MICROSERVICE FOR CREATING GEOFENCES

Victoria Velkova[1], Rosen Ivanov[2]

[1,2]Technical University - Gabrovo, 4 "H. Dimitar", Gabrovo, Bulgaria
v.velkova@tugab.bg, rs.soft.bg@gmail.com

Abstract

This paper presents the architecture and implementation of a microservice for creating geofences and exporting their coordinates in GeoJSON format. The service allows describing geofences in three ways: (1) by a polygon; (2) by a circle with a given center and radius; and (3) by a circle that is approximated by a polygon. The service provides the ability to populate geofences with geo hashes in order to very quickly calculate which geofence the clients of a location-based service fall into. Different levels of accuracy and geohash generation strategies are supported. At this stage, the service exports the geofence information in a format adapted for MongoDB database. Experiments have been conducted for different types of geospatial queries with geofences generated by the proposed service (buildings and open spaces of the Technical University of Gabrovo). The experiments show that all types of queries are executed in real time with active 2dsphere database indexing.

Keywords: Geofencing, Push notification.

INTRODUCTION

A geofence is a virtual perimeter set as a geographic boundary from the real world [1]. The area of the geofence can be set to form a radius around a specific location or to follow specific geographic contours. Geofencing is mainly used by Location Based Services (LBS) [2]. Through geofencing, it is possible to track when a client or group of clients enter or exit a geofence. User feedback is mainly implemented by sending push notifications to clients' mobile devices.

The most common use of geofencing in 2022 is for location-based marketing and advertising [3]. Geofencing enables the sending of real-time hyperlocal notifications to mobile app users based on their location. A distinction must be made between geotargeting and geofencing. By using geotargeting, advertisers can specify the location around which they want their customers to receive advertising messages. This is mainly implemented by using Bluetooth Low Energy (BLE) beacons [4]. Geofencing allows greater functionality and unlike geotargeting does not require additional hardware for its implementation. Geofencing can be as large as a city or neighborhood, or as small as a building. Geofences can have different shapes - most commonly a circle or polygon.

There are many uses of geofencing, but one of the main benefits is that businesses can get feedback from their customers and use this information in real-time to improve the customer experience. Geofencing allows businesses to narrow down their pool of potential customers to people who visit a specific geographic area. Adapting geofencing into marketing can be an effective tool to not only increase sales, but also improve the customer experience.

Geofencing is a technology that is easily implemented in modern mobile technologies. Statistics for 2022 [5] show that (1) geofencing is compatible with 92% of smartphones in use; (2) mobile ads via geofencing have double the click-through rate; (3) 53% of shoppers have visited a retailer after receiving a push notification from it; and (4) 90% of mobile messages are read within 3 minutes.

Geofencing-based services can use their own mobile app or the geofencing capabilities

of social networks. The social network Facebook provides the ability to set up geofencing around a location with a radius of up to one mile. Instagram allows the generation of polygon-shaped geofences. An additional feature is creating a temporary geofence for a live event. Twitter allows more than one location to be selected for geofencing purposes. It is possible to select an audience that speaks a specific language within a geofence, allowing customization of the ad depending on the language customers speak. Snapchat also supports geofencing, additionally offering geofilters - filters specific to certain areas.

Services that use their own mobile app provide greater functionality. In this case, it relies mainly on cloud services that support geofencing. There are numerous cloud services that support geofencing and segmentation of users. OneSignal [6] is a multiplatform notification service. OneSignal supports segmentation of users based on their GPS position and specific segmentation using custom tags. The problems with OneSignal are basically two: 1) Geospatial segmentation is only a rectangular shaped area; and 2) The number of segments is limited - 6 on the free plan and 20 on the professional plan. The PlotProjects [7] service offers better possibilities related to segmentation of users. The free plan allows defining up to 1000 geofences. With this service, we can describe a geofence in two ways - by a polygon and a circle with a certain radius. Both services do not allow the generation of the geohash signature for geofences. PlotProject allows the use of GPS maps in the creation of geofences, but does not offer an easy way to correct their contour. Exporting the data describing geofences is only possible with a paid plan activated.

The aim of the development is to create a microservice that offers a user interface through which geofences of arbitrary shape can be created quickly and intuitively using GPS maps. The service allows exporting the data in GeoJSON format to a MongoDB database. It is also possible to create and update the database on-line. It is possible to obtain geohashes for each geofence with the accuracy desired by the user.

