**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**Viện Công Nghệ Thông Tin và Truyền Thông**



**BÁO CÁO BÀI TẬP LỚN**

**TÌM KIẾM VÀ TRÌNH DIỄN THÔNG TIN**

**Giảng viên hướng dẫn:** *TS Nguyễn Bá Ngọc*

**Sinh viên tham gia: Nguyễn Hoàng Thông 20163913**

**Nguyễn Danh Nam 20162814**

Hà Nội, 2020

**MỤC LỤC**

[1. Thu thập dữ liệu 3](#_Toc27327830)

[1.1. Công cụ thu thập dữ liệu 3](#_Toc27327831)

[1.2. Nguồn và kịch bản thu thập dữ liệu 4](#_Toc27327832)

[1.2.1. Nguồn và định dạng dữ liệu thu thập 4](#_Toc27327833)

[1.2.2. Kịch bản thu thập dữ liệu 4](#_Toc27327834)

[2. Hệ thống tìm kiếm dữ liệu 5](#_Toc27327835)

[2.1. Giới thiệu Apache Solr 5](#_Toc27327836)

[2.2. Schema trong Solr 5](#_Toc27327837)

[2.2.1. Các kiểu trường dữ liệu (Field Types) 6](#_Toc27327838)

[2.2.2. Định nghĩa một trường dữ liệu 6](#_Toc27327839)

[2.3. Cú pháp truy vấn trong Solr 7](#_Toc27327840)

[2.3.1. So khớp từ khóa (Keyword matching) 7](#_Toc27327841)

[2.3.2. Wildcard matching 7](#_Toc27327842)

[2.3.3. So khớp xấp xỉ 7](#_Toc27327843)

[2.3.4. Tìm kiếm theo khoảng 7](#_Toc27327844)

[2.3.5. Boosts 8](#_Toc27327845)

[2.4. Đưa dữ liệu vào Solr 8](#_Toc27327846)

[2.4.1. Thêm plugin VietnameseAnalyzer và VietnameseTokenizer vào Solr 8](#_Toc27327847)

[2.4.2. Tạo loại trường dữ liệu mới 9](#_Toc27327848)

[2.4.3. Tạo schema: 9](#_Toc27327849)

[2.4.4. Đẩy dữ liệu vào Solr 10](#_Toc27327850)

[2.5. Tìm kiếm dữ liệu với Solr 11](#_Toc27327851)

[2.6. Đánh giá hệ thống tìm kiếm 12](#_Toc27327852)

[2.6.1. Xây dựng tập test 12](#_Toc27327853)

[2.6.2. Đánh giá 12](#_Toc27327854)

[3. Xây dựng Web Service tìm kiếm thông tin 12](#_Toc27327855)

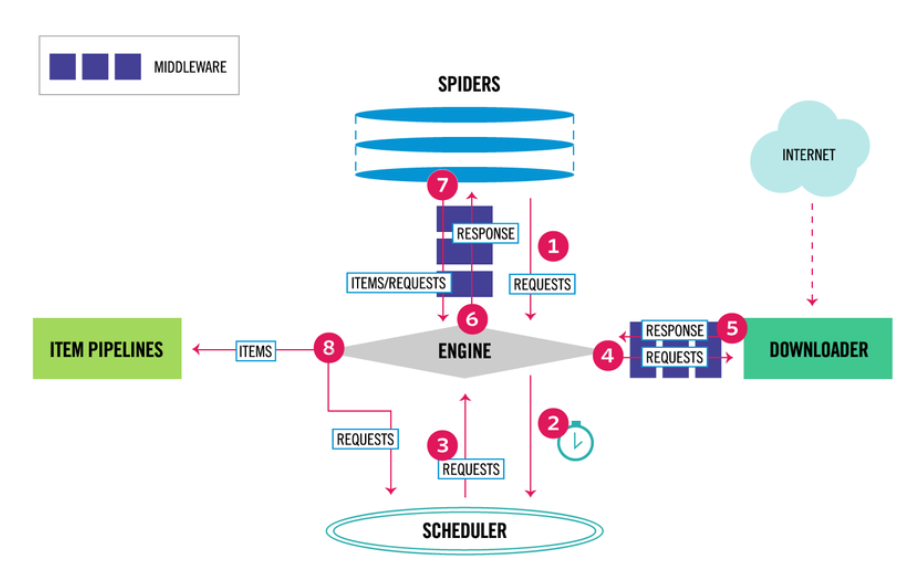
[3.1. Xây dựng API backend 13](#_Toc27327856)

