Name: atharv sharma
email: atharv228.as@gmail.com

Testing the API

To ease the usage of the assessment without any setup, the project has been made live by hosting frontend on netlify and backend on vercel functions:

You can test the API by visiting the frontend:
https://hybrid-search.netlify.app

The backend is hosted on Vercel and can be accessed via the following endpoint:
https://hybrid-search-backend.vercel.app/api/elastic-search

Project Repositories

1. GitHub link for backend: https://github.com/atharvsharma1998/hybrid-search-backend
2. GitHub link for frontend: https://github.com/atharvsharma1998/hybrid-search-app
3. Github Link for creating tables/documents: https://github.com/atharvsharma1998/python-scripts

Technology Stack

This project utilises Elasticsearch for its search capabilities, making it an excellent choice for building a scalable search backend without the need to develop complex indexing and search algorithms from scratch. Elasticsearch also offers powerful data ingestion pipelines and connectors, which can seamlessly integrate with existing backends like PostgreSQL, MongoDB, and other popular cloud databases.

Hybrid Search Implementation

The hybrid search functionality combines full-text search with vector search (using k-NN). For ranking search results, the Reciprocal Rank Fusion (RRF) method is utilised. The api performs full text search on author, title and content fields and vector search over vector_representation. Elasticsearch provides a comprehensive SDK that simplifies the implementation of these advanced search features. While it's possible to build similar functionality using PostgreSQL, leveraging Elasticsearch's built-in capabilities offers significant advantages in terms of development speed and flexibility.

Optimisation considered:

1) Database Restructuring:
Given that Elasticsearch uses document-based storage rather than traditional relational databases, the backend schema was restructured to effectively map magazine content to magazine information. Below is the schema used for the magazine data:

```
{

"title": "text",
"author": "text",
"publication_date": "date",
"category": "text",
"content":
        {
                “text": “text",
                "vector_representation": "dense_vector(dims: 1536)"
        }
}
```

2) Pagination:

Elasticsearch natively supports pagination, which optimizes query performance by returning data in manageable chunks. This built-in feature reduces the need for developers to implement custom pagination logic, simplifying the development process and improving overall efficiency.

Performance Report:

Currently, there are around 30000 documents indexed in the Elasticsearch database. During testing, the maximum response time observed was approximately 120ms, as reported by the Elasticsearch API. It's important to note that some delays may occur due to the frontend and Vercel functions being hosted in different regions, which could be optimized through better hosting strategies.

Search Performance
The search functionality involves querying with random vectors, which occasionally results in increased search times. This can also affect the quality of the matches received.

API Request and Response Example in postman:

You can make post request to the url: https://hybrid-search-backend.vercel.app/api/elastic-search

Request:
To perform a search using both a keyword query and a 1563-dimension vector, send a POST request to the endpoint with the following JSON payload:

When sending the request:

```
{
   "query": "example search keyword",
   "vector": [0.376, 0.957, …, , 0.396,] // 1563 dimension vector
}
```

Response:
The API responds with a JSON object containing search results, including the id, title, author, content, and relevance score for each result. Additionally, it includes the timeTaken field, which indicates the time taken by the server to process the request (in milliseconds).

```
{
   "results": [
      {
         "id": "2328",
         "title": "Example example situation director seven news.",
         "author": "Pamela Farrell",
         "content": [
            "Choose investment bad result recently total others not. Window by respond ready understand. Since hit five task yard student ready perhaps. Job investment girl the. Prevent finally step.",
            "Former indeed pattern people since change. American group war. He first since what significant. Answer employee majority particularly season none. Appear college campaign letter."
         ],
         "relevance": 0.04761905 // score for the search
      },
      // Additional results...
   ],
   "timeTaken": 14  // Time taken in milliseconds
}
```