University of Pittsburgh

INFSCI 2140– Information Storage and Retrieval (Fall 2024)

School of Computing and Information

Term Project

Verse Vibe – Find and Play Songs By Lyrics

Varun Shelke

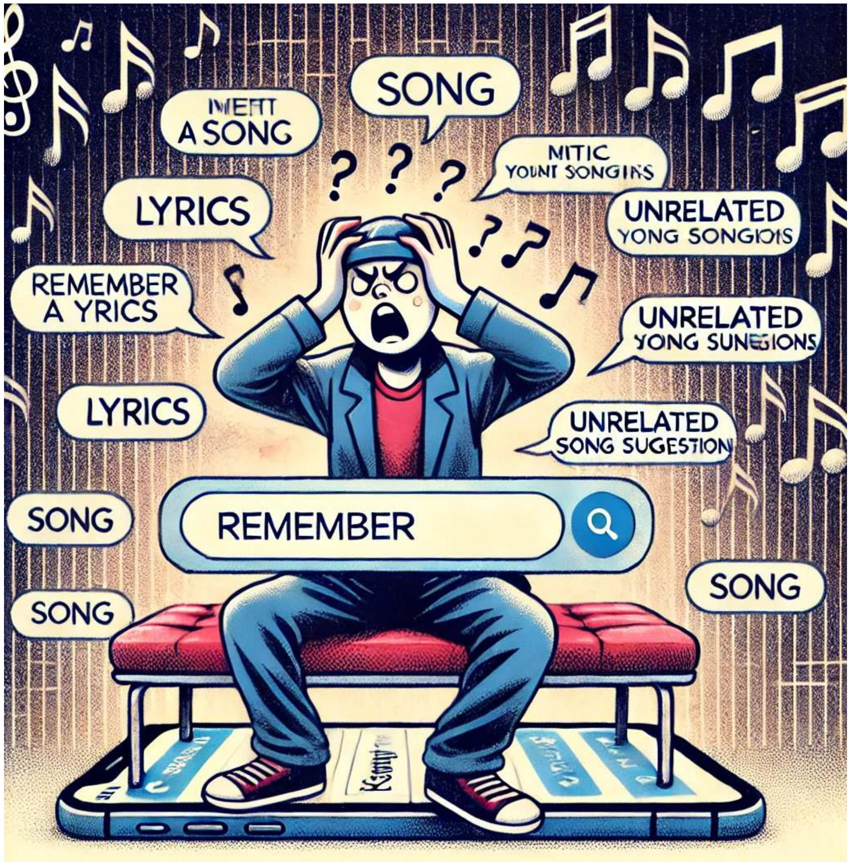
Mohammed Misran

Satvik Tandon

December 13, 2024

Problem Statement

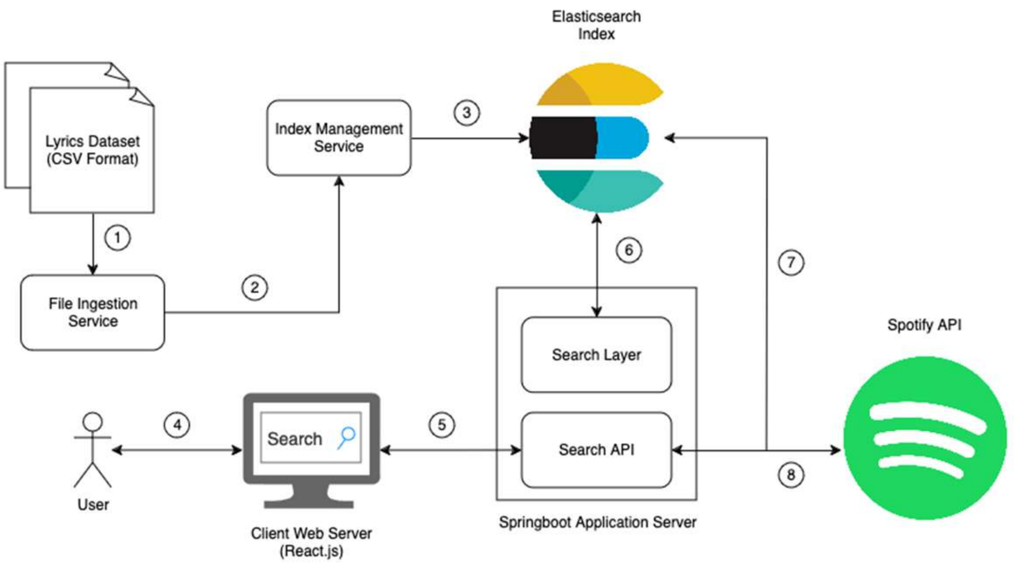
- Many music listeners often remember only parts of a song's lyrics but struggle to identify the song or artist.
- Problems?
  - ○ **Difficulty in Song Identification**
  - ○ **Lack of Specialized Tools**
- What Verse Vibe does?
  - Helps users find songs by inputting fragments of lyrics.
  - Provides a streamlined solution for effortless song discovery.