DESCRIPTION OF THE SERVICE

The service is implemented as a Node.js microservice. The microservice architecture enables the rapid building of complex programming systems with a high level of reliability. Each microservice works with its own database. The NoSQL database MongoDB in the Azure cloud infrastructure is used. The interface between the microservices and the databases is implemented using the MongoDB App Service (former name Realm). The technology stack that the microservice uses is as follows:

- Node.js - open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.
- Express - a web framework for Node.js. This framework is used to communicate with the microservice through a browser.
- Realm Web - package for authentication and communication with MongoDB databases using HTTPS protocol.
- Mapbox GL-JS - an open source JavaScript library that uses Mapbox GL to render interactive GPS maps. It is used to describe the contours of geofences.
- Mapbox GL Geocoder – this JavaScript library adds a geocoding control to a GPS map, allowing users to search for objects on the map.
- Turf - JavaScript library for geospatial data analysis and processing.

The service supports a Web interface through which users can very quickly create the geofences they need. Fig. 1 shows the components of the user interface.

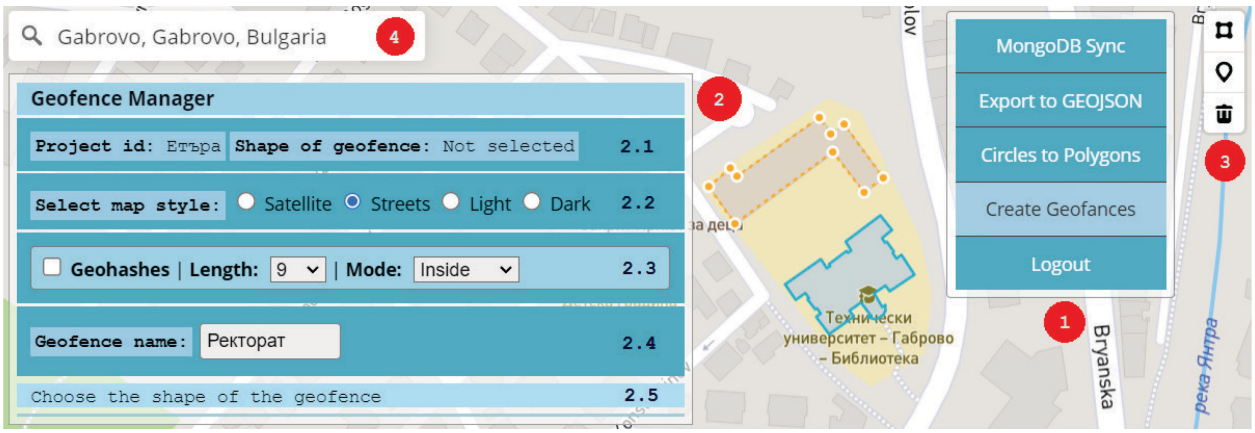
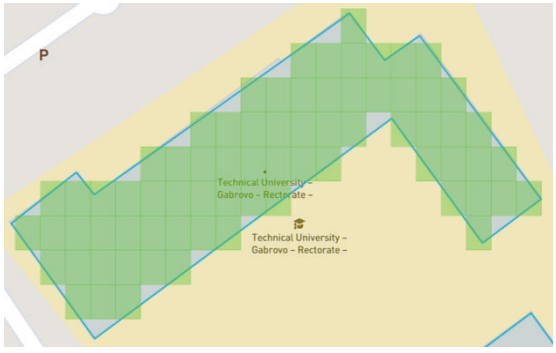


Fig. 1. *User interface of the service*

Access to the service requires authorization. It is implemented through menu 1 (Login). If you want to synchronize in real time the entry, update and deletion of geofences with the database you must select menu item "MongoDB Sync". Some databases do not support working with geofences that are described by circles. In this case, menu item "Circles to Polygons" converts all entered circles to polygons with a specified number of points. The service allows to save the entered geofences to a file in GeoGSON format via menu item "Export to GEOJSON". When you want to start entering geofences select menu item "Create Geofences." In this case, you get access to menu 2 and 3, as well as the search bar for the desired geographical area or object - 4. The search for the desired location is implemented using the Mapbox GL Geocoder.

