



searchgoose

Team 9

201411296 이선명

201411312 장하나

201711375 권혁규

201711413 이유진

Overall

- 작품명: searchgoose
- 설명: go로 구현하는 RESTful 분산 검색엔진
- Input HW: Multiple Linux Servers
- Input SW: **bleve** (text indexing library), **Hadoop Yarn** (Cluster Coordination)
- 구현할 것
 - Document DB: 입력받은 데이터를 저장할 database.
 - Index DB: 빠른 검색을 위해 데이터를 가공해둬서 저장할 database.
 - Indexing Service: 입력받은 데이터를 가공하는 서버 모듈
 - Query Executor: Index DB에 쿼리후 Document DB에서 데이터를 가져올 서버 모듈
 - HTTP API: 데이터 저장 / 조회 / 검색 API 모듈

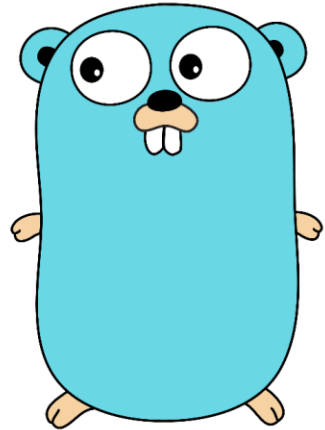
Description - 검색엔진?

- Full text search 가능한 일종의 NoSQL
- 왜 / 어떤 상황에서 필요한가.
 - 기존 Relational Database의 한계
 - 테이블 간 관계의 이유로 샤딩(분산)이 자유롭지 않음
 - 문자열 검색이 어려움.
-> SELECT * FROM table_1 LIKE '%test%' ?
특정 문자열이 포함되어 있는 것만 조회가 가능하다.
 - 검색엔진에선
 - 데이터를 전부 key:value로 저장,
데이터는 모두 독립되어 있으므로 key를 기준으로 해시, 샤딩이 자유롭게 가능
 - 문자열 검색을 위한 Indexing
-> Are you looking for a **deeper** understanding of the **Java** programming language
deeper와 java라는 keyword로 검색이 가능하게.

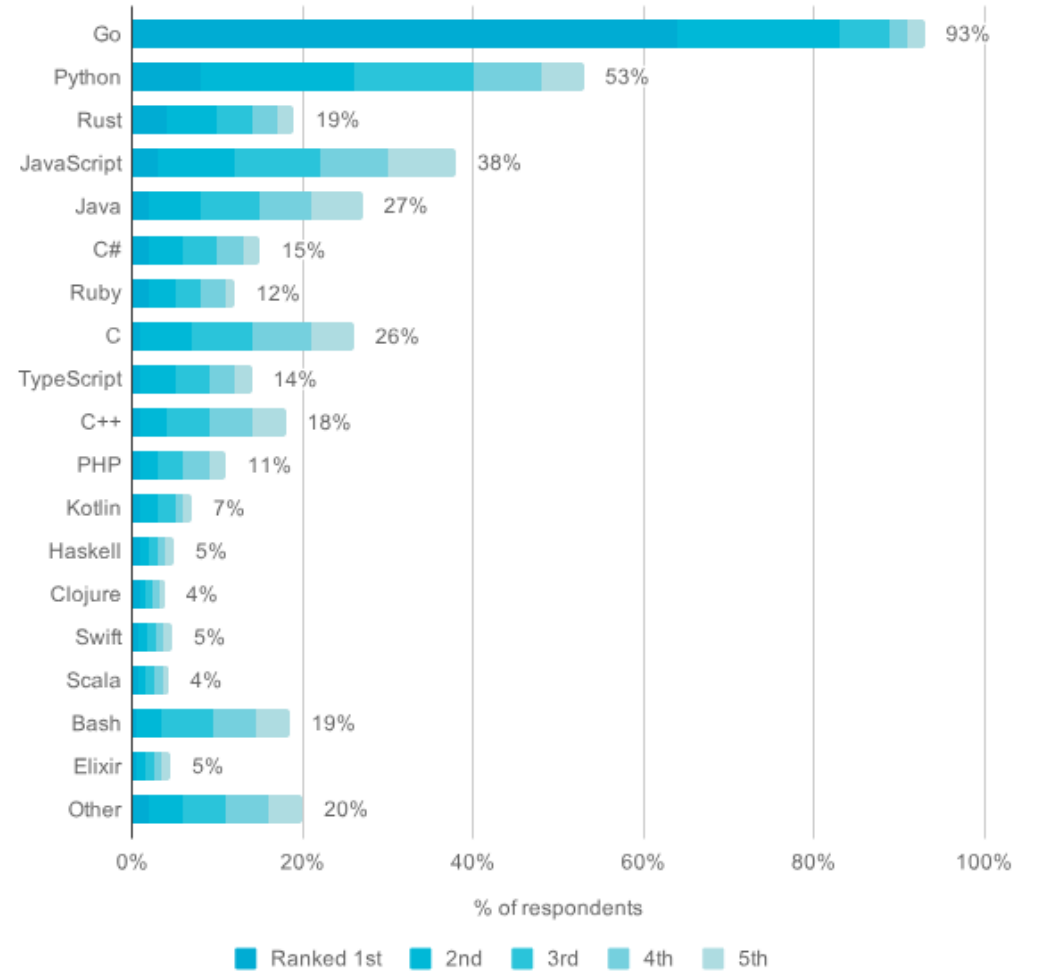
Go?



