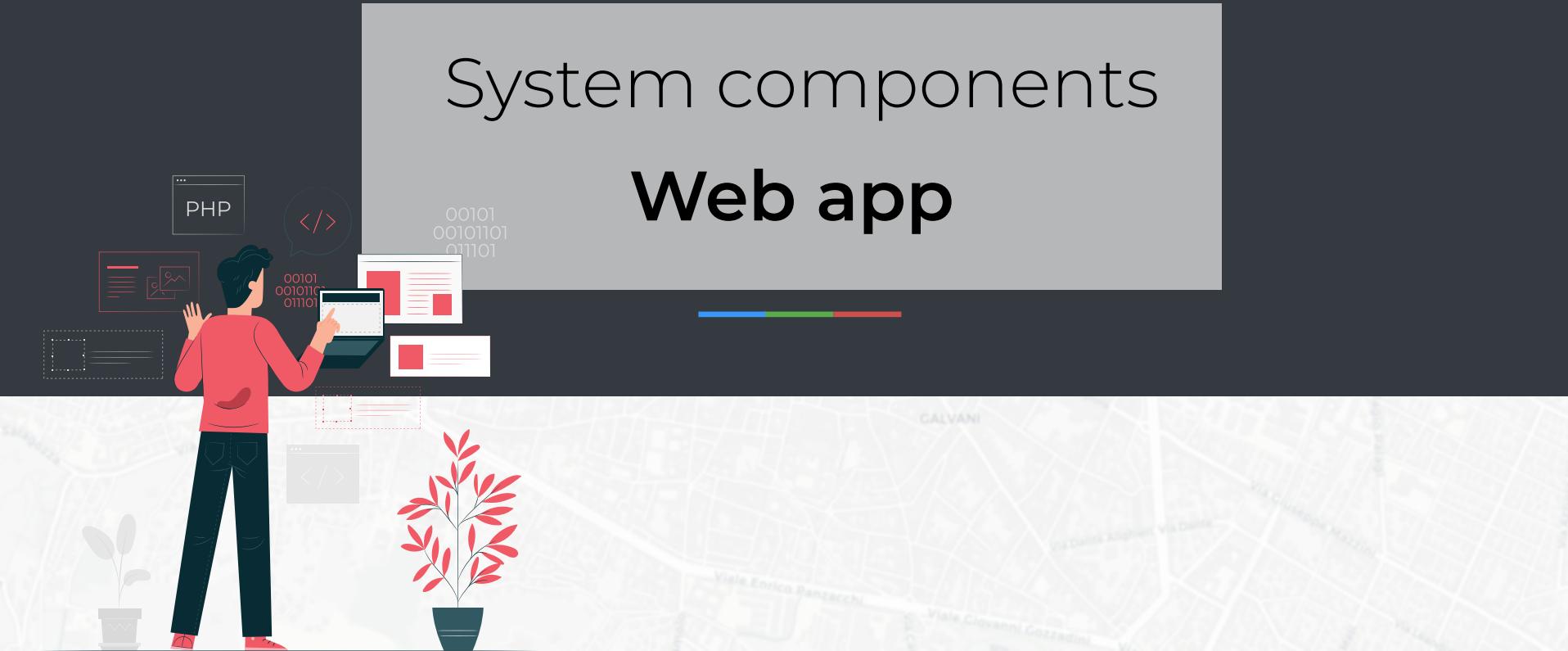


30 km/h



System components

Web app



Geofence areas

Colored polygons represent geofence and its traffic **intensity** for each activity.

Path lines

Path lines and arrival points represent users **trajectories**, for each activity.

