Tribhuvan University

Institute of Engineering

Pulchowk Campus

Big Data Technologies

Project Report

Restaurant Finder

Submitted by:

Aayush Lamichhane
075BCT005

Bishal Katuwal
075BCT028

Bishant Baniya
075BCT030

Gobind Pd Sah
075BCT038

Submitted to:

Department of Electronics and Computer Engineering
Pulchowk Campus

Submitted on:

30th March, 2023

Contents

1 Introduction **4**
1.1 Background 4
1.2 Problem Statement 4

2 Technologies used **4**
2.1 MongoDB 4
2.2 ElasticSearch 4
2.3 Flask 4
2.4 HTML/CSS 5

3 Methodology **5**

4 Code **5**

5 Results **8**

6 Conclusion **10**

7 Appendix **10**

A Project Link **10**

B MongoDB data sample **10**

C Elasticsearch index Mapping **10**

List of Figures

1 Search Restaurants with Indian Cuisine 8
2 Search Restaurants with Chinese Cuisine 9
3 Search Restaurants with Russian Cuisine 9

List of Tables

1 Sample Data from MongoDB 10
2 Elasticsearch index mapping for restaurant data 10

1 Introduction

1.1 Background

The purpose of this project is to create a restaurant finder application that allows users to search for restaurants by cuisine using Elasticsearch. The application retrieves data from a MongoDB database and transforms it to be indexed into Elasticsearch. Users can then input a cuisine type into the search bar and the application will return a list of restaurants matching the search criteria.

1.2 Problem Statement

Finding a restaurant that serves a specific cuisine can be time-consuming and difficult, especially in a large city with many dining options. The goal of this project is to create a web-based application that allows users to search for restaurants based on cuisine type and location, and view the results in a user-friendly format.

2 Technologies used

The project relies on two primary technologies: MongoDB and Elasticsearch.

2.1 MongoDB

MongoDB is a NoSQL document-oriented database. It is designed to store data in JSON-like documents, making it easy to store and retrieve data from the database. MongoDB is highly scalable and can handle large volumes of data. In this project, MongoDB is used to store data about restaurants, including their names, cuisines, and addresses.

2.2 ElasticSearch

ElasticSearch is an open-source, distributed search engine. It is based on the Lucene library and provides full-text search capabilities. ElasticSearch is highly scalable and can handle large volumes of data. It is commonly used for logging, monitoring, and analytics. Elasticsearch is used in this project to index and search through the restaurant data stored in MongoDB, and to return relevant search results based on the user's input.

2.3 Flask

Flask is a lightweight web framework for Python. It is used to build web applications quickly and easily. Flask is flexible and allows developers to customize it to their needs. It provides support for various extensions that can be used to add additional functionality. The web-based application is built using the Flask web framework. The application allows users to search for restaurants based on cuisine type and location, and returns results in a user-friendly format that includes the name, cuisine, address, and borough of each restaurant.

2.4 HTML/CSS

HTML is a markup language used to create web pages. It defines the structure and content of the page. CSS is a style sheet language used to add style and formatting to web pages. It defines the presentation of the page, including layout, colors, fonts, and more. Together, HTML and CSS are used to create visually appealing and functional web pages. Our application also includes a front-end interface that allows users to input their search criteria and view the results in real-time.

3 Methodology

The application is built using Python Flask Framework, which allows for easy integration with Elasticsearch and MongoDB. A MongoDB database was used to store the restaurant data, which was then transformed to be indexed into Elasticsearch.

The Elasticsearch index was created from Mongo BD with the following mapping properties: "name", "address", "cuisine", "borough". The "address" field contains sub-properties for "building", "coord", "street", "zipcode", and "borough".

When a user inputs a cuisine type into the search bar, the application queries Elasticsearch for restaurants matching the search criteria. The results are then displayed on a search results page in an HTML template.

The HTML templates were designed to be simple and user-friendly. The home page displays a search bar for the user to input their search criteria. The search results page displays the restaurant results in a clean, organized manner. The HTML templates were styled using CSS to improve the visual appearance of the pages.