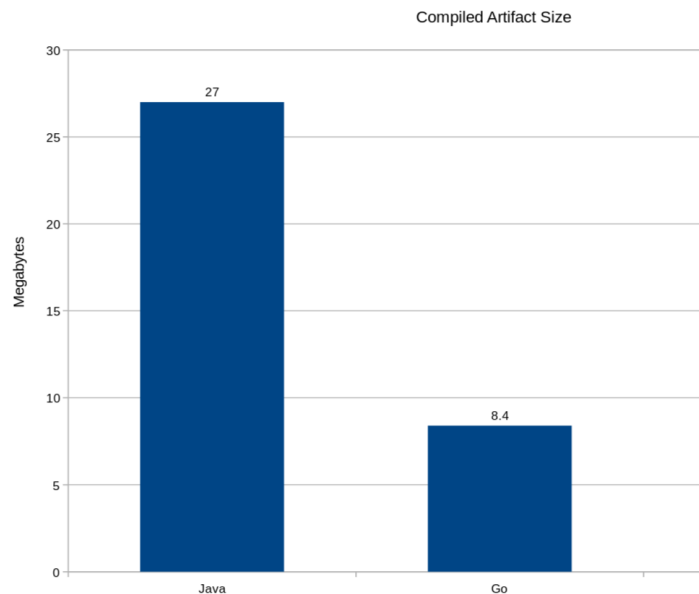
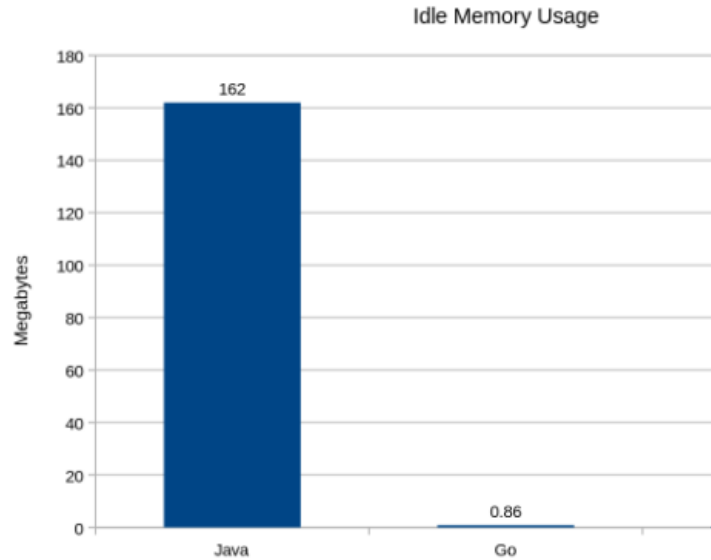
GO



