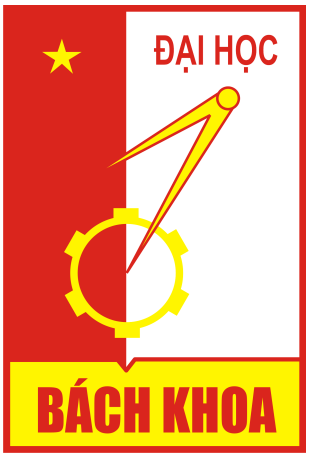
Đại học Bách khoa Hà Nội
Trường Công nghệ Thông tin và Truyền thông



Bài tập lớn

Môn: Tìm kiếm thông tin

Đề tài: Máy tìm kiếm chủ đề động vật

Giảng viên hướng dẫn:	TS. Nguyễn Bá Ngọc	
Nhóm PHM bao gồm:	Nguyễn Hải Minh	20210611
	Phạm Văn Phong	20215448
	Đỗ Văn Hoàng	20215378
Mã lớp:	154046	
Mã HP:	IT4863	

Hà Nội, tháng 12 năm 2024

Mục lục

Lời mở đầu

Tìm kiếm thông tin (Information retrieval, IR) là trích xuất các tài nguyên thông tin (thường là văn bản) có tính chất phi cấu trúc từ trong một nguồn tài nguyên thông tin lớn (thường được lưu trên máy tính), nhằm đáp ứng các nhu cầu thông tin.

Trong thực tế, các máy tìm kiếm được ứng dụng ở đa dạng các lĩnh vực: tìm kiếm sản phẩm hỗ trợ bán hàng, tìm hiểu thị trường, tìm kiếm tài liệu khoa học, hỗ trợ các hệ thống quản lí doanh nghiệp... Có thể nói các ứng dụng của lĩnh vực "Tìm kiếm thông tin" là nhiều vô kể.

Do đó, trong khuôn khổ bài tập lớn môn học "Tìm kiếm thông tin", nhóm PHM đã lựa chọn thử sức với đề tài **"Xây dựng máy tìm kiếm chủ đề động vật"**, nhóm thực hiện triển khai một máy tìm kiếm với bộ dữ liệu tập trung vào thông tin các loài động vật. Sau đó tiến hành đánh độ hiệu quả của các phương pháp triển khai khác nhau. Mục tiêu khi thực hiện bài tập là áp dụng các kiến thức đã được học để giải quyết vấn đề trong thực tiễn và đưa ra được sản phẩm có tính ứng dụng.

Báo cáo bao gồm 7 phần chính như sau:

- **Phần 1:** Giới thiệu bài toán
- **Phần 2:** Thu thập dữ liệu
- **Phần 3:** Cấu hình máy tìm kiếm
- **Phần 4:** Xây dựng hệ thống
- **Phần 5:** Đánh giá kết quả
- **Phần 6:** Kết luận
- **Phần 7:** Tài liệu tham khảo

.

1 GIỚI THIỆU BÀI TOÁN

Động vật đóng vai trò quan trọng trong hệ sinh thái, nghiên cứu khoa học, giáo dục và thậm chí là giải trí. Tuy nhiên, trong bối cảnh thông tin trên Internet ngày càng đa dạng và phức tạp, việc tìm kiếm thông tin chính xác về các loài động vật là một thách thức lớn. Các công cụ tìm kiếm thông thường không đủ chuyên sâu để đáp ứng nhu cầu này.

Một công cụ tìm kiếm chuyên biệt về động vật không chỉ giúp tiết kiệm thời gian mà còn nâng cao hiệu quả tìm kiếm thông tin đáng tin cậy. Đây sẽ là nguồn tài nguyên quan trọng cho học sinh, giáo viên, nhà nghiên cứu và cả những người yêu thích động vật.

Mục tiêu của dự án là xây dựng một công cụ tìm kiếm tối ưu, tập trung vào các thông tin về động vật. Hệ thống sẽ cung cấp dữ liệu phong phú về các loài động vật, bao gồm đặc điểm nhận dạng, môi trường sống, chế độ ăn uống và tình trạng bảo tồn của mỗi loài động vật cần tìm kiếm.

Hình 1: Chưa có máy tìm kiếm tiếng việt cấu hình chuyên sâu cho thông tin động vật

2 THU THẬP DỮ LIỆU

2.1 Công nghệ sử dụng

Quá trình crawl dữ liệu từ web sử dụng ngôn ngữ lập trình Python là chủ yếu, trong đó các thư viện chính sử dụng là:

- **Selenium:** Giả lập, tự động hóa trình duyệt web để xử lí các scripts, thu thập dữ liệu động.
- **Asyncio và aiohttp:** Xử lý các yêu cầu HTTP bất đồng bộ.
- **BeautifulSoup:** Phân tích cú pháp HTML để trích xuất dữ liệu.