[3.2. Xây dựng giao diện web 13](#_Toc27327857)

# Thu thập dữ liệu

## Công cụ thu thập dữ liệu

* Trên Internet tồn tại 1 lượng dữ liệu khổng lồ với đa dạng các định dạng như: ảnh, văn bản, âm thanh, video, ... Việc tận dụng được lượng dữ liệu khổng lồ này sẽ vô cùng có lợi trong công nghệ thông tin, đặc biệt là trong việc tạo dữ liệu mẫu, thử nghiệm các mô hình học máy, học sâu hay thử nghiệm các phương pháp trình diễn và tìm kiếm dữ liệu.
* Tồn tại rất nhiều các công cụ hay thư viện tích hợp sẵn cho việc thu thập dữ liệu như: Selenium, BeautifulSoup, Scrapy, … với mỗi loại công cụ hay thư viện đều sở hữu những thế mạnh riêng của chúng. Tùy thuộc vào kinh nghiệm cũng như thế mạnh của từng framework hay thư viện mà các lập trình viên có thể tự lựa chọn cho mình công cụ phù hợp để thu thập dữ liệu. Trong phần thu thập dữ liệu của bài tập lớn lần này, nhóm chúng em sử dụng Scrapy để thu dữ liệu văn bản trên các trang báo điện tử. Lý do nhóm em quyết định lựa chọn và sử dụng Scrapy:
* Scrapy được thiết kế tương thích cao với ngôn ngữ lập trình Python, cũng là ngôn ngữ nhóm em sử dụng trong bài tập lớn
* Các thành viên trong nhóm đã có kinh nghiệm làm việc với Scrapy trong các dự án trước đó nên có thể dễ dàng làm quen lại và tiến hành thu thập dữ liệu trong thời gian ngắn.
* Scrapy là 1 framework mạnh mẽ, với sự kết hợp của các thư viện xử lý request url phổ biến, nó có thể thu thập được đa dạng các loại, kiểu dữ liệu có trên mạng Internet hiện tại
* Scrapy có khả năng thu thập dữ liệu trên nhiều trang web có cấu trúc khác nhau dựa trên 1 số thư viện được phát triển sau này, xử lý được vấn đề mà trước đó là thế mạnh lớn của Selenium (trang web được load bằng JavaScript).
* Tài liệu về Scrapy vô cùng chi tiết, lượng người dùng và cộng đồng sử dụng lớn khiến Scrapy dễ tiếp cận.
* Cách cấu hình dễ dàng, phương pháp lưu dữ liệu dễ hiểu giúp người dùng có thể cấu trúc dữ liệu lưu về theo định dạng linh hoạt tùy ý.



Ảnh 1: Sơ đồ cấu trúc trong Scrapy

## Nguồn và kịch bản thu thập dữ liệu

### Nguồn và định dạng dữ liệu thu thập

* Dữ liệu thu thập là thông ti mô tả các bộ phim trên website phimmoi.net
* Định dạng dữ liệu trả về: JSON gồm các trường như sau:
* name\_vi: tên phim tiếng Việt
* name\_en: tên phim tiếng Anh
* status: tình trạng (Tình trạng tập phim, đã hoàn thành hay vẫn đang chiếu)
* year: năm phát hành phim
* actors: danh sách các diễn viên của bộ phim
* directors: tên đạo diễn phim
* country: quốc gia sản xuất
* kind: thể loại phim
* url: đường dẫn link đến phim
* url\_image: đường dẫn ảnh poster phim