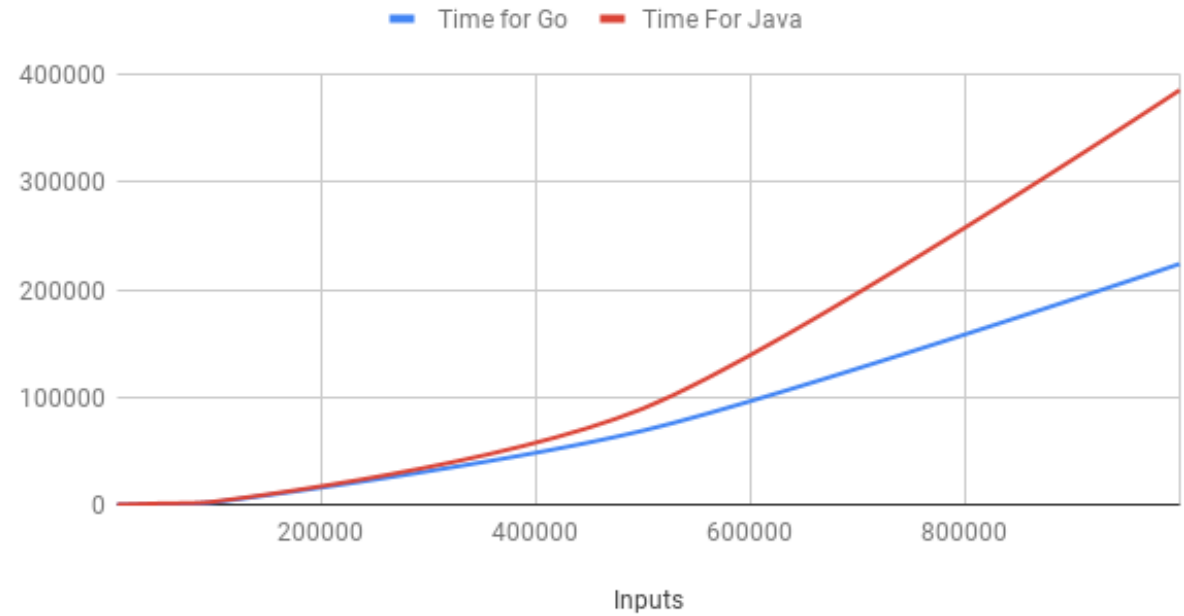
Rank the following languages in terms of your preference:



