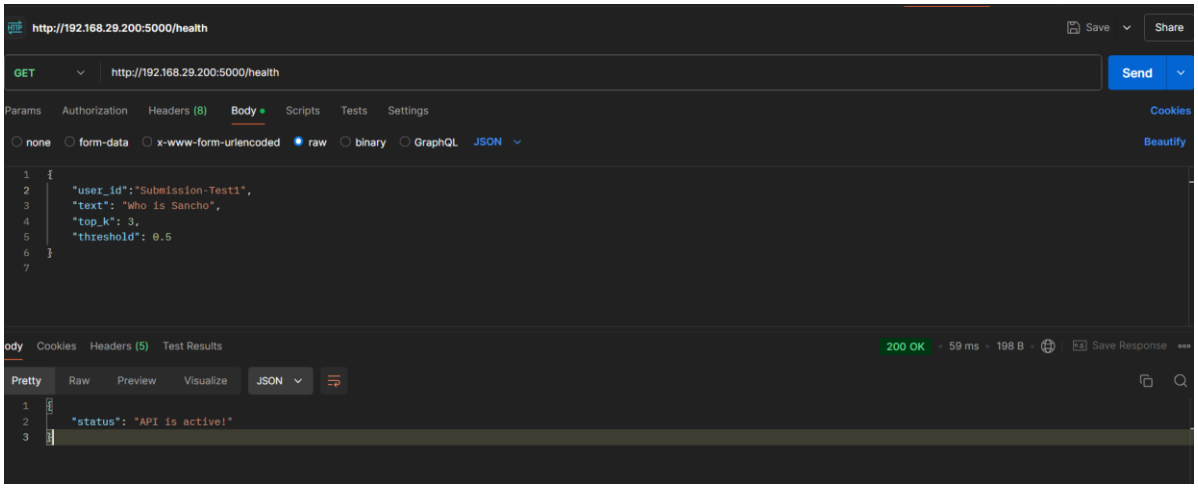
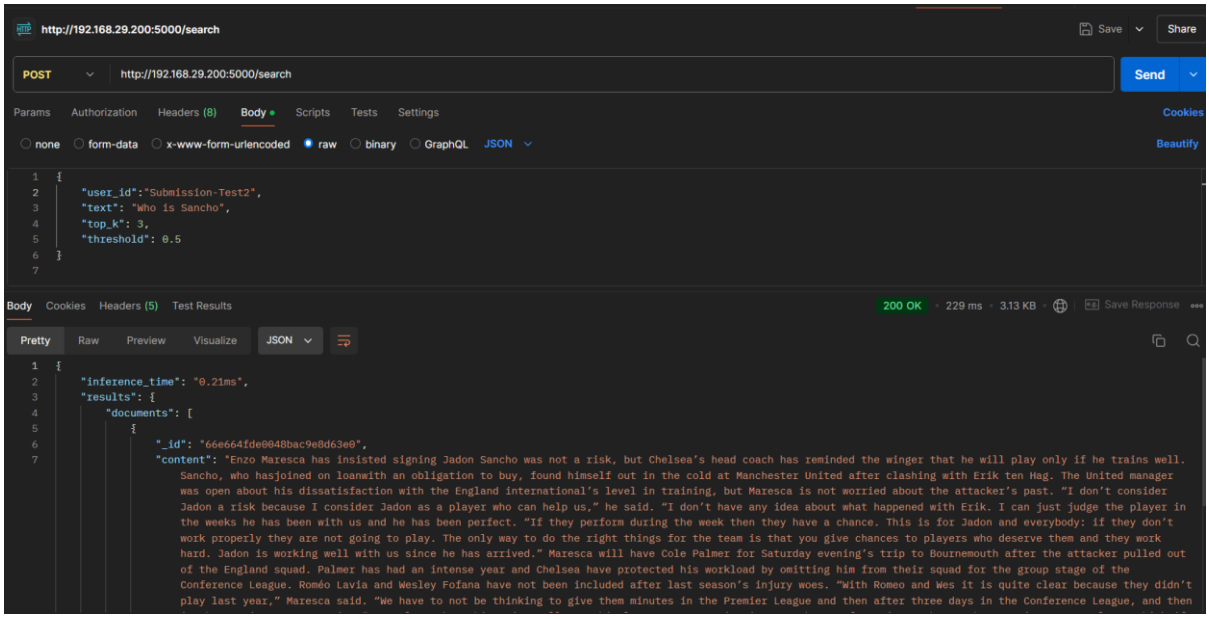
DOCUMENT - RETRIEVAL SYSTEM