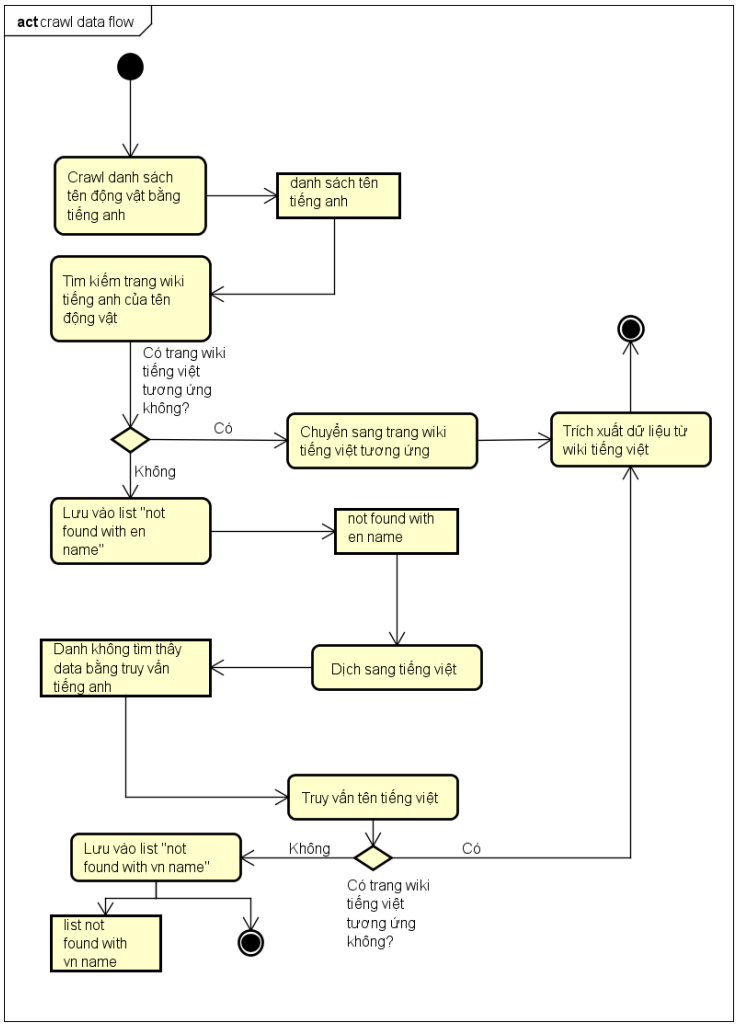
- Một số thư viện hỗ trợ xử lí thời gian chờ, xử lí json...

2.2 Phương pháp thu thập dữ liệu

Hai nguồn tài liệu chủ yếu nhất trong dự án lần này là 2 trang web: https://animalia.bio/, https://vi.wikipedia.org/ Trong đó:

- **Animalia:** Lấy danh sách tên động vật bằng tiếng anh
- **asyncio và aiohttp:** Xử lý các yêu cầu HTTP bất đồng bộ.

Luồng thu thập dữ liệu:



Hình 2: Luồng thu thập dữ liệu động vật trên web

Mã giả:

Listing 1: Mã Python thu thập dữ liệu

```python
import asyncio
import aiohttp

async def get_animal_list_from_animalia():
    animal_list = []
    return animal_list

async def find_wiki_page_in_english(session, animal_name):
    en_wiki_url = f"https://en.wikipedia.org/wiki/{animal_name}"
    async with session.get(en_wiki_url) as response:
        if response.status == 200:
            return en_wiki_url
        else:
            with open("not_found_with_en_name.txt", "a") as f:
                f.write(f"{animal_name}\n")
            return None

async def find_wiki_page_in_vietnamese(session, en_wiki_url):
    vn_wiki_url = en_wiki_url.replace("en.wikipedia.org", "vi.wikipedia.org")
    async with session.get(vn_wiki_url) as response:
        if response.status == 200:
            return vn_wiki_url
        else:
            with open("not_found_with_vn_name.txt", "a") as f:
                f.write(f"{en_wiki_url}\n")
            return None

async def extract_data_from_vn_wiki(session, vn_wiki_url):
    #code to extract and save data
    return "success"

async def main():
    animal_list = await get_animal_list_from_animalia()

    async with aiohttp.ClientSession() as session:
        tasks = []

