Buzzdiggr news feed API

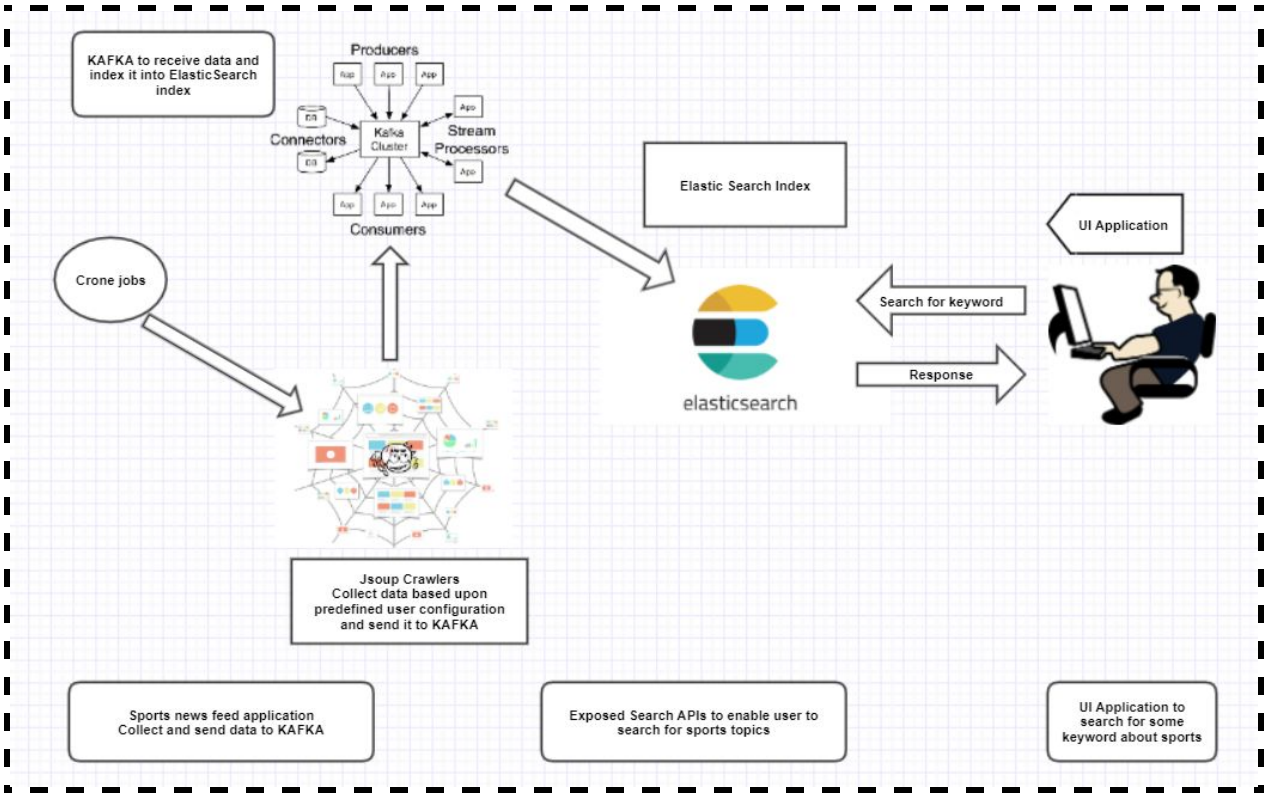
Overview

System to collect sports news feed from different sources, indexing these data and make it searchable through REST APIs.

System design

The system is composed of two running processes, first of them is responsible for collecting data through crawlers and send these data to KAFKA. Then the second process receives these data and index it into ElasticSearch index, Then the user can use buzzdiggr ui application to search about sports content with some keywords.

Below is a mock of the current implemented data flow:



We will include these two systems into the production microservices solution to leverage the benefits of configuration management, load balancing, clustering, auto-scaling, service registery and internal load balancing .