1. API Design

- `health` endpoint:
  Checks if the API is running or not



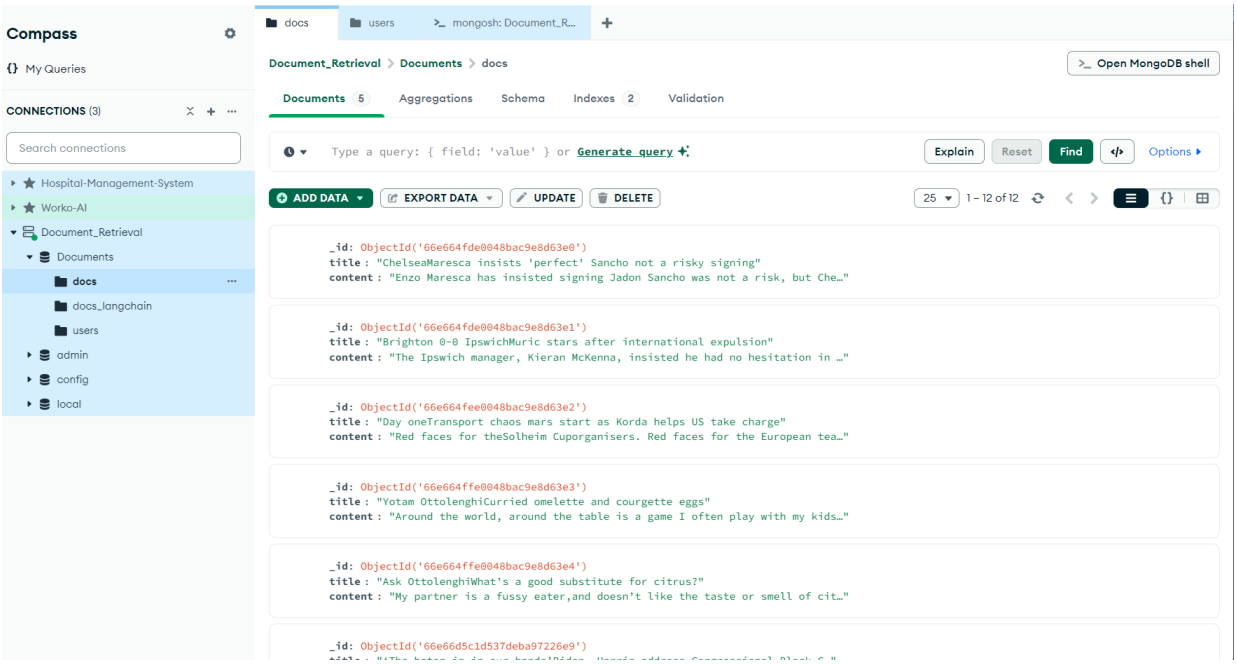
- `search` endpoint
  Returns the results for the query along with inference time