```

```
        for animal_name in animal_list:
            en_wiki_url = await find_wiki_page_in_english(session,
                animal_name)
            if en_wiki_url is None:
                continue

            vn_wiki_url = await find_wiki_page_in_vietnamese(session,
                en_wiki_url)
            if vn_wiki_url:
                tasks.append(extract_data_from_vn_wiki(session,
                    vn_wiki_url))

        await asyncio.gather(*tasks)

    with open("not_found_with_en_name.txt", "r") as f:
        not_found_names = f.readlines()

    for name in not_found_names:
        pass

if __name__ == "__main__":
    asyncio.run(main())
```

Trong quá trình thực hiện thu thập dữ liệu, nhóm thực hiện lập trình bất đồng bộ để tận dụng tài nguyên của máy tính, tuy nhiên hạn chế của cách làm này là phụ thuộc vào tài nguyên mạng, nếu số lượng yêu cầu bất đồng bộ quá lớn so với tài nguyên mạng có thể cung cấp, lại làm chậm quá trình cào dữ liệu.

Do là đề tài thuộc phạm trù bài tập lớn, không mang tính thương mại, nên nhóm đã bỏ qua các điều khoản chặn bot của trang web, đối với các dữ liệu được đánh dấu là không cho phép truy cập hay không cho phép lưu trữ, nhóm không tiến hành thu thập.

2.3 Kết quả thu thập dữ liệu:

Kết quả thu thập dữ liệu, trên trang wikipedia tiếng việt, thu thập được hơn 22600 tài liệu thông tin các loài động vật, với các trường sau là bắt buộc:

- **title:** Tiêu đề tài liệu, thường là tên loài
- **url:** đường dẫn đến tài liệu gốc

- **description:** Mô tả ngắn gọn về nội dung tài liệu
- **content:** Nội dung chi tiết về loài, có thể bao gồm: đặc tính loài, khu vực phân bố, mức độ bảo vệ,...
- **classify:** Phân loại khoa học
- **categories:** Phân loại tài liệu tại trang web gốc


```
{
    "title": "Hàu Mỹ",
    "url": "https://vi.wikipedia.org/wiki/H%C3%A0u_M%E1%BB%B9",
    "classify": [
        "Animalia",
        "Mollusca",
        "Bivalvia",
        "Ostreoida (trang không tồn tại)",
        "Ostreidae",
        "Crassostrea",
        "Loài"
    ],
    "description": "Hàu Mỹ (Danh pháp khoa học: Crassostrea virginica) là một loài hàu trong họ Ostreidae phân bố ở Mỹ. Đây là một loại hàu có giá trị kinh tế và được nuôi nhiều ở Mỹ. ",
    "first_img": "https://upload.wikimedia.org/wikipedia/commons/e/e5/OysterBed.jpg",
    "content": "Hàu nuôi Hàng triệu con hàu đang được nuôi tại các vùng biển ở New York và các thành phố khác nhằm làm sạch môi trường nước bị ô nhiễm. Con hàu và các loài thủy sản có vỏ khác có thể làm sạch các chất độc và bụi bẩn. Việc khôi phục số lượng loài hàu tại sông Hudson gần Yonkers, bắc New York vì loài hàu giúp cải thiện môi trường sống thủy sinh, có tác dụng thu hút các loài thủy sản và sinh vật biển khác vào khu vực chúng sống. Các con hàu này chỉ nên sử dụng để làm sạch ô nhiễm, không nên ăn hay thu hoạch để bán. Một mẫu (0,4 ha mặt nước nuôi trồng) với 1 triệu con hàu cần khoản chi phí ít nhất là 50.000 USD. Mỗi con hàu có khả năng lọc khoảng 189,26 lít nước bẩn mỗi ngày.Thị trường Hiện nay Mỹ nhập khẩu hầu hết các sản phẩm hàu nuôi, khai thác, hun khói và các loại khác cùng tăng trưởng từ 16 - 123%, riêng sản phẩm hàu hun khói được Mỹ nhập khẩu nhiều nhất và chiếm đến 46,3% tổng giá trị, trong khi sản phẩm hàu khai thác chỉ chiếm chưa đến 2% thị phần. Xuất khẩu hàu của Mỹ cũng tăng trưởng nhẹ 1% về khối lượng và 6% về giá trị so với cùng kỳ năm trước, mặt hàng hàu tươi sống chiếm tỷ trọng cao nhất trên 70% tổng xuất khẩu mặt hàng này của Mỹ với khối lượng 1.820 tấn và giá trị 13,3 triệu USD, trong khi sản phẩm hàu đông lạnh và các loại khác của Mỹ lại xuất chủ yếu sang Hồng Kông.Trước đây Trung Quốc đã từng nhập khẩu khoảng 4.140 con Hàu các loại từ bang Washington Mỹ, nay Trung Quốc dừng nhập khẩu Hàu biển từ Mỹ do nhiễm khuẩn, việc nuôi dưỡng và sản xuất Hàu ở khu vực eo biển Hood bị nhiễm vi khuẩn Vibrio parahaemolyticus do ô nhiễm. Đồng thời một số lượng nhỏ loại thực phẩm này đã được phía Mỹ tiến hành xuất khẩu sang các nước như Trung Quốc, Thái lan và Indonexia. Trung Quốc đã nhập khẩu khoảng 4.140 con Hàu nhiễm bệnh. Hiện nay trên thị trường hải sản tươi sống tại thành phố Bắc Kinh không còn phát hiện thấy việc kinh doanh Hàu nhiễm bệnh có nguồn gốc từ Mỹ.Loại hàu mang tên Cape Neddick/Blue Point Oysters được khách hàng ưa thích dùng ăn sống vừa bị thu hồi tại Mỹ do nghi nhiễm vi khuẩn Vibrio parahaemolyticus. Loại khuẩn này dễ gây bệnh cho những người có hệ miễn dịch yếu, triệu chứng nhiễm khuẩn bao gồm tiêu chảy, buồn nôn, nôn, sốt và ớn lạnh. Thông thường các triệu chứng xảy ra trong vòng 24 giờ, kéo dài ba ngày.",
    "categories": [
        "Crassostrea",
        "Động vật được mô tả năm 1791",
        "Hàu",
        "Động vật Mỹ",
        "Động vật thân mềm thương mại",
        "Động vật thân mềm ăn được"
    ]
}
```