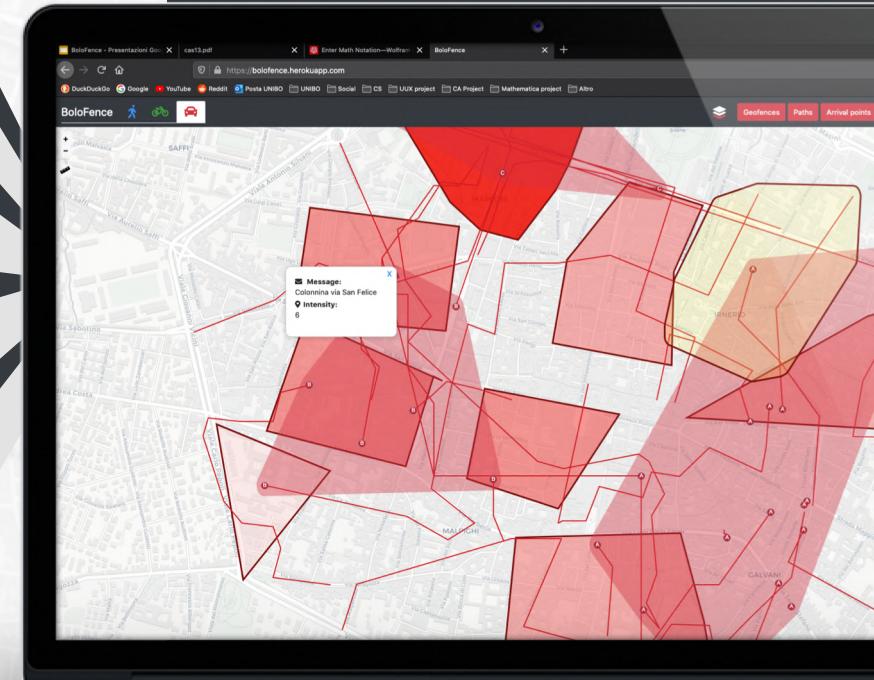
K-means clustering

Dynamic k-means tool is available.

New geofence adviser

Personalized DBSCAN tool suggests the ideal area for a **new geofence**.