Java vs Go

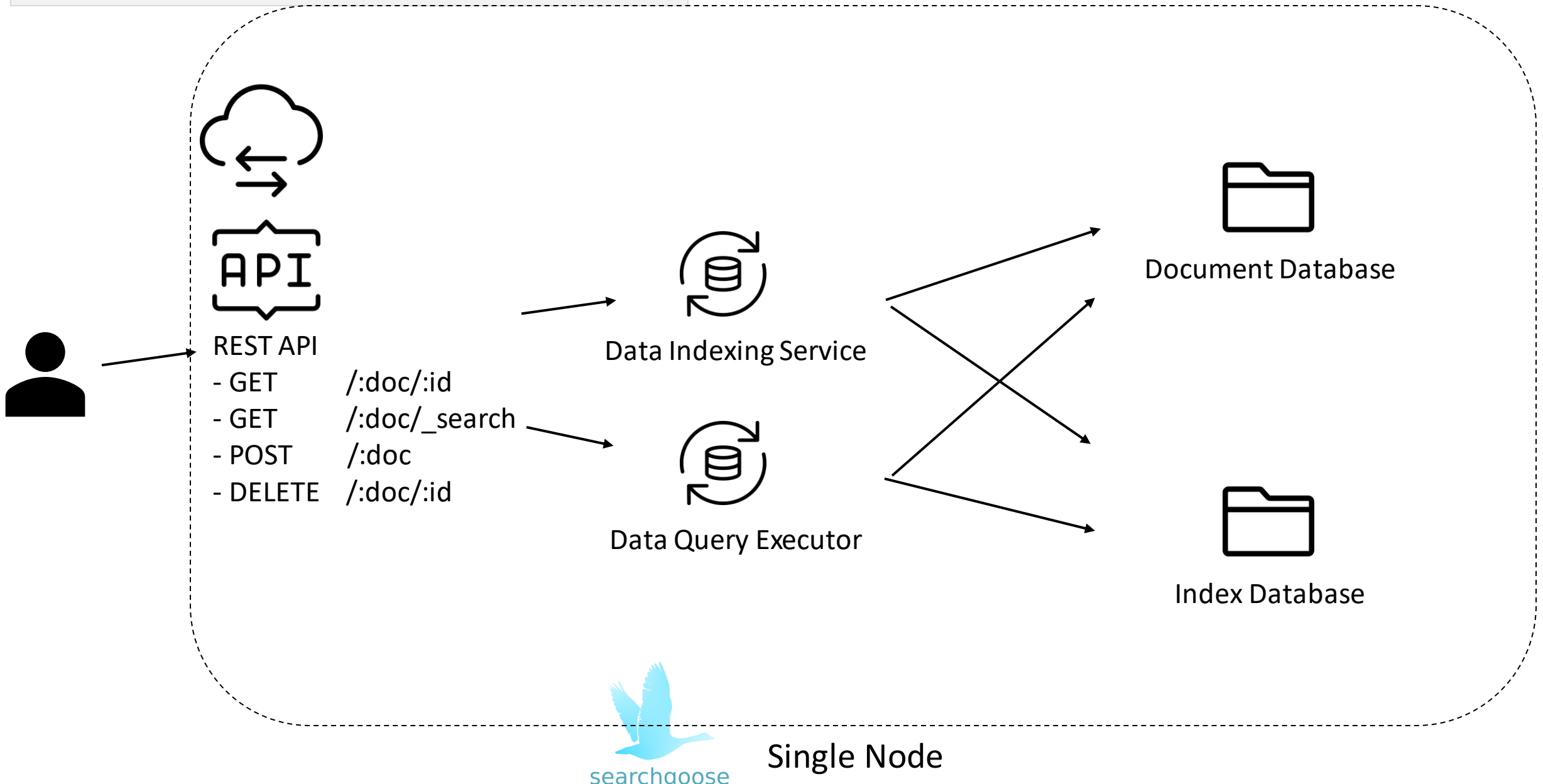


Time for Go and Time For Java

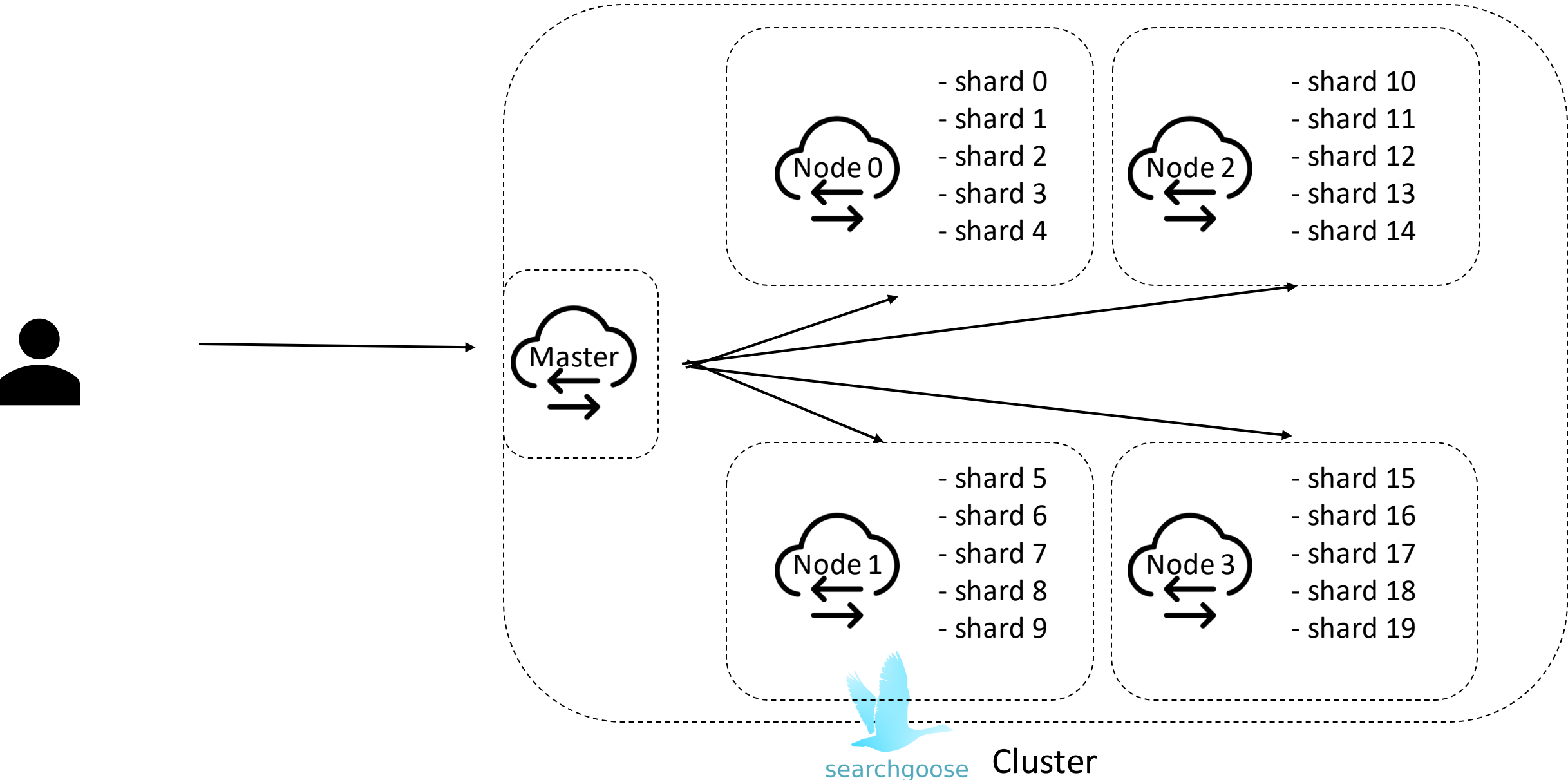


Go로 성능을 높여보자!

최종 산출물의 형태 및 기능



Distributed Server Architecture



Alternative Solutions: 오픈소스 검색엔진



Java로 구현되어 있음.

엘라스틱을 활용한 성공 사례

엘라스틱은 전 세계적으로 수많은 고객들과 다양한 사례들을 보유하고 있습니다. 고객 여러분들의 사용 목적에 맞는 사례를 확인해보시기 바랍니다.

A screenshot of the Elasticsearch website's case study section. It features a grid of case studies. The first case study is for Samsung SDS, titled 'ENTERPRISE SEARCH' and 'Samsung SDS'. The text describes how SDS used Elasticsearch for their chatbot, Brity. Below the case studies are two dropdown menus for '솔루션' (Solution) and '산업' (Industry). At the bottom, there are four case study cards for Sovren, Entel, Devsisters, and PSCU, each with their respective logos and names.