Hình 3: Một mẫu dữ liệu trong bộ dữ liệu

Tuy nhiên bộ dữ liệu này sau được tinh gọn lại thành 3 trường: *title*, *content*, *link* để tối ưu hoá khả năng mở rộng của máy tìm kiếm.


```
crawl_data > content_final_no_cate_3field > { } 0. Acanthonus armatus.json > ...
{
    "title": "Acanthonus armatus",
    "link": "https://vi.wikipedia.org/wiki/Acanthonus_armatus",
    "content": "Acanthonus armatus là một loài cá được tìm thấy ở vùng đại dương nhiệt đới và cận nhiệt đới ở độ sâu từ 1.171 đến 4.415 mét (3.842 đến 14.485 ft). Loài này phát triển đến chiều dài 37,5 cm (14,8 in) SL. Nó là thành viên duy nhất được biết trong chi của nó. Nội dung:  Phân loài: Động vật, Động vật có dây sống, Actinopterygii, Ophidiiformes, Ophidiidae (trang không tồn tại), Albert Günther, Loài,  first_img: https://upload.wikimedia.org/wikipedia/commons/9/93/Acanthonus_armatus.jpg"
}
```


Hình 4: Mẫu dữ liệu tối giản 3 trường

Sau đó, từ dữ liệu 8 phân ngành động vật, chọn ra 125 tài liệu có độ dài lớn nhất đưa vào bộ 1000 tài liệu theo yêu cầu.

3 CẤU HÌNH TÌM KIẾM

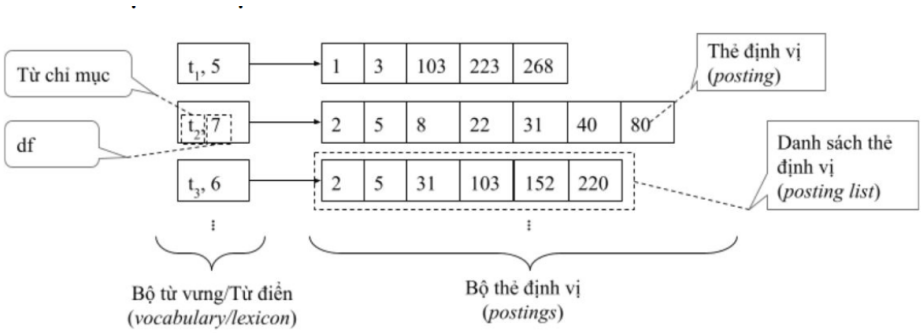
3.1 Giới thiệu Elasticsearch

Elasticsearch là một công cụ tìm kiếm và phân tích dữ liệu phân tán, được xây dựng trên nền tảng Apache Lucene. Nó cho phép lưu trữ, tìm kiếm và phân tích một lượng lớn dữ liệu trong thời gian thực, đồng thời hỗ trợ các loại dữ liệu đa dạng như văn bản, số, và dữ liệu địa lý. Elasticsearch thường được sử dụng để phát triển các ứng dụng tìm kiếm toàn văn bản (full-text search), phân tích nhật ký (log analytics), giám sát (monitoring), và phân tích dữ liệu kinh doanh. Với khả năng mở rộng mạnh mẽ, hiệu năng cao và một hệ sinh thái phong phú (như Kibana, Beats và Logstash), Elasticsearch là lựa chọn phổ biến trong việc quản lý và khai thác dữ liệu lớn trong các hệ thống hiện đại.