### Kịch bản thu thập dữ liệu

* Chiến lược thu thập dữ liệu gồm 2 phần chính:
* Thu thập dữ liệu trên từng bộ phim: Trong 1 bài viết, nguồn html của trang sẽ gồm các tag mà trong các tag đó sẽ chứa đoạn văn bản mà ta cần thu thập. Nhiệm vụ của người thu thập sẽ là đi tìm các Xpath dẫn đến các đoạn văn bản cần thu thập đó, sau đó đẩy chúng vào 1 ItemPipeline được định nghĩa sẵn. ItemPipeline gồm các trường mà ta định thu thập (đã trình bày trong phần trên)
* Để truy cập được vào các trang chi tiết của từng bài viết, nhóm em thiết lập cơ chế tìm kiếm tất cả các đường dẫn trên cùng 1 trang. 1 trang báo điện tử có thể gồm rất nhiều trang trên từng thể loại (thường sắp xếp theo trình tự thời gian). Tại cuối mỗi trang thường sẽ có nút dẫn đến trang kế tiếp. Nhiệm vụ của người thu thập sẽ là viết 1 luật để tìm ra nút dẫn đến trang kế đó (tìm xpath tương ứng với nút đó) sau đó viết hàm thu thập tất cả các đường dẫn trên từng trang và thực hiện callback tới hàm thu dữ liệu trên từng bài chi tiết. Ở đây thế mạnh của scrapy được thể hiện: Luật mà người dùng viết vừa có khả năng tìm ra nút tới trang kế, vừa có thể giới hạn được số trang tối đa cần thu thập trước khi dừng lại
* Định dạng của dữ liệu thu được được định nghĩa trong pipelines, có thể là trong exporter. Định dạng trả về của nhóm là JSON. Chú ý cần cấu hình lại Pipelines và các thông số về thu thập khác trong file settings.

# Hệ thống tìm kiếm dữ liệu

## Giới thiệu Apache Solr

* Apache Solr là một máy chủ tìm kiếm mã nguồn mở được viết bằng ngôn ngữ Java. Solr cho phép người dùng dễ dàng tạo máy tìm kiếm dữ liệu.
* Apache Solr cho tốc độ tìm kiếm nhanh bởi Solr dùng chỉ mục ngược để tìm kiếm dữ liệu.
* Đối với Solr, một văn bản (Document) là một đơn vị của tìm kiếm và chỉ mục. Một chỉ mục bao gồm một hoặc nhiều văn bản và một văn bản bao gồm một hoặc nhiều trường (Field). Trong thuật ngữ cơ sở dữ liệu, một văn bản tương ứng với một hàng, còn một trường tương ứng với một cột.

## Schema trong Solr

* Trước khi thêm các văn bản vào Solr, ta cần xác định schema – cấu trúc lưu trữ dữ liệu, được lưu trữ trong file schema.xml (phiên bản cũ, được sửa đổi trực tiếp) hay managed-schema (phiên bản mới, được sửa đổi thông qua API do Solr cung cấp).
* Một schema định nghĩa:
  + Những kiểu trường dữ liệu nào được sử dụng
  + Những trường nào nên được sử dụng như là khóa duy nhất hay khóa chính
  + Nhữn trường dữ liệu nào là bắt buộc
  + Cách đánh chỉ mục và tìm kiếm đối với mỗi trường

### Các kiểu trường dữ liệu (Field Types)

* Trong Solr, mỗi trường dữ liệu có một kiểu dữ liệu, các kiểu dữ liệu có thể được mở rộng từ các kiểu dữ liệu có sẵn trong Lucene. Một số kiểu dữ liệu cơ bản có sẵn trong Solr: float, long, double, date, text.
* Solr cho phép định nghĩa các kiểu trường dữ liệu mới bằng các kết hợp các bộ lọc (filters) và các bộ tách từ (tokenizers).

Ví dụ: Kiểu dữ liệu được nhóm định nghĩa ra để lưu trữ text dưới ngôn ngữ tiếng Việt:

|  |
| --- |
| <fieldType name="text\_vn" class="solr.TextField">  <analyzer type="index" class="org.apache.lucene.analysis.vi.VietnameseAnalyzer"/>  <analyzer type="query" class="org.apache.lucene.analysis.vi.VietnameseAnalyzer"/>  </fieldType> |

### Định nghĩa một trường dữ liệu

* Một trường dữ liệu bao gồm:
  + name: tên của trường
  + type: kiểu dữ liệu của trường dữ liệu
  + indexed: trường dữ liệu có được đánh chỉ mục hay không
  + stored: giá trị gốc của trường dữ liệu có được lưu trữ hay không
  + multiValued: trường dữ liệu có nhiều giá trị hay không

Ví dụ:

|  |
| --- |
| <field name="id" type="text" indexed="true" stored="true" multiValued="true"/> |

* Khi dữ liệu được thêm vào Solr, nó sẽ đi qua một chuỗi biến đổi trước khi được thêm vào chỉ mục, đây được gọi là pha phân tích. Kết quả của pha phân tích là một chuỗi các tokens được thêm vào chỉ mục.