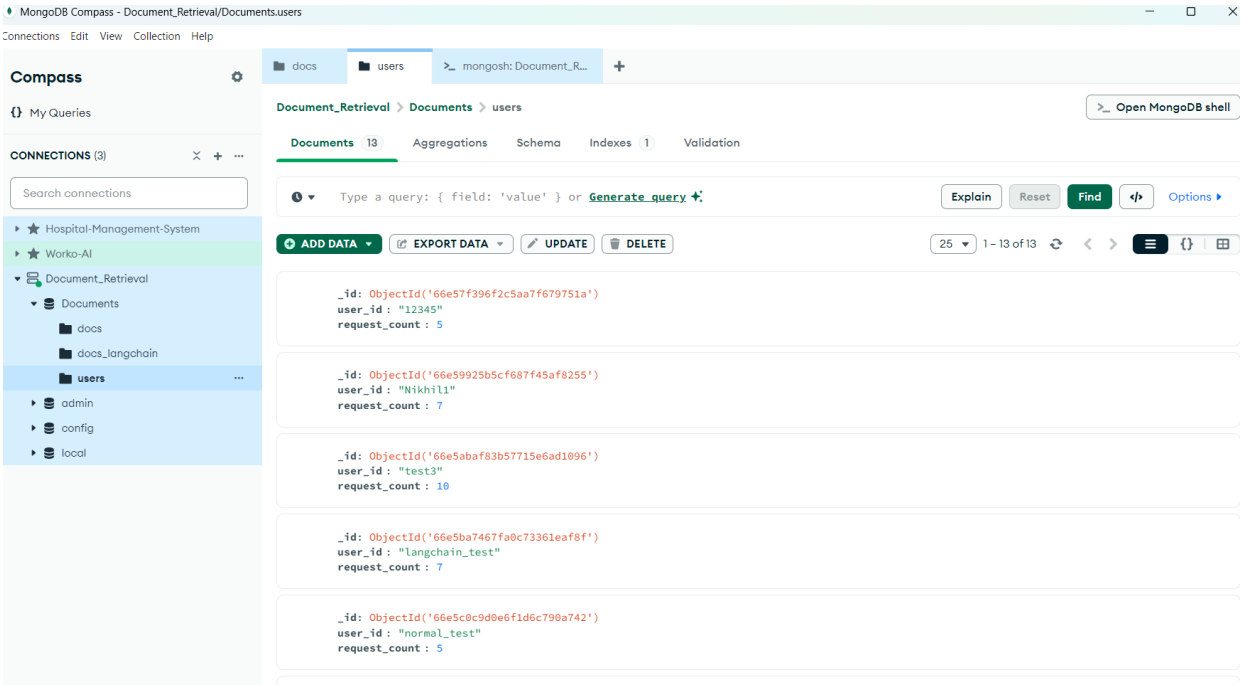
2. Database Interaction

- The database used is **MongoDB** document database
- Two collections namely docs and docs and users are created. Docs containing the news articles web scraped from **The Guardian News**, the thread for which will be running in the background when server runs.
- Users collection contains the user_id of the Users and the no. of requests the user gave for the search endpoint of the API.