Các định nghĩa liên quan:

1. **Chỉ mục ngược (Inverted Index):**

   - Là một cấu trúc chỉ mục để tra cứu nhanh danh sách các văn bản chứa một từ được cho
   - Có thể chia chỉ mục ngược thành hai phần có thể lưu trữ tách biệt: Bộ từ vựng và bộ thẻ định vị. Kích thước bộ thẻ định vị có thể lớn hơn nhiều so với bộ từ vựng.



Hình 5: Cấu trúc chỉ mục ngược

2. **Xếp hạng BM25**

   BM25 (Best Matching 25) là một thuật toán xếp hạng tài liệu dựa trên mô hình truy vấn xác suất, được sử dụng rộng rãi trong các hệ thống tìm kiếm để đánh giá mức độ liên quan giữa một tài liệu và một truy vấn tìm kiếm.

   BM25 cải thiện các phương pháp truyền thống bằng cách đưa ra hai yếu tố chính:

   - **Độ dài tài liệu (Document Length):** Cân bằng ảnh hưởng của các tài liệu

ngắn và dài.

- **Tần suất từ khóa (Term Frequency):** Tăng cường trọng số cho các tài liệu chứa nhiều từ khóa, nhưng có mức độ giảm dần.

Công thức xếp hạng BM25 được viết như sau:

$$\text{BM25}(D, Q) = \sum_{t \in Q} \text{IDF}(t) \cdot \frac{f(t, D) \cdot (k_1 + 1)}{f(t, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)} \tag{1}$$

Công thức tính trọng số nghịch đảo tần suất:

$$\text{IDF}(t) = \log \frac{N - n(t) + 0.5}{n(t) + 0.5} + 1 \tag{2}$$

Từ công thức (1), ta thấy có 2 tham số có thể điều chỉnh là k_1 và b:

- $k_1 >= 0$: kiểm soát mức độ ảnh hưởng của tần suất xuất hiện từ khoá trong tài liệu, k_1 càng tăng, các tài liệu có chưa từ khoá trong truy vấn cuất hiện nhiều lần càng được ưu tiên, (thậm chí thái quá).
- $0 <= b <= 1$: Kiểm soát mức độ ảnh hưởng của độ dài tài liệu đến kết quả truy vấn. Với b càng tiến tới 1, độ dài tài liệu càng ảnh hưởng tới điểm số. Thông thường cấu hình mặc định của elasticsearch là 0.75

Giải thích ý nghĩa kí hiệu:

- Q: Tập các từ khóa trong truy vấn.
- t: Một từ khóa trong truy vấn Q.
- $f(t, D)$: Tần suất xuất hiện của từ t trong tài liệu D (Term Frequency).
- $|D|$: Độ dài tài liệu D (tổng số từ trong tài liệu).
- avgdl: Độ dài trung bình của các tài liệu trong tập dữ liệu.
- k_1: Tham số điều chỉnh, thường $k_1 = 1.2$ hoặc $k_1 = 2.0$.
- b: Tham số điều chỉnh độ dài tài liệu, thường $b = 0.75$.
- $\text{IDF}(t)$: Trọng số nghịch đảo tần suất tài liệu.
- N: Tổng số tài liệu trong tập dữ liệu.
- $n(t)$: Số tài liệu chứa từ khóa t.