Using section 2.2 of menu 1 you can set the type of GPS map you want to use. GPS maps from OpenStreetMap and NASA are rendered using the Mapbox GL-JS library. Four GPS map types are supported: satellite, street, light and dark mode. Information about the selected project name and the currently selected geofence shape (polygon or circle) is obtained via section 2.1. Section 2.4 is used to specify the name of a selected geofence. Section 2.5 dynamically obtains information specific to the selected mode of operation, such as GPS coordinates of geofence points or geohash values for a given geofence. Section 2.3 enables the calculation of a geohashes for a selected geofence. A geohash is a string of a certain length that is a unique identifier of a specific geographic region with a rectangular shape. The longer the string, the more accurately the region is described. Points that are in close geographical proximity will have the same geohash. This allows a very quick calculation of whether a point is inside or outside a region. It is possible to choose the precision of the geohash description (length) and the algorithm to fill the geofence with geohashes. The possible fill modes are “inside” and “intersect”. In the “inside” fill mode, the center of each geohash is in the geofence. In “intersect” mode, all geohashes that fall within the geofence or intersect the geofence contour are searched for with a user-specified overlap factor. Fig. 2 shows the filling of a selected geofence with geohashes, as well as their symbolic values.



a)

Geohashes (72):

sxd5bj6cc	sxd5bj69x	sxd5bj6c8	sxd5bj6c9	sxd5bj6cd
sxd5bj6ce	sxd5bj6cs	sxd5bj69m	sxd5bj69q	sxd5bj69r
sxd5bj6c2	sxd5bj6c3	sxd5bj6c6	sxd5bj6c7	sxd5bj6ck
sxd5bj6cm	sxd5bj695	sxd5bj69h	sxd5bj69j	sxd5bj69n
sxd5bj69p	sxd5bj6c0	sxd5bj6c1	sxd5bj6c5	sxd5bj6ch
sxd5bj6cj	sxd5bj6cn	sxd5bj68c	sxd5bj68f	sxd5bj68g
sxd5bj68u	sxd5bj68v	sxd5bj68y	sxd5bj68z	sxd5bj6bb
sxd5bj6bu	sxd5bj6bv	sxd5bj6by	sxd5bj6bz	sxd5bj62t
sxd5bj62w	sxd5bj62x	sxd5bj688	sxd5bj689	sxd5bj68d
sxd5bj68e	sxd5bj68s	sxd5bj68t	sxd5bj68w	sxd5bj6bt
sxd5bj6bw	sxd5bj6bx	sxd5bj708	sxd5bj62k	sxd5bj62m
sxd5bj62q	sxd5bj62r	sxd5bj682	sxd5bj683	sxd5bj686
sxd5bj687	sxd5bj68k	sxd5bj6bq	sxd5bj62j	sxd5bj62n
sxd5bj62p	sxd5bj680	sxd5bj681	sxd5bj684	sxd5bj4ry
sxd5bj4rz	sxd5bj4xb			

b)

Fig. 2. *Geohashing: (a) visual and (b) symbolic representation of geohashes*

The menu 3 can be used to select the shape of the geofence (polygon or circle) or the geofence delete mode. The service allows at any time editing the contour of an already entered geofence - adjusting the position of selected points, as well as entering additional points between any two points. It is allowed to have a geofence within another geofence.

EXPERIMENTS

In order to test the feasibility of the proposed microservice, a MongoDB database with geofences was created. The geofences describe 11 buildings and one sports ground of the Technical University - Gabrovo. The database was created by importing the GeoJSON file that the proposed service generates. Fig. 3 shows the format in which the geofences are described.

```
{
   "name": "Ректорат",
   "region": {
     "type": "Polygon",
     "coordinates": [[[
        25.315752044161997,
        42.87559546661549],...
     ]]]
   }
}
```

Fig. 3. *Geofence in GeoJSON format*

The MongoDB database provides multiple operators for working with geospatial queries, for example:

- $geoWithin - check if a GPS location (client position) is in geofence.
- $geoIntersect - check if a GPS location or area (polygon) is in a geofence or has intersections with a geofence.
- $near - check if a GPS location near each of the geofences without considering the Earth's radius.
- $nearSphere - check if a GPS location near each of the geofences considering the Earth's radius.
- $geoNear - content aggregation operator - checks if a GPS location near each of the geofences and returns the distance to nearby geofences.

The $near and $nearSpere geo operators require a 2dsphere index to be set for the database.