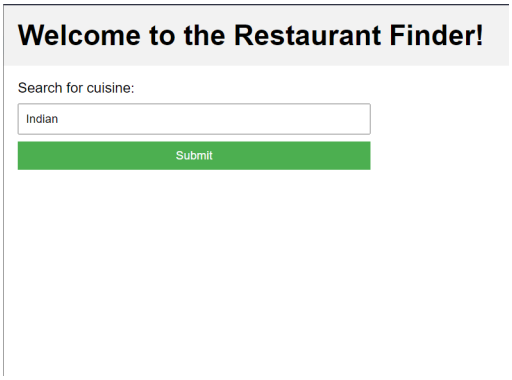
4 Code

```
from flask import Flask, request, render_template, jsonify,url_for,redirect
from pymongo import MongoClient
from elasticsearch import Elasticsearch

# Set up MongoDB client

connection_string = STRING HERE'
mongo_client = MongoClient(connection_string)
mongo_db = mongo_client['sample_restaurants']
mongo_collection = mongo_db['restaurants']

# # Set up Elasticsearch client
es_client = Elasticsearch(
    hosts=[{'host':'localhost','port':9200,'scheme':'https'}],
    basic_auth= ('USERNAME','PASSWORD'),
    verify_certs=False)
index_name = "restaurant"

index_body = {
    "mappings": {
        "properties": {
```

```
            "name": {"type": "text"},
            "address": {
                "properties": {
                    "building": {"type": "text"},
                    "coord": {"type": "geo_point"},
                    "street": {"type": "text"},
                    "zipcode": {"type": "text"},
                    "borough": {"type": "text"}
                }
            },
            "cuisine": {"type": "text"}
        }
    }
}
if not es_client.indices.exists(index = index_name):
    es_client.indices.create(index=index_name, body=index_body)
    Mentries = mongo_collection.find()

    documents = []
    for entry in Mentries:
        doc = {
        "name": entry["name"],
        "borough": entry["borough"],
        "cuisine": entry["cuisine"],
        "address": {
            "building": entry["address"]["building"],
            "street": entry["address"]["street"],
            "zipcode": entry["address"]["zipcode"]
        }
        }
        documents.append(doc)
    # Index transformed data into Elasticsearch index
    for i, doc in enumerate(documents):
        es_client.index(index=index_name, id=i, body=doc)

# Set up Flask app
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('home.html')

@app.route('/all')
def dispAll():
    entries = mongo_collection.find()
    return render_template('entries.html', entries=entries)

@app.route('/submit', methods=['POST'])
```

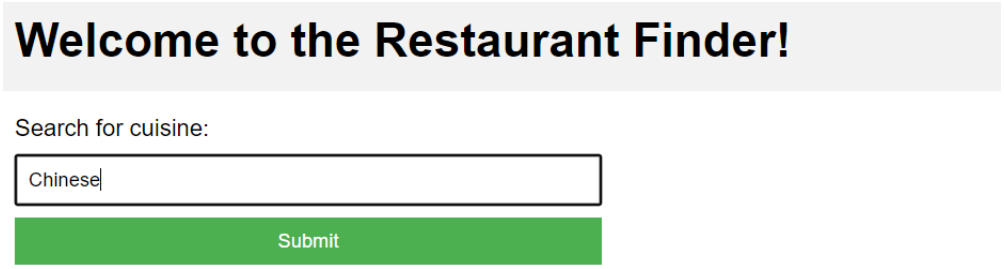
```
def submit():
    query = request.form['query']
    return redirect(url_for('search',query=query))

@app.route('/search', methods=['GET', 'POST'])
def search():
    data = request.args.get('query')
    query = {
        "query": {
            "match": {
                "cuisine": data
            }
        }
    }
    result = es_client.search(index=index_name, body=query)
    hits = result["hits"]["hits"]
    response = [
        {
            "name": hit["_source"]["name"],
            "cuisine": hit["_source"]["cuisine"],
            "borough": hit["_source"]["borough"],
            "address": hit["_source"]["address"]
        }
        for hit in hits
    ]
    return render_template('entries.html', entries=response)
    # return jsonify({"restaurants": response}), 200

if __name__ == '__main__':
    app.run(debug=True)
```

5 Results



Name	Borough	Address	Cuisine
Santoor Indian Restaurant	Queens	257-05, Union Turnpike, 11004	Indian
Gandhi	Manhattan	345, East 6 Street, 10003	Indian
Indigo Indian Cuisine	Manhattan	357, East 50 Street, 10022	Indian
Bangal Curry	Manhattan	27, Park Place, 10007	Indian
Taste Of India Ii Restaurant	Staten Island	287, Newdorp Ln, 10306	Indian
Haveli	Manhattan	100, Second Avenue, 10003	Indian
Air India Lounge	Queens	4, - Jfk Airport, 11430	Indian
Diwan-E-Khaas	Manhattan	2628, Cedar Street, 10005	Indian
Pongal Indian Cuisine	Manhattan	110, Lexington Avenue, 10016	Indian
India Passage Restaurant	Brooklyn	7407, 3 Avenue, 11209	Indian