3.2 Cấu hình chỉ mục tìm kiếm trên elasticsearch

Cấu hình index chi tiết:

```
{
    "settings": {
      "analysis": {
        "analyzer": {
            "vn_text_analyzer": {
                "type": "custom",
                "tokenizer": "standard",
                "filter": ["lowercase", "my_stop_filter"]
            },
            "vn_text_analyzer_no_accent": {
                "type": "custom",
                "tokenizer": "standard",
                "filter": ["lowercase", "asciifolding", "my_stop_filter
                    "]
            }
        },
        "tokenizer": {
            "my_ngram_tokenizer": {
                "type": "ngram",
                "min_gram": 3,
                "max_gram": 4
            }
        },
        "filter": {
            "my_stop_filter": {
                "type": "stop",
                "stopwords_path": "stopword.txt"
            }
        }
    },
    "similarity": {
      "bm25_custom_content": {
          "type": "BM25",
          "k1": 2.0
      }
  }
```

```
    },
    "mappings": {
      "properties": {
        "title": {
          "type": "text",
          "analyzer": "vn_text_analyzer",
          "fields": {
              "no_accent": {
                  "type": "text",
                  "analyzer": "vn_text_analyzer_no_accent"
              }
          }
      },
        "link": {
          "type": "keyword"
        },
        "content": {
          "type": "text",
          "analyzer": "vn_text_analyzer",
          "similarity": "bm25_custom_content",
          "fields": {
              "no_accent": {
                  "type": "text",
                  "analyzer": "vn_text_analyzer_no_accent"
              }
          }
      }
      }
    }
  }
```

Chú thích:

Cấu hình mapping với 3 trường ban đầu: title, content, link. Vì thế để tối đa hoá tìm kiếm trên chỉ mục, khi đánh chỉ mục cho các tài liệu sau này, cần có ít nhất 3 trường: *title*, *content*, *link*.

- Do tài liệu là tiếng việt, cần sử dụng $Multi - field$ để tạo thêm trường không dấu cho tìm kiếm không dấu

- **Analyzer:** Phần *analyzer* định nghĩa các bộ phân tích tùy chỉnh cho văn bản tiếng Việt, bao gồm một bộ xử lý văn bản có dấu và một bộ xử lý văn bản không dấu. Các bộ này sử dụng bộ tách từ (*tokenizer*) và các bộ lọc (*filter*) như chuyển chữ thường, loại bỏ dấu, và loại bỏ từ nối để hỗ trợ tìm kiếm chính xác hơn.
- **Tokenizer:** Phần *tokenizer* định nghĩa cách cắt văn bản thành các đoạn nhỏ (n-grams) với độ dài từ 3 đến 4 ký tự. Điều này giúp cải thiện khả năng tìm kiếm khi người dùng nhập các cụm từ không đầy đủ hoặc sai chính tả.
- **Filter:** Phần *filter* bao gồm bộ lọc từ dừng (*stop filter*), sử dụng danh sách từ dừng tùy chỉnh từ tệp *stopword.txt*. Bộ lọc này loại bỏ các từ không mang ý nghĩa quan trọng để tập trung vào các từ khóa trong tìm kiếm.
- **Similarity:** Phần *similarity* tùy chỉnh thuật toán BM25 với các tham số điều chỉnh khác nhau cho các trường *content*. Điều này giúp cải thiện xếp hạng kết quả tìm kiếm dựa trên ngữ cảnh và mức độ liên quan.

4 XÂY DỰNG HỆ THỐNG

4.1 Các tài nguyên sử dụng

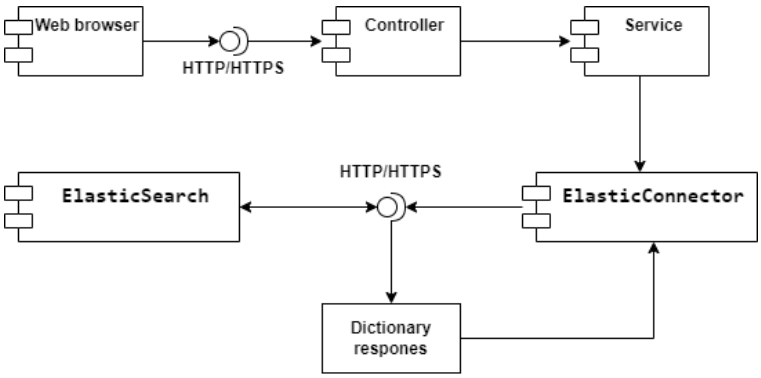
Mã nguồn chương trình tại: https://github.com/haiminhnguyenn/animal-search-engine

Để xây dựng trang web tìm kiếm hoàn chỉnh, nhóm sử dụng các tài nguyên sau:

- **Elasticsearch:** một thư viện của python, giúp backend kết nối với elasticsearch thông qua một cổng http/https
- **Flask:** Ứng dụng web xây dựng back-end, quản lý các route APIs bằng Flask.
- **Jquery:** Xử lí gọi APIs tại front-end, được sử dụng kèm theo html, css.
- Một số Thư viện hỗ trợ xử lí dữ liệu khác.