Các trường có giá trị indexed là true là các trường được xử lý bởi pha phân tích trước khi thêm vào chỉ mục

Các trường có giá trị stored là true được Solr lưu trữ giá trị text gốc ở trong chỉ mục.

## Cú pháp truy vấn trong Solr

### So khớp từ khóa (Keyword matching)

* Tìm kiếm từ: Tìm kiếm từ “gods” trong trường dữ liệu name\_en:

|  |
| --- |
| name\_en:gods |

* Tìm kiếm cụm từ: Tìm kiếm cụm từ “the gods” trong trường dữ liệu name\_en:

|  |
| --- |
| title:“the gods” |

* Sử dụng kết hợp với AND và OR:
  + Tìm kiếm cụm từ “the gods” trong trường name\_en và cụm tự “Trung quốc” trong trường country:

|  |
| --- |
| name\_en:“the gods” AND country:“Trung quốc” |

* + Tìm kiếm cụm từ “the gods” trong trường name\_en và cụm tự “Trung quốc” trong trường country, hoặc từ “2009” trong trường year:

|  |
| --- |
| (title:"foo bar" AND content:"quick fox") OR year:2009 |

* Tìm kiếm từ “gods” trong trường name\_en mà không có “the”:

|  |
| --- |
| title:gods -title:the |

### Wildcard matching

* Ký tự đơn nhất (?): Tìm kiếm cho bất cứ từ nào có 4 ký tự bắt đầu với “gods” trong trường name\_en

|  |
| --- |
| name\_en:gods? |

* Dãy ký tự (\*): Tìm kiếm cho bất cứ từ nào bắt đầu bằng “gods” trong trường name\_en

|  |
| --- |
| name\_en:gods\* |

### So khớp xấp xỉ

* Lucence cho phép tìm kiếm các từ với một khoảng cách cụ thể: Tìm kiếm “the gods” với khoảng cách 4:

|  |
| --- |
| “fthe gods”~4 |

* Lưu ý:
  + Với độ xấp xỉ là 0: tương đương với so khớp chính xác
  + Với độ xấp xỉ là 1: tương đương với đổi chỗ từ
  + Với độ xấp xỉ rất lớn (ví dụ 1000000): tương đương với phép AND

### Tìm kiếm theo khoảng

* Tìm kiếm theo khoảng cho phép câu truy vấn so khớp với nhiều văn bản có giá trị của các trường nằm trong một khoảng giá trị cụ thể nào đó.

Ví dụ: tìm kiếm height trong khoảng từ 150 đến 175

|  |
| --- |
| height:[150 TO 175] |

* Tìm kiếm theo khoảng có thể bao gồm hoặc bỏ qua cận trên / dưới:
  + Sử dụng “[“ và “]” để bao gồm các cận trên / dưới
  + Sử dụng “{“ và “}” để bỏ qua cận trên / dưới

Có thể sử dụng kết hợp cả 2 kiểu dấu ngoặc trên

### Boosts

* Lucence / Solr cung cấp mức độ liên quan đối với các văn bản dựa trên các thuật ngữ được tìm thấy. Để nhấn mạnh vào một thuật ngữ sử dụng dấu ^ đi cùng với một mức độ nhấn mạnh đằng sau thuật ngữ được tìm kiếm, mức độ nhấn mạnh càng cao, các thuật ngữ càng liên quan được tìm thấy

Ví dụ:

|  |
| --- |
| “đội tuyển” “bóng đá Việt Nam”^4 |

Từ “đội tuyển” có mức độ nhấn mạnh là 1, còn “bóng đá Việt Nam” có mức độ nhấn mạnh cao hơn là bóng đá Việt Name