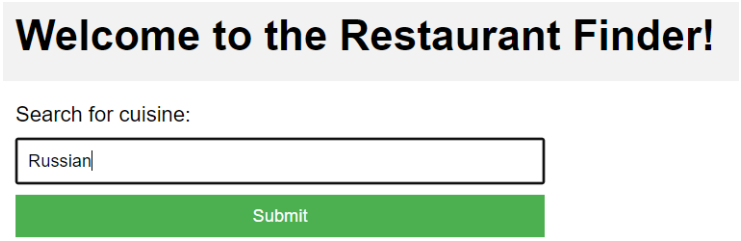
Figure 1: Search Restaurants with Indian Cuisine



My MongoDB Entries

Name	Borough	Address	Cuisine
Joe'S Shanghai Restaurant	Manhattan	9, Pell Street, 10013	Chinese
New Feng'S Garden	Queens	88-26, Parsons Boulevard, 11432	Chinese
Jj Garden Restaurant	Queens	75-21, 31 Avenue, 11370	Chinese
Peking Duck House	Manhattan	236, East 53 Street, 10022	Chinese
New Wah Kitchen	Bronx	1474, Westchester Avenue, 10472	Chinese
Island Taste	Staten Island	3161, Amboy Road, 10306	Chinese
Nueva Villa China Restaurant	Queens	3910, 103 Street, 11368	Chinese
Lee'S Villa Chinese Restaurant	Brooklyn	152, Lawrence Street, 11201	Chinese
Choy Le Chinese Restaurant	Brooklyn	425, Avenue U, 11223	Chinese
S Dynasty	Manhattan	511, Lexington Avenue, 10017	Chinese

Figure 2: Search Restaurants with Chinese Cuisine



My MongoDB Entries

Name	Borough	Address	Cuisine
Uncle Vanya	Manhattan	315, West 54 Street, 10019	Russian
National Restaurant & Catering	Brooklyn	273, Brighton Beach Avenue, 11235	Russian
Cafe Paris	Brooklyn	3178, Coney Island Ave, 11235	Russian
Russian Samovar	Manhattan	256, West 52 Street, 10019	Russian
Russian Turkish Baths	Manhattan	268, East 10 Street, 10009	Russian
Cafe Volna	Brooklyn	3145, Brighton 4 Street, 11235	Russian
Anyway Cafe	Manhattan	34, East 2 Street, 10003	Russian
Tatiana Restaurant	Brooklyn	3152, Brighton 6 Street, 11235	Russian
Russia Vodka Room	Manhattan	265, West 52 Street, 10019	Russian
Varenichnaya - Pelmini Varenichky	Brooklyn	3086, Brighton 2 Street, 11235	Russian

Figure 3: Search Restaurants with Russian Cuisine

6 Conclusion

This restaurant finder application is a useful tool for users looking to discover new restaurants based on cuisine preferences. The Elasticsearch index ensures that search queries are fast and accurate, while the MongoDB database provides a reliable source of restaurant data. The simple and user-friendly design of the application allows for easy navigation and a positive user experience. Overall, this project successfully demonstrates the use of Elasticsearch and MongoDB in a Flask application for finding restaurants by cuisine.

7 Appendix

A Project Link

https://github.com/bishal216/RandomStuffs/tree/main/BigData/Project

B MongoDB data sample

Cuisine	**Name**	**Borough**	**Address**	**...**
American	The Capital Grille	Manhattan	120 W 51st St	...
Chinese	Joe's Shanghai	Queens	136-21 37th Ave	...
Italian	Carbone	Manhattan	181 Thompson St	...
Japanese	Sushi Yasuda	Manhattan	204 E 43rd St	...
Mexican	Cosme	Manhattan	35 E 21st St	...
Pizza	Di Fara Pizza	Brooklyn	1424 Ave J	...
Thai	SriPraPhai	Queens	64-13 39th Ave	...
Vegetarian	Dirt Candy	Manhattan	86 Allen St	...

Table 1: Sample Data from MongoDB

C Elasticsearch index Mapping

Field	**Data Type**	**Index**	**Description**
name	text	analyzed	Name of the restaurant
cuisine	text	analyzed	Type of cuisine served
borough	text	analyzed	Borough in which the restaurant is located
address	text	analyzed	Address of the restaurant
rating	float	not analyzed	Rating of the restaurant on a scale of 1-5
price	integer	not analyzed	Price level of the restaurant (1-4)
location	geo_point	not analyzed	Latitude and longitude of the restaurant

Table 2: Elasticsearch index mapping for restaurant data