- Docs Collection



- Users Collection



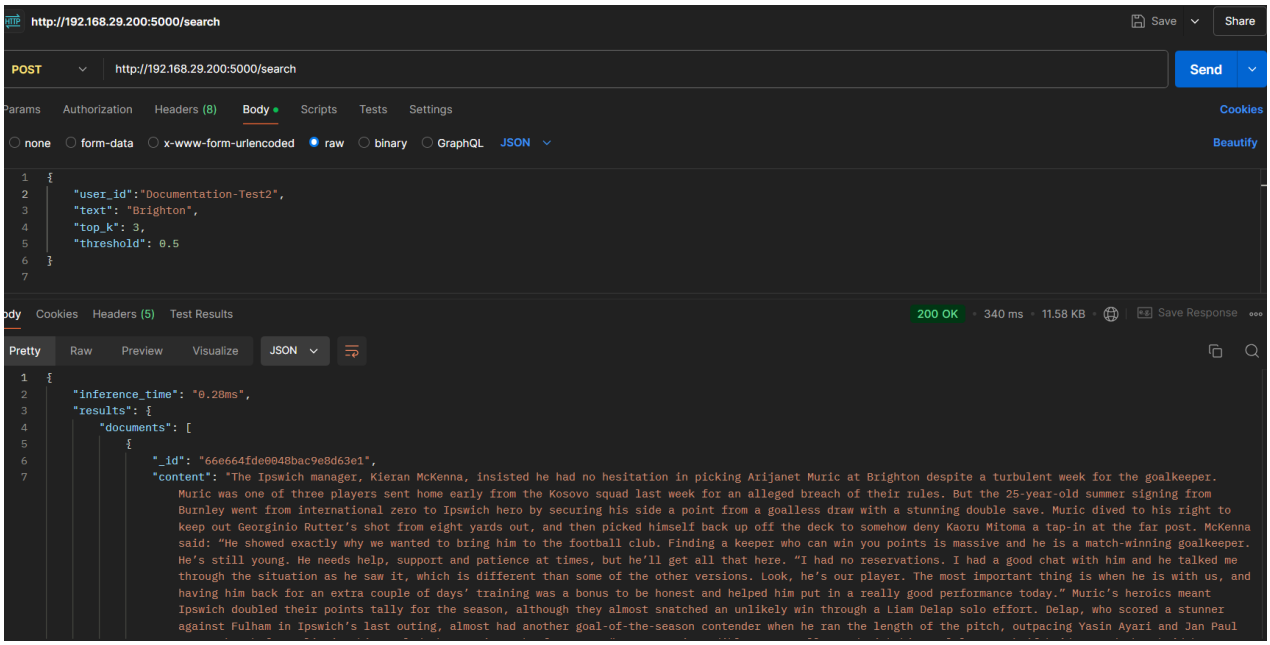
3. Concurrency

Threads has been used to scrape through the News Website and add the scraped content to the database.