Solution and Resources

1. File Ingestion Service pre-processes the documents given in CSV format (Kaggle Genius Song Lyrics Dataset).
2. Index Management Service consumes the pre-processed documents creates an index in Elasticsearch.
3. Created index is stored in the Elasticsearch datastore.
4. End-user searches by entering part of the lyrics in the search box.
5. Query is sent to the application server via the Search API.
6. Search Layer is responsible for querying the Elasticsearch index to find the matching documents.
7. Spotify API is used to fetch the related songs' metadata.
8. The relevant song along with its metadata is returned to the end user.

VerseVibe - Architecture Diagram



Elastic Search Index

What is Elasticsearch?

- Distributed, RESTful search engine.
- Handles structured and unstructured data efficiently.

Objective:

- Index and search song data.
- Enhance user experience with relevant and optimized search.

elasticsearch

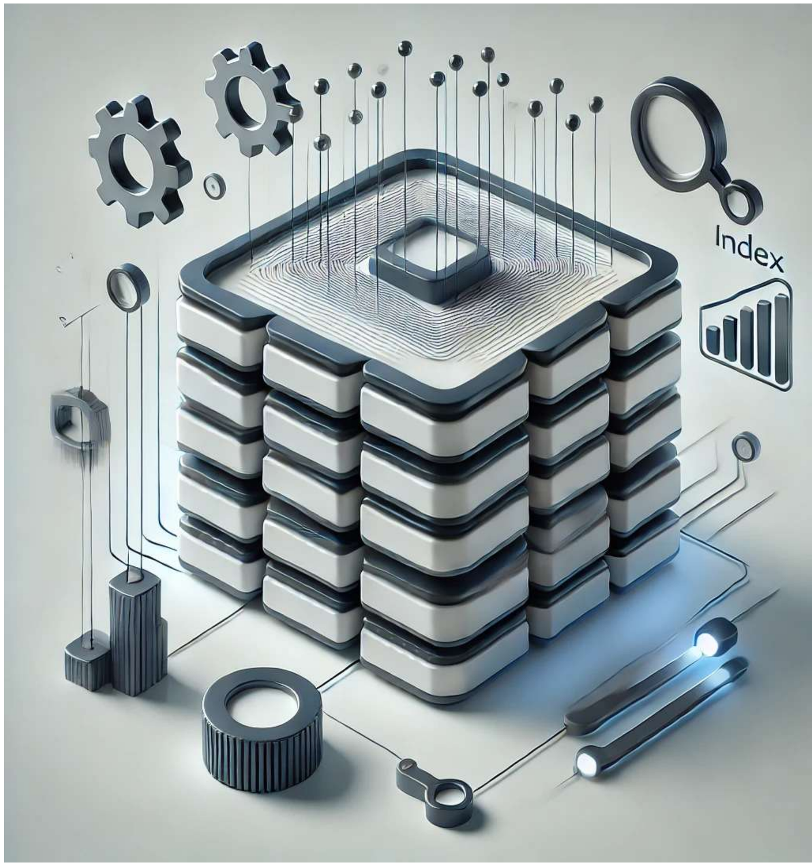
Index Configuration

Settings:

- Custom Song Analyzer
- Tokenizer: standard
- Filters:
  - Lowercase
  - ASCII folding
  - Porter stemmer

Mappings:

- Title, Artists, and Lyrics
- text' type with custom 'song_analyzer'
- Keyword sub-field for exact matching
- Non-indexed fields: tag, id, language.



elasticsearch

University of Pittsburgh

Index Management Service

Features

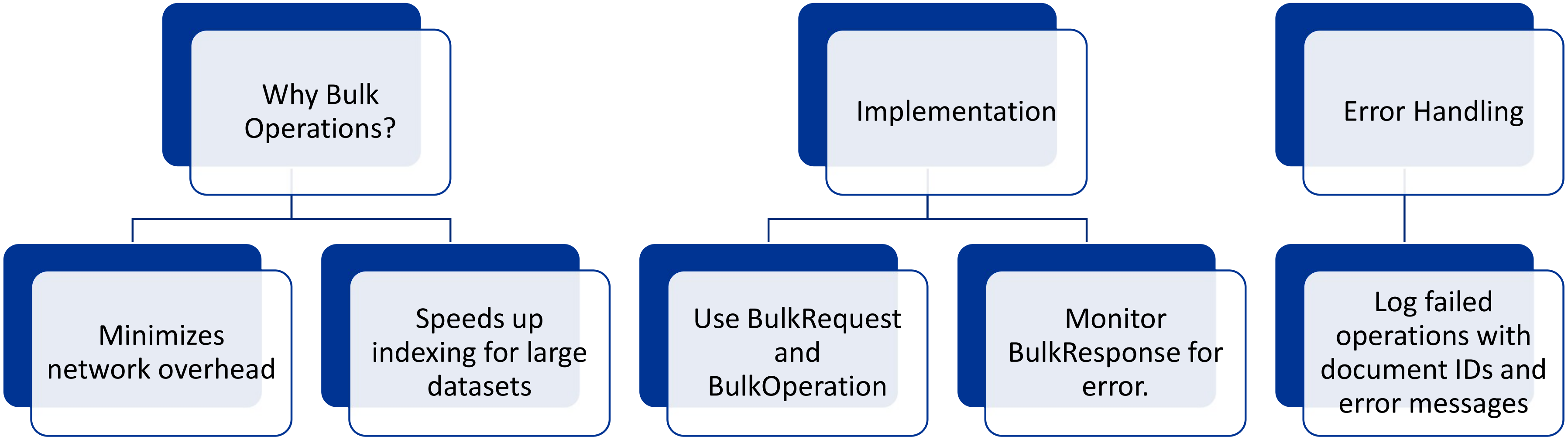
- Dynamic index creation with JSON mappings.
- Bulk indexing for high performance.
- Deletes and re-creates indices for schema updates.

Key Methods

- createIndex(): Checks and creates an index.
- deleteIndexIfExists(): Removes stale indices.
- bulkIndex(): Indexes documents in batches.

University of Pittsburgh

Bulk Operations



Why Bulk Operations?

- Minimizes network overhead
- Speeds up indexing for large datasets

Implementation

- Use BulkRequest and BulkOperation
- Monitor BulkResponse for error.

Error Handling

- Log failed operations with document IDs and error messages

Search Service

Features:

- Multi-field search across 'title', '**lyrics**^1.5', and 'artist'

Query Workflow:

- Accept user input (search term)
- Build MultiMatchQuery
- Fetch results from Elasticsearch
- Extract relevant fields from hits
- Example Query: ‘I tried so hard and got so far'

University of Pittsburgh

Spotify Service

Purpose

- Enhance the dataset with real-time API data

Features

- Dynamic search queries for track and artist
- Authorization via Spotify tokens

Workflow

- Build query using track and artist names
- Send HTTP GET request to Spotify API
- Parse JSON response for track metadata

Scope for Future Enhancements

- Error handling while bulk indexing
- Multi-document support
- Multi-source document support
- Multi-language support
- Advanced filtering techniques (such as n-gram analyzer) to optimize phrase matching
- Search highlighting for matched lyrics content

University of Pittsburgh

School of Computing and Information

Thank you