## Đưa dữ liệu vào Solr

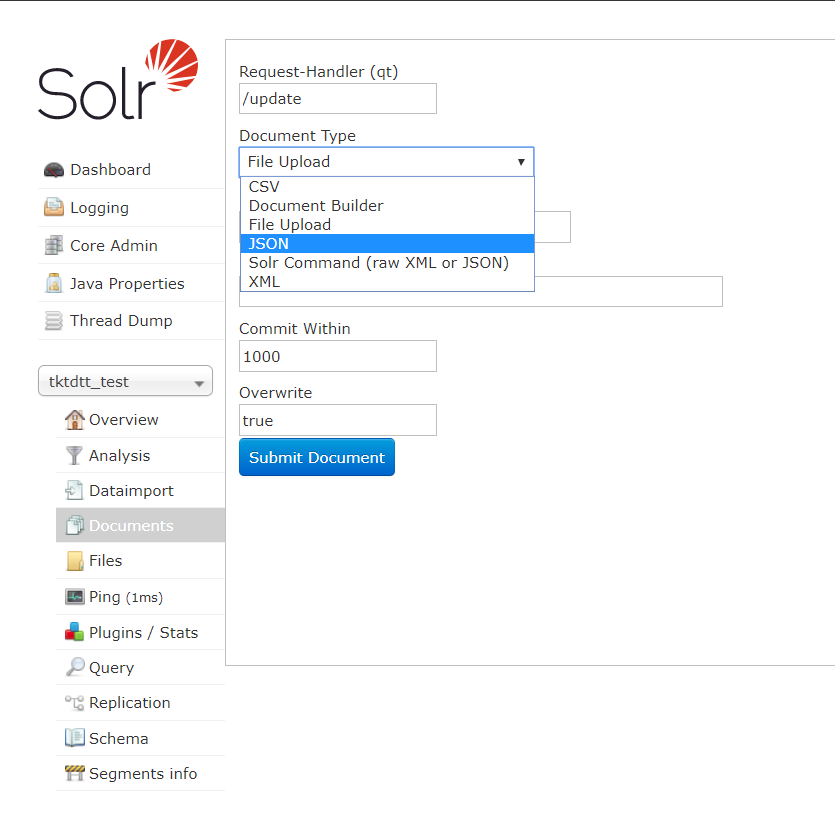
* Các văn bản được thu thập thuộc domain tin tức, mỗi văn bản bao gồm các trường:
  + name\_vi
  + name\_en
  + status
  + year
  + actors
  + director
  + country
  + kind
  + url
  + url\_image
* Để đưa dữ liệu đã thu thập được vào Solr, đầu tiên phải tạo schema cho dữ liệu đã thu thập được. Với dữ liệu text tiếng Việt, nhóm đã sử dụng tách từ tiếng Việt thay vì tách từ bằng khoảng trắng mặc định như trong tiếng Anh.

### Đẩy dữ liệu vào Solr

* Sử dụng Solr API để đẩy dữ liệu vào Solr:

|  |
| --- |
| curl 'http://localhost:8983/solr/tktdtt\_test/update?commit=true' --data-binary /path/to/json/file -H 'Content-type:application/json' |

* Sử dụng thư viện pysolr
* Sử dụng giao diện Solr Admin tại [http://localhost:8983/solr/#/](http://localhost:8983/solr/" \l "/<collection_name>/documents)<collection\_name>/documents



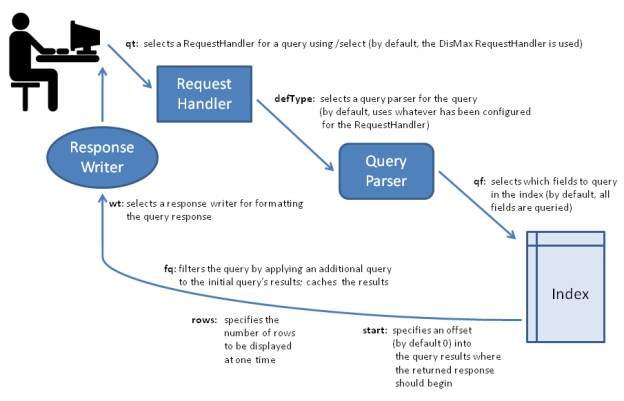
Ở tùy chọn Document Type:

* + Chọn File Upload để upload file
  + Các tùy chọn CSV, JSON, XML để đẩy dữ liệu dưới dạng text trực tiếp với format phù hợp với tùy chọn.