Collection Name	Documents	Logical Data Size
students-100	100	18.85KB
students-1000	1000	188.48KB
students-10000	10000	1.84MB
students-200	200	37.7KB
students-2000	2000	376.95KB
students-50	50	9.42KB
students-500	500	94.24KB
students-5000	5000	942.38KB

Fig. 4. *Sample service client databases*

The $geoWithin and $geoIntersect operators can work with and without indexing. To test the runtime of these operators, eight databases were created that contain information about clients, including their GPS locations. The databases differ only in the number of clients (from 50 to 10000), as shown in Fig. 4.

```
{
 "location": {
  $geoIntersects: {
   $geometry: {
    "type": "Polygon",
    "coordinates": [[
      [longitude,latitude],
      ...
     ]]
   }
  }
 }
}
```

```
{
 "region": {
  $near: {
   $geometry: {
    "type": "Point",
    "coordinates":
       [longitude,latitude]
   },
   $minDistance: 0,$maxDistance: 20
  }
 }
}
```

Fig. 5. *Database query syntax: $geoIntersect ($geoWithin) and $near*

The position of the clients (students) is randomly generated to fall within the region of Technical University - Gabrovo.

Fig. 5 shows the format of the tested geospatial queries. The operators $geoWithin and $geoIntersect are used to test the time it takes to compute which clients are in a geofence specified using the $geometry operator. The $near operator is used to check which geofences are close (20 m) to the specified client whose position is specified by GPS location (longitude, latitude).

The response time for all geospatial queries was below 1ms, regardless of the number of clients in the students-xxxxx databases. The analysis was implemented using the MongoDB Compass app and Explain Plan tool. The reason for these very good results is that MongoDB uses internal geohasing for geofences and client positions when 2dshrere indexing is enabled.

A Node.js application was developed (see Fig. 6) that displays the position of all clients that are in a given geofence. The aim is to estimate the response time while considering the time delay in the communication channels. Experiments show that with active 5G network connectivity this time is below 290ms (195ms on average). The reason for this delay is the use of MongoDB App Service (Realm) to access the MongoDB databases.

CONCLUSION

A microservice has been developed to graphically describe geofences using polygons and circles. The geofence information can be exported as a GeoJSON file or directly saved to a MongoDB database. The service allows editing of already entered geofences by moving and deleting contour points or entering new points. The microservice is part of a service for delivering personalized content to visitors of open-air museums [8]. The architecture of the microservice allows it to be easily integrated to other projects that use geofencing in order to deliver personalized content and improve the customer experience. In the future, it is planned to enable the export of geofences in a format specific to other databases that support geospatial queries.

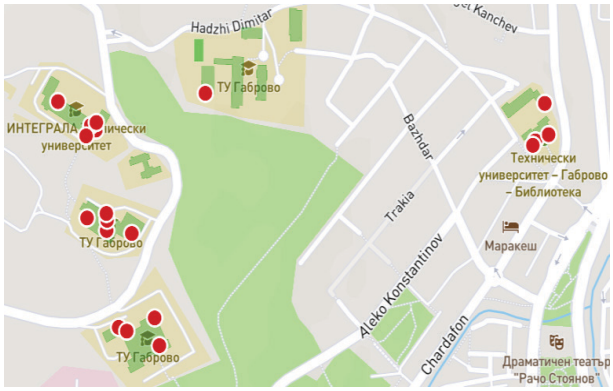


Fig. 6. *Visualization of the geofences (green) and the position of the clients that are in geofences (red dots)*

REFERENCES

[1] Statler, S. Geofencing: Everything you need to know. In Beacon Technologies (pp. 307-316). Apress, Berkeley, CA, 2016.

[2] Rahate, S. W., & Shaikh, M. Z., Geo-fencing infrastructure: Location based service. International Research Journal of Engineering and Technology, 3(11), 1095-1098, 2016.

[3] Ho, Y. J. I., Chen, X. O., Mao, S., Catch Me If You Can! The Economic Analysis of Geofencing, 2022.

[4] Alzoubi, H., Alshurideh, M., Kurdi, B., Akour, I., & Aziz, R., Does BLE technology contribute towards improving marketing strategies, customers' satisfaction and loyalty? The role of open innovation. International Journal of Data and Network Science, 6(2), 449-460, 2022.

[5] Plotprojects. What is Geofencing Marketing? How to Add it to Your Strategy, Oct. 1, 2022, from https://www.plotprojects.com/blog/what-is-geofencing-marketing-how-to-add-it-to-your-strategy/

[6] OneSignal, OneSignal - Customer Messaging Delivered, https://onesignal.com, Oct. 2022.

[7] PlotProjects, The best geofencing software plugin for mobile apps. https://www.plotprojects.com, Oct. 2022.

[8] Ivanov, R., Velkova, V., Delivering personalized content to open-air museum visitors using geofencing, Int. Conference Digital Presentation and Preservation of Cultural and Scientific Heritage, Vol. 12, pp. 141-150, 2022.

Acknowledgements

This research has been partially funded by Bulgarian Ministry of Education and Science, project №2003E and project №2009E.