4.2 Kiến trúc chương trình

Biểu đồ thành phần thể hiện cấu trúc chương trình:



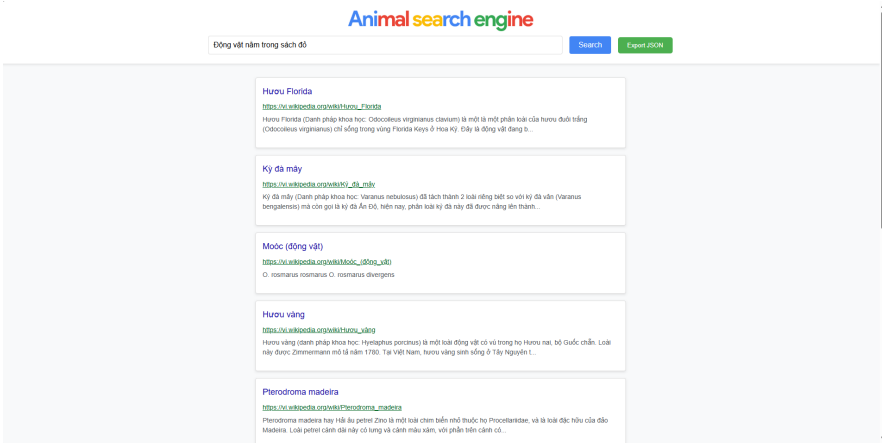
Hình 6: Biểu đồ thể hiện cấu trúc chương trình

Trong đó:

- **Web browser:** giao diện UI hiển thị kết quả tìm kiếm, các thao tác với index elasticsearch.
- **Controller:** Nhận request từ web browser thông qua restfulAPI
- **Service:** Xử lí logic, trả về dữ liệu cho controller
- **Elastic Connector:** Module hỗ trợ giao tiếp giữa serivce và Elasticsearch
- **Elasticsearch:** Công cụ quản lí index, thực hiện truy vấn, giao tiếp với Connector thông qua giao thức HTTP/HTTPS

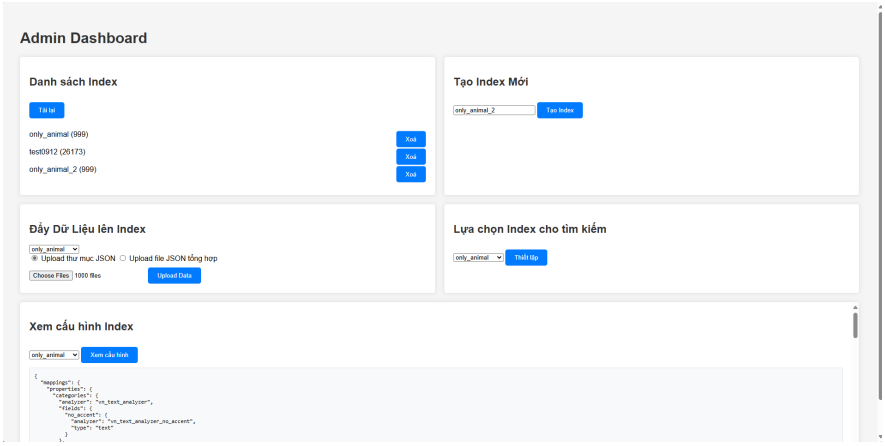
4.3 Hình ảnh minh hoạ giao diện chương trình:

1. Giao diện tìm kiếm



Hình 7: Giao diện tìm kiếm

2. Giao diện quản lí elasticsearch



Hình 8: Giao diện quản lí elasticsearch

5 KẾT QUẢ THỰC NGHIỆM

5.1 Phương pháp đánh giá:

Kết quả triển khai xây dựng máy tìm kiếm được đánh giá bằng phương pháp đánh giá độ chính xác trung bình của truy vấn (AP - Average Precision) và trung bình của AP (MAP - Mean Average Precision).

Công thức tính độ chính xác trung bình AP:

$$AP = \frac{1}{R} \sum P@K_i \tag{3}$$

Trong đó:

- R**:** số tài liệu phù hợp với truy vấn trên toàn bộ dữ liệu thử nghiệm
- $P@K_i$**:** (số lượng văn bản phù hợp trong i kết quả đầu tiên)/i

  Công thức tính AP trung bình (MAP):

$$MAP = \frac{1}{|Q|} \sum_{q}^{Q} AP_q \tag{4}$$

Trong đó:

- Q*:* Tổng số truy vấn kiểm thử
- AP_q**:** là độ chính xác trung bình của trung vấn thứ q trong tất cả Q truy vấn

5.2 Kết quả:

Thực hiện trên 50 tài liệu thử nghiệm, với 3 truy vấn và phiên bản không dấu, phiên bản sai chính tả nhẹ, lấy ra 10 kết quả đầu tiên:

- *Động vật sống dưới nước:* có 5 tài liệu phù hợp trong tập thử nghiệm.
- *Động vật sách đỏ Việt Nam:* có 8 tài liệu phù hợp trong tập thử nghiệm.
- *Động vật ăn thịt:* có 12 tài liệu phù hợp trong tập thử nghiệm.

