Geospatial Queries using Python and MongoDB

Submitted by
Siddharth Kumar Yadav
(0701CS191059)

Submitted to
Rekha Singh
[Assistant Professor
Python Programming]

PROJECT OVERVIEW

Searching of Available Stores within a particular radius of circle

Our project revolves around building a fast solution to achieve the aim using Python by implementing basic syntaxes,loops,functions and some other libraries

Techstack used :

1) Python -> Backend
2) Javascript -> UI
3) MongoDB -> Storage

Use case

1) For delivery ranges
2) For security purposes & triggering alerts
3) Location Tracking, many more…

What is API?

Application programming interfaces, or APIs, simplify software development and innovation by enabling applications to exchange data and functionality easily and securely.

Source : IBM

What is Flask?

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.

What is MongoDB?

MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need. (NoSQL Database)

Environment Setup

- Setting up a MongoDB cluster on cloud
- Inserting some random data in database
- Basic Installation of Python & MongoDB
- Single Page web UI application with a search box

https://python-lab-project-sky-uecu.surge.sh/

API : Searching of Available Stores within particular radius of circle

Approach 1 : Traverse the documents of the collections and for each records we find the distance between the user Current Location and store Coordinates and check weather its inside or outside the provided range and eventually return them

Approach 2: Using Inbuilt Geospatial Queries support for MongoDB

Approach 2 helped us to reduce the response time by 55%

$$\frac{(173ms - 78ms)}{(173ms)} * 100\ \% = \mathbf{54.91\%}$$

Analysis for Approach 1

```
def is_inside(x1,y1,x2,y2):
    # approximate radius of earth in km
    R = 6373.0
    lat1 = radians(x1)
    lon1 = radians(y1)
    lat2 = radians(x2)
    lon2 = radians(y2)
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) *
sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))
    distance = R * c
    # print("Result:", distance,"Km")
    return distance
```

Geospatial Queries

These is an inbuilt feature available in MongoDB to perform filter on coordinates , providing programmer a flexibility to focus more on business logic then writing computation logics at server, its fast optimized and requires less code

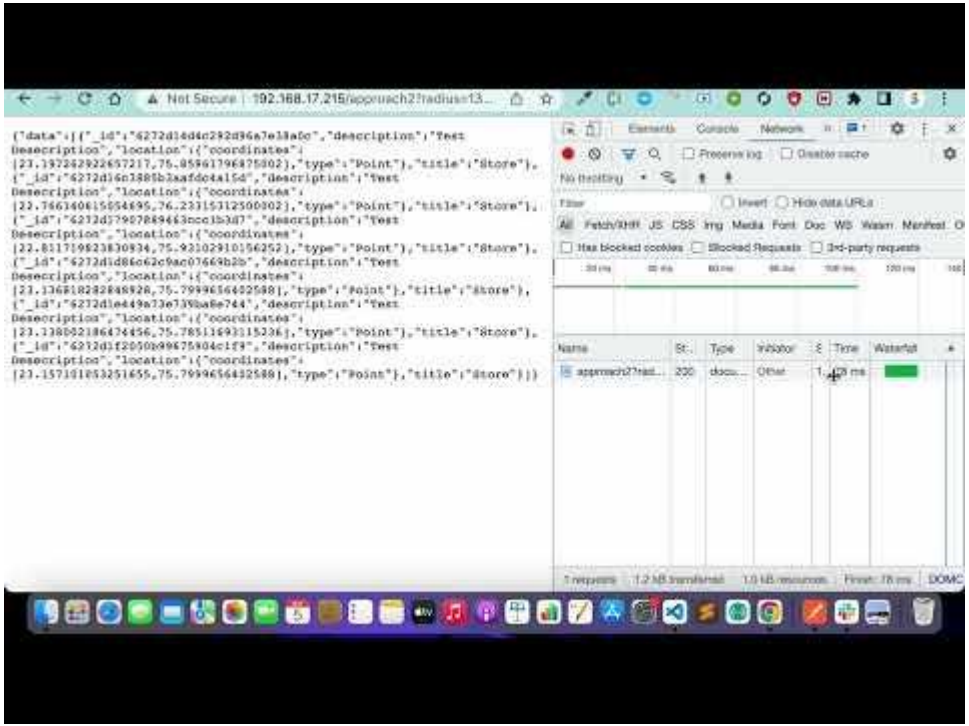
With this approach we first create a Index in MongoDB (indexes are used to product fast output for find commands it uses pointers to prepare a list of fields on which basis index was created for) , with reference to a field in document location field in our case & then we execute the query

- With Older approach we have to write a function to find distances and filter the records which occupies server space & memory as python function is getting called, but with geospatial queries entire filtering and execution is done on MongoDB cluster
- Fast response

Analysis for Approach 2 & Comparison with Approach 1

Index Creation
db.storesAgain.create_index([("location", GEOSPHERE)])

```
def index2():
    args = request.args
    radius = float(args.get("radius"))
    latitude = float(args.get("lat"))
    longitude = float(args.get("lng"))
    if radius is None or latitude is None or longitude is None:
        return jsonify({"success": False, "message": "Please enter
valid radius,lat and lng field in query"})
    records = (db.storesAgain.find({"location": {"$geoWithin": {
        "$centerSphere": [[latitude,longitude], (radius)/6371]}}}))
    ans = []
    for store in records:
        store['_id'] = str(store['_id'])
        ans.append(store)
    return jsonify({"data": ans})
```



Outcome

- Learnt to implement syntaxes , loops , functions , working with flask , parameter retrieval from query of an end point,
- Reading MongoDB and mongo lib
- Performing Deployment to heroku
- Using rsuite - ReactJS frontend framework for Frontend,
- working with Google Maps Javascript library

Hosted Links

Frontend
https://python-lab-project-sky-uecu.surge.sh/

Backend
https://python-project-sky.herokuapp.com/

[Github Repository Link]

References

https://pymongo.readthedocs.io/en/stable/examples/geo.html

https://www.npmjs.com/package/google-maps-react