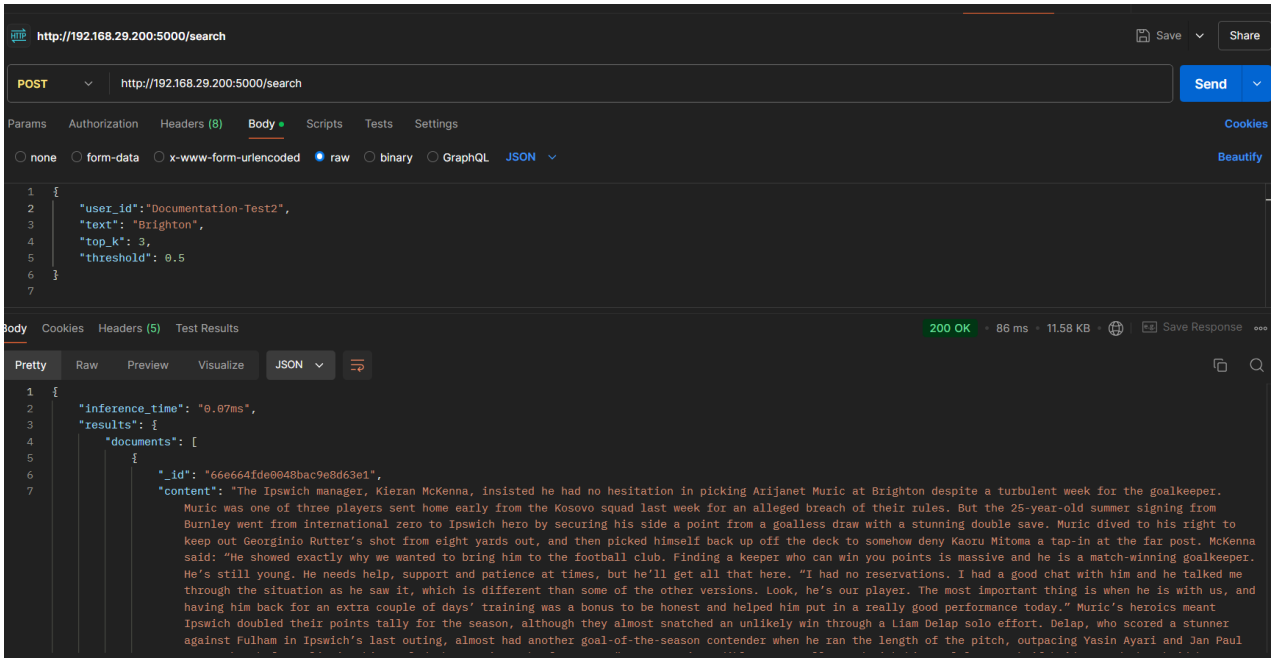
4. Caching and Optimization

This project uses Redis as the caching mechanism to improve the performance of a document retrieval system. Caching is essential in reducing the load on the database and improving the overall speed of repeated queries. This section explains why Redis was chosen and the caching method implemented.

In-Memory Storage for High Performance: Redis stores data in memory, which allows for incredibly fast data access. This makes it an ideal choice for caching, as results can be retrieved much faster than querying a database. Since search results in our document retrieval system may be requested multiple times, the speed advantage of Redis significantly reduces the response time for repeated queries.

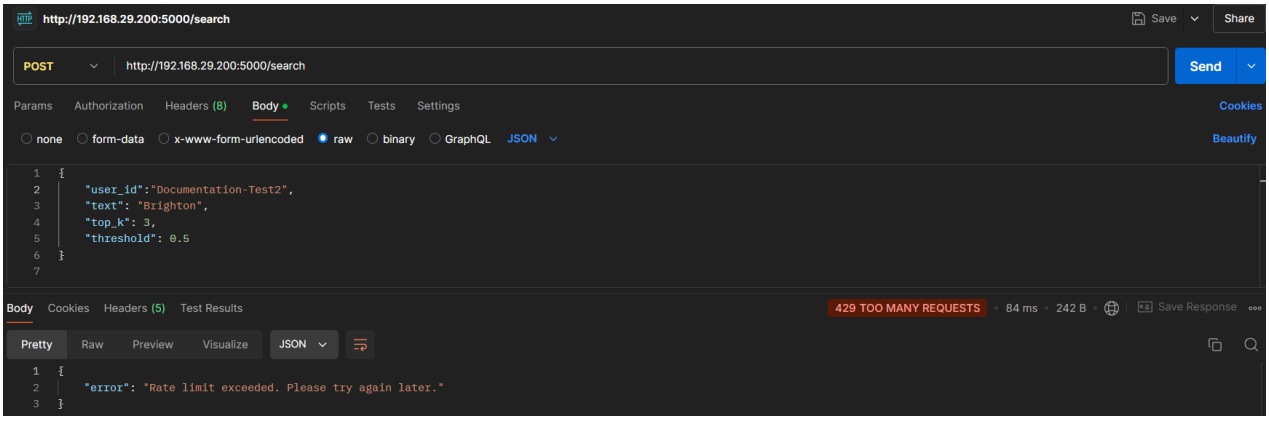


The inference time is 0.20ms, for the first search. Now the search results would be stored in the cache memory and on subsequent searches, the server would fetch the results from the cache memory, reducing the time and optimizing the searching and querying strategies.



On subsequent search, the inference time is reduced drastically, inferring that the results have been fetched from the cache memory.

5. **Error Handling and logging**
   - Rate limits:
     If a user searches the same query for 5 times in a row in a time window of 60 seconds, rate limit will be exceeded and the server returns a 429 error.



   - Logging:
     The server logs the API requests made by the user in a separate database named users. It stores the user id and the number of requests made by the user.
   - Error Handling:
     Proper error-handling mechanisms have been implemented, and all the edge cases have been taken account of.