

**TRƯỜNG ĐẠI HỌC TRÀ VINH  
TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**



**ISO 9001:2015**

**VÕ TRỌNG NGHĨA**

**TÌM HIỂU CÔNG NGHỆ ELASTICSEARCH  
VÀ XÂY DỰNG HỆ THỐNG  
TÌM KIẾM TÀI LIỆU THÔNG MINH**

**ĐỒ ÁN TỐT NGHIỆP  
NGÀNH CÔNG NGHỆ THÔNG TIN**

**TRÀ VINH, NĂM 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH  
TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**



**ISO 9001:2015**

**TÌM HIỂU CÔNG NGHỆ ELASTICSEARCH  
VÀ XÂY DỰNG HỆ THỐNG  
TÌM KIẾM TÀI LIỆU THÔNG MINH**

**ĐỒ ÁN TỐT NGHIỆP  
NGÀNH CÔNG NGHỆ THÔNG TIN**

Giảng viên hướng dẫn : ThS **NGUYỄN KHẮC QUỐC**

Sinh viên thực hiện: **VÕ TRỌNG NGHĨA**

Mã số sinh viên: **110121273**

Lớp : **DA21TTC**

Khoá : **2021**

**TRÀ VINH, NĂM 2025**

## MỤC LỤC

CHƯƠNG 1 : TỔNG QUAN.....	1
1.1 Lý do chọn đề tài .....	1
1.2 Mục tiêu của đề tài .....	1
1.3 Đối tượng nghiên cứu .....	2
1.4 Phạm vi nghiên cứu .....	2
1.5 Phương pháp nghiên cứu .....	3
1.5.1 Phương pháp nghiên cứu tài liệu .....	3
1.5.2 Phương pháp phân tích và thiết kế hệ thống .....	3
1.5.3 Phương pháp thực nghiệm.....	4
CHƯƠNG 2 : NGHIÊN CỨU LÝ THUYẾT .....	5
2.1 Elasticsearch.....	5
2.1.1 Giới thiệu chung .....	5
2.1.2 Kiến trúc tổng quan của Elasticsearch .....	5
2.1.3 Cơ chế hoạt động .....	6
2.1.4 Các loại truy vấn trong Elasticsearch .....	6
2.1.5 Ưu điểm của Elasticsearch .....	6
2.1.6 Ứng dụng thực tế .....	6
2.2 React .....	7
2.2.1 Khái niệm .....	7
2.2.2 Cấu trúc cơ bản của Component.....	7
2.2.3 JSX .....	8
2.2.4 Virtual DOM.....	8
2.2.5 Luồng dữ liệu một chiều .....	8
2.2.6 State và Props .....	8
2.2.7 Ưu điểm của React.....	9
2.3 Express.js .....	10
2.3.1 Khái niệm .....	10
2.3.2 Đặc điểm của Express.js.....	10
2.3.3 Ưu điểm của Express.js.....	11
2.4 RESTful API .....	12
2.4.1 Khái niệm .....	12
2.4.2 Cách hoạt động của RESTful API .....	12
2.5 Node.js .....	13
2.5.1 Khái niệm .....	13
2.5.2 Đặc điểm của Node.js .....	14
2.5.3 Ưu điểm của Node.js.....	14
2.6 MongoDB.....	15
2.6.1 Khái niệm .....	15
2.6.2 Những tính năng nổi bật.....	16
2.6.3 Ưu điểm của MongoDB .....	17
2.6.4 Nhược điểm của MongoDB .....	17
2.7 Python .....	17
2.7.1 Giới thiệu.....	17
2.7.2 Ưu điểm của Python.....	18
2.7.3 Nhược điểm của Python .....	19
2.7.4 Ứng dụng Python trong xử lý tài liệu PDF và tìm kiếm ngữ nghĩa .....	19
2.7.5 Thư viện OCR.....	20

2.7.6	Thư viện sentence-transformers .....	22
2.7.7	Thư viện elasticsearch.....	23
2.7.8	Thư viện torch.....	25
2.7.9	Thư viện python-dotenv .....	27
CHƯƠNG 3 : HIỆN THỰC HÓA NGHIÊN CỨU .....		29
3.1	Mô tả.....	29
3.2	Yêu cầu chức năng .....	31
3.3	Yêu cầu phi chức năng .....	31
3.4	Kiến trúc hệ thống .....	31
3.5	Luồng xử lý tìm kiếm .....	32
3.6	Mô tả về các chức năng của hệ thống.....	33
3.7	Mô tả về actor.....	34
3.8	Lược đồ use case .....	36
3.9	Lược đồ tuần tự - Quá trình tìm kiếm tài liệu .....	40
3.10	Lược đồ hoạt động - Luồng xử lý upload tài liệu .....	41
3.11	Các bước cài đặt Elasticsearch.....	42
3.12	Cài đặt môi trường phát triển.....	44
3.13	Cài đặt các thư viện Python .....	45
CHƯƠNG 4 : KẾT QUẢ NGHIÊN CỨU.....		48
4.1	Giao diện trang chủ .....	48
4.2	Giao diện tìm kiếm .....	49
4.3	Giao diện đăng nhập.....	51
4.4	Giao diện đăng ký.....	52
4.5	Giao diện trang admin .....	54
4.6	Giao diện quản lý tài liệu.....	55
4.7	Giao diện quản lý tài khoản người dùng .....	55
4.8	Giao diện tải file.....	56
4.9	Lịch sử tìm kiếm.....	57
CHƯƠNG 5 : KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....		58
5.1	Kết luận.....	58
5.2	Hướng phát triển.....	58
DANH MỤC TÀI LIỆU THAM KHẢO.....		59
PHỤ LỤC.....		60

## LỜI MỞ ĐẦU

Trong thời đại công nghệ số phát triển mạnh mẽ, việc tìm kiếm và khai thác thông tin một cách nhanh chóng, chính xác đã trở thành nhu cầu thiết yếu trong nhiều lĩnh vực. Công nghệ Elasticsearch, với khả năng tìm kiếm và phân tích dữ liệu hiệu quả, đã mở ra nhiều cơ hội để xây dựng các hệ thống thông minh, đáp ứng yêu cầu ngày càng cao của người dùng. Nhận thấy được tầm quan trọng của công nghệ này, tôi đã lựa chọn đề tài “Tìm hiểu công nghệ Elasticsearch và xây dựng hệ thống tìm kiếm tài liệu thông minh” cho khóa luận tốt nghiệp của