Kết quả các văn bản phù hợp trên 10 kết quả trả về được biểu diễn tại bảng sau:

Truy vấn	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	AP
Động vật sống dưới nước	0	0	0	1	0	0	0	0	0	0	0.050
Động vật ăn thịt	1	1	1	1	1	0	0	1	1	1	0.611
Động vật sách đỏ Việt Nam	0	1	1	0	1	1	0	0	1	0	0.499
ĐỘng vậ sống dướis nướ	0	0	0	0	0	0	1	0	0	0	0.029
động vật ăng thịy	1	1	0	0	0	1	0	0	0	1	0.710
Động vậ sácg đỏ việt nam	0	1	1	0	1	1	1	1	1	1	0.697
Dong vat song duoi nuoc	0	0	0	0	0	1	0	0	0	1	0.054
Dong vat an thit	1	1	1	1	1	0	1	1	1	1	0.710
Dong vat sach do Viet Nam	1	0	1	0	0	0	1	0	0	1	0.042

Hình 9: Đánh giá độ chính xác trung bình trong 10 kết quả với 9 truy vấn mẫu

Từ đây, ta tính được $MAP = 0.306$

5.3 Nhận xét:

- Dựa vào giá trị độ chính xác trung bình MAP, ta thấy khi thử nghiệm trên tập 50 tài liệu, độ chính xác chưa cao như mong muốn, có thể do một số điểm sau:
  - Cấu hình chưa đủ tốt
  - Tập dữ liệu chưa đủ lý tưởng để thử nghiệm
- Trên tập tài liệu lớn hơn, truy vấn đạt độ chính xác cao hơn

6 KẾT LUẬN

Trong báo cáo này, nhóm PHM thực hiện cấu hình và triển khai một máy tìm kiếm dựa trên công cụ tìm kiếm elasticsearch, với chủ đề tìm kiếm chính là "thông tin về động vật". Nhóm rút ra được một số kết luận sau đây:

Về ưu điểm, nhóm đã đáp ứng được cơ bản yêu cầu của đề tài, vận dụng được các kiến thức về máy tìm kiếm, đánh giá kết quả tìm kiếm, thu thập dữ liệu từ web để thực hiện đề tài. Nhóm cũng đã xây dựng được hệ thống tìm kiếm chuyên sâu về động vật có tính ứng dụng thưc tiễn. Hệ thống tìm kiếm PHM cũng có khả năng mở rộng không chỉ với tập dữ liệu về động vật, với một số điều chỉnh về bộ dữ liệu đưa vào, hệ thống có thể thực hiện tìm kiếm trên cả các tập dữ liệu với nhiều chủ đề khác nhau mà vẫn cho ra kết quả tìm kiếm từ ổn trở lên.

Về nhược điểm, trước hết quá trình thu thập dữ liệu cho dự án còn thô sơ, chưa tận dụng được tối đa các thư viện tiện ích có sẵn. Cấu hình chỉ mục tìm kiếm còn chưa tối ưu, hiệu suất tìm kiếm chưa cao, chưa thoã mãn được hoàn toàn nhu cầu người sử dụng hệ thống.

Vì còn nhiều thiếu sót, trong tương lai, nhóm PHM tiếp tục cải thiện hệ thống tìm kiếm bằng một số cách: Thử nghiệm các cấu hình chỉ mục tốt hơn giúp tối ưu độ chính xác của kết quả. Cải thiện tìm kiếm mờ (truy vấn không dấu, sai chính tả, tìm kiếm đồng nghĩa). Ngoài ra còn có thể ứng dụng trí tuệ nhân tạo vào trong xử lí truy vấn giúp đáp ứng nhu cầu người sử dụng.

7 TÀI LIỆU THAM KHẢO

[1] Bài giảng môn "Tìm kiếm thông tin", Trường Công nghệ thông tin và Truyền thông, Đại Học Bách Khoa Hà Nội

[2] Tài liệu chính thức của Flask - https://flask.palletsprojects.com/en/stable/

[3] Tài liệu chính thức của Elasticsearch -https://www.elastic.co/docs

[4] Danh sách stopword của github Van-Duyet Le - https://github.com/stopwords/vietnamese-stopwords/blob/master/vietnamese-stopwords.txt