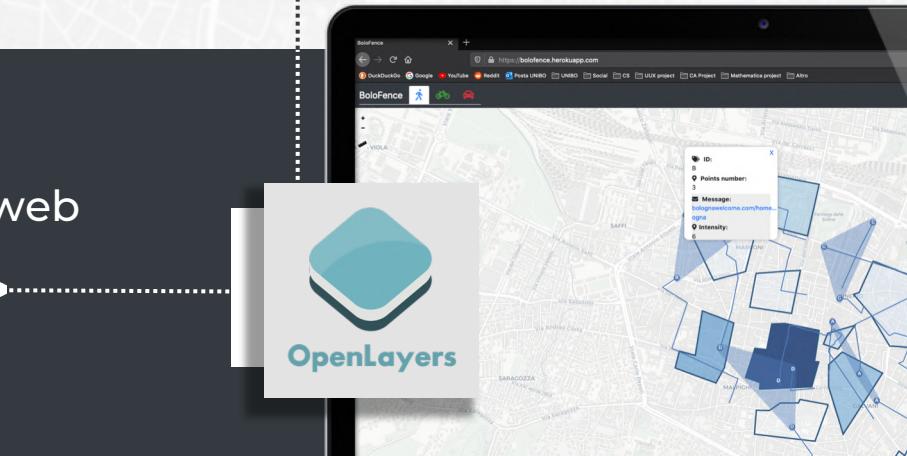
Features



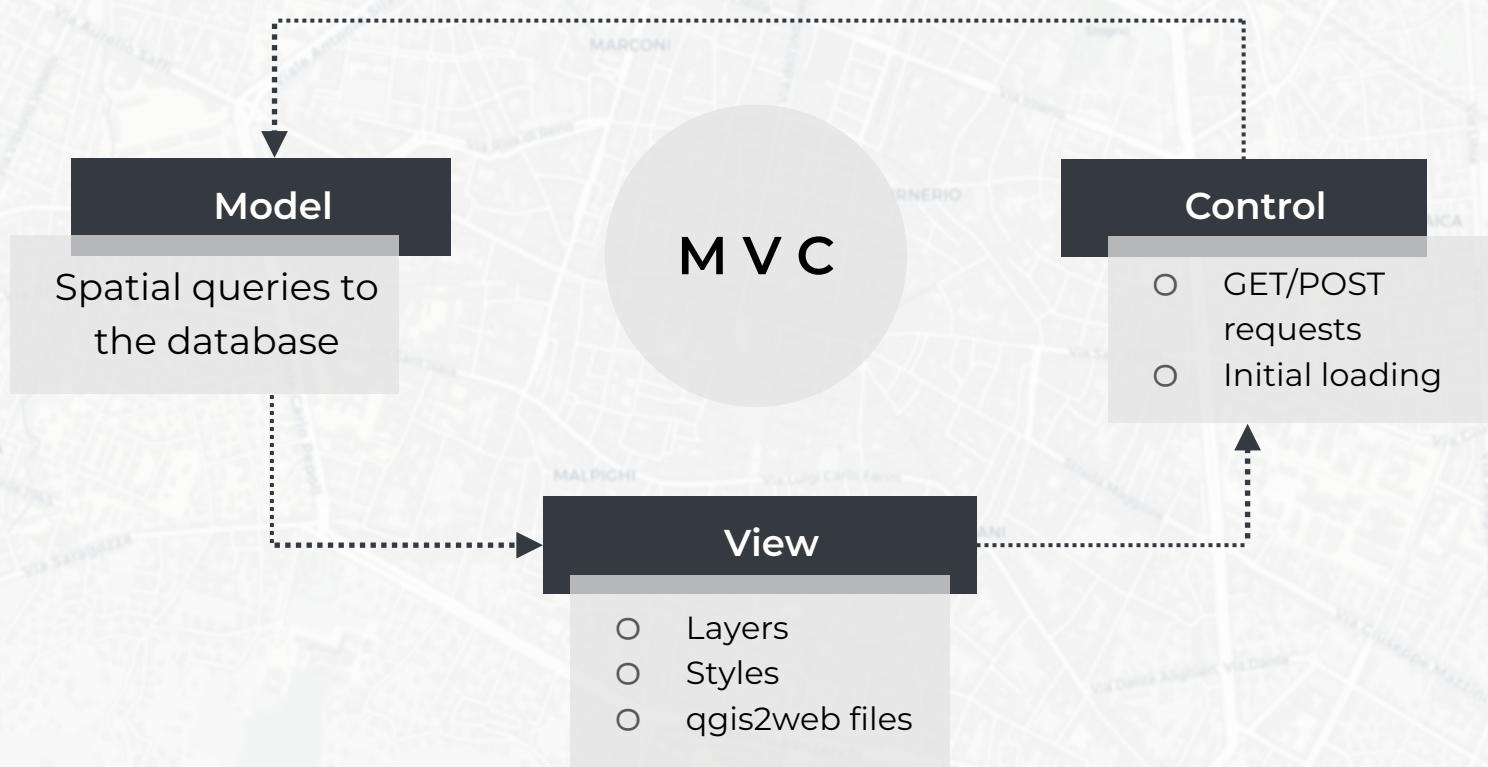
Visit now : <https://bolofence.herokuapp.com/>

Design

- Project structure has been defined through **QGIS 3.10**;
- **qgis2web** plugin transformed it into html/js/css code that exploits **OpenLayers** library;
- The result has been integrated into the CodeIgniter project and **connected** to the database.



Implementation



Geofence adviser



Generate tool **suggests** potentially the **ideal area** where a new geofence may have an high trigger rate.

1

Most relevant paths' points are retrieved by computing the **closest point** between each pair of paths

2

Distance between paths is tracked and the **eps value** is determined according to the **user percentile** value

3

DBSCAN algorithm is applied on the computed data

4

The cluster's shape with the **highest intensity** is returned

Testing and evaluation



The system has been tested both
in a **real situation** and through
position emulator (Lockito)



Avg. delay between
real geofence entrance
and notification

Lockito



Avg. delay between
real geofence entrance
and notification

Live testing



First activity
sampling avg. delay

Live testing



Avg. Dev. between real
geofence entrance
and notification

Live testing

Conclusions



The provided service accomplishes the given assignment and grants satisfactory **performance** and **reliability** in order to be used in real life situations.

Results

Future works

- Activity recognition **ad hoc** API
- Trajectory prediction
- Ad hoc clustering algorithms