**2.4.2. Module phân loại tiếng anh và tiếng việt**

## Tìm kiếm dữ liệu với Solr

* Biểu đồ tóm tắt quá trình tìm kiếm với Solr:



* Solr cung cấp API để tìm kiếm dữ liệu thông qua địa chỉ <http://localhost:8983/solr/<collection_name>/select?q=query> với các tham số:
  + defType: Lựa chọn bộ xử lý câu truy vấn
  + sort: sắp xếp các kết quả tìm kiếm theo thứ tự tăng dần hay giảm dần dựa vào score hoặc những điều kiện cụ thể khác
  + start: độ lệch (mặc định là 0), vị trí bắt đầu trong kết quả tìm kiếm trả về.
  + rows: giới hạn số lượng kết quả trả về
  + fq: bộ lọc đối với kết quả trả về
  + fl: giới hạn thông tin trả về bằng danh sách các trường trả về
  + wt: định dạng dữ liệu trả về

## Đánh giá hệ thống tìm kiếm

### Xây dựng tập test

* Thực hiện 10 câu truy vấn khác nhau trên nhiều chủ đề, mỗi câu truy vấn nhóm em lấy top 30 kết quả tìm kiếm thu được từ Solr bổ sung vào tập test.
* Các câu truy vấn được sử dụng trong bài tập lớn là:
* Bóng đá Việt Nam
* Tai nạn giao thông
* Giáo dục
* Quốc phòng
* Ngoại giao
* Kinh tế thị trường
* Âm nhạc
* Giải trí
* Pháp luật nhà nước
* Dụng cụ thể thao

### Đánh giá

Kết quả thống kê:

- 10 câu truy vấn

- 205 kết quả tìm được

- 154 kế quả có liên quan đến câu truy vấn

- 51 kết quả không liên quan đến câu truy vấn

Đánh giá bằng các công thức tính độ chính xác:

|  |  |  |
| --- | --- | --- |
| **P@10** | **R@10** | **F@10** |
| 92.22 % | 49.64 % | 64.54 % |

|  |  |  |
| --- | --- | --- |
| **P@15** | **R@15** | **F@15** |
| 89.62 % | 67.19 % | 76.80 % |

|  |  |
| --- | --- |
| **MAP** | 92.44 % |

# Xây dựng Web Service tìm kiếm thông tin

* Hệ thống tìm kiếm dữ liệu với kịch bản tìm kiếm bài báo với dữ liệu báo đã thu thập được.
* Kịch bản tìm kiếm: Người dùng sẽ nhập một câu truy vấn bất kỳ vào ô tìm kiếm, hệ thống sẽ trả về các bài báo phù hợp với câu truy vấn của hệ thống



## Xây dựng API backend

* Nhóm xây dựng 1 API duy nhất đó là API tìm kiếm. API nhận input thu được từ người dùng và thực hiện tìm kiếm trong Solr như đã trình bày ở phần 2. Kết quả trả về được trình diễn qua giao diện tương tác người dùng.
* Câu truy vấn được tách từ sử dụng công cụ word\_tokenize của underthesea (do dữ liệu nhóm sử dụng là dữ liệu tiếng Việt nên cần có bộ tách từ tiếng Việt nhằm tăng hiệu quả truy vấn). API được viết bằng Flask- thư viện viết API dành cho Python cực kì mạnh mẽ và dễ sử dụng, việc kết nối với Solr dựa trên pysolr. Tất cả các thư viện đều đặc trưng dành cho Python- ngôn ngữ mà nhóm em sử dụng trong bài tập lớn lần này.
* Khi backend nhận được yêu cầu truy vấn từ người dùng, thành phần Build Query sẽ tách từ yêu cầu nhận được và xây dựng câu truy vấn theo cú pháp truy vấn của Solr. Sau đó, backend sẽ dùng thư viện pysolr để truy vấn từ Solr, sau đó kết quả thu được trả về người dùng và hiển thị lên giao diện web.

Ví dụ:

* + Input: “bóng đá Việt Nam”
  + Tokenized text: “bóng\_đá Việt\_Nam”
  + Query: (title:bóng\_đá title:Việt\_Nam) AND (description:bóng\_đá description:Việt\_Nam) AND (content:bóng\_đá content:Việt\_Nam)

## Xây dựng giao diện web

* Giao diện Web được xây dựng sử dụng Flask gồm 3 trang chính:
* Trang chủ hiển thị logo cũng như thanh tìm kiếm: bao gồm ô nhập input cũng như nút tìm kiếm
* Trang hiển thị kết quả tìm kiếm 2 gồm tiêu đề, link tới bài gốc cũng như tóm tắt bài báo
* Trang hiển thị chi tiết: Hiển thị chi tiết bài báo truy cập bao gồm các trường đã thu thập được (đã trình bày ở phần 1 trong bản báo cáo)