ENTERPRISE SEARCH

Samsung SDS

SDS가 생각하는 Chatbot의 방향을 설명하고 Elasticsearch 기반으로 구축된 삼성 SDS Chatbot Brity를 소개합니다.

[자세히 보기 >](#)

솔루션 ▾ 산업 ▾

sovren
Sovren

e) entel
Entel

DEVSISTERS
Devsisters

PSCU
PSCU

Risk Analysis

1. 구현할 건 많은 데 시간은 부족
Database, Indexing, Query executor, HTTP API ...
=> 오픈소스의 도움이 필요
2. Golang이라는 러닝커브
3. 분산시스템 구현의 어려움
=> 구현 성공사례 참조
4. 검색엔진에 대한 배경지식 부족
=> reference 참조
5. 데이터가 100만 개를 넘어갈 때도 1s 이내에 검색/저장할 수 있어야 한다.
=> 먼저 Basic Operation이 가능하도록 구현을 해보고, 테스트를 거치며 속도를 높여간다.
6. 한글을 포함해 다양한 언어를 지원하긴 어려움
=> 우선은 영어만 동작하도록 구현

Success Criteria

- Basic Operation

GET API - 문서 조회 API

POST API - 문서 생성 API

DELETE API - 문서 삭제 API

_SEARCH API - 문서 검색 API

분산환경에서 이러한 Basic Operation이 잘 작동해야함.

- Data fair storing / indexing

데이터를 각 노드에 최대한 fair하게 sharding