What is missing to complete the system design picture

1. Load balancer to distribute traffic.
2. Every system will be an instance of cluster which will scale up and down according to traffic.
3. Need to add akka actors and kafka streams to process data in a real time flavor.
4. Application logs need to be sent to kafka to use them by many consumer such as files which will be processed by big data tools to get useful reports and track user behaviors, also we can index logs to make use of data visualization by ELK.
5. Consider using python for crawling instead of java jsoup to increase data flow and for more accurate data.
6. The application will be fully dockerized to integrate with CI/CD tools.
7. Using openAPI for facilitating the orchestration development of APIs between different teams.
8. Using sonar qube for code quality.

System Technical aspects

- ## Buzzdiggr news feed

  The system uses

  1. Spring boot for fast development and easy to integrate with other frameworks.
  2. Crone jobs to start crawlers to collect data from websites.
  3. Lombok to increase code readability and offers some ready features like builder, getters and setters.
  4. Kafka to send data through topics to be indexed in Elasticsearch.

- ## Buzzdiggr Search API

  The system uses

  1. Spring boot for fast development and easy to integrate with other frameworks.
  2. Lombok to increase code readability and offers some ready features like builder, getters and setters.
  3. Kafka to index data into Elasticsearch index.
  4. Elasticsearch to hold searchable content of news.
  5. Exposed APIs to search for sports news feed.

- ## Tech choices

  1. Spring boot (Rest - Kafka - Elastic - actuator - profiling)
  2. Swagger for API documentation.
  3. Java 8.
  4. Jsoup for crawling and parsing html pages.
  5. Maven for dependency management.
  6. Docker for portability of application.
  7. Angular for searching and presenting data.
  8. Kafka as a message broker.
  9. Elastic as a text search engine.
  10. Profiling and environment variables for portability and easy provisioning of new machines (auto-scaling).

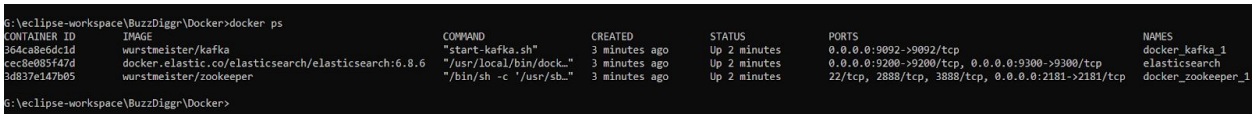
Get started

- ## Prerequisites:
    1. JDK 1.8
    2. Docker
    3. Docker compose
    4. Angular CLI: 8.3.20
    5. Node: 10.16.0
- ## Clone github repository:

  “git clone https://github.com/mahmoudhakam/buzzdiggr-newsfeed.git”
- ## Define environment variables:

  “ACTIVE_ENV” → (dev or prod) for profiling
  “Feeds_LOG_PATH” → log path for new feeds system
  “SEARCH_LOG_PATH” → log path for new search system
- ## Making the application up and running:

1. Navigate to Docker folder then execute “docker-compose up -d” then wait until all docker containers are built, then make sure that all containers are up and running, by command “docker ps”



2. Navigate to folder “BuzzdiggerSportsFeeds” then execute command “mvn clean install” then go to the target folder and execute “java -jar BuzzdiggrSportsNewsFeed.jar”. Make sure it is up on port “8011”
3. Navigate to folder “BuzzdiggerSportsSearchAPI” then execute command “mvn clean install” then go to the target folder and execute “java -jar BuzzdiggrSearchAPI.jar”. Make sure it is up on port “8010”
4. Navigate to folder “buzzdiggr-ui” then execute command “npm i” then execute “npm start”. It will navigate you to http://localhost:4200/login
5. Then you can search with some of these word as it is configured to simulate user configuration .

“رياضة” - "ليفربول" - "محمد صلاح" - "كورة" - "زمالك" - "برشلونة" - "ريال مدريد" - "سوسيداد" - "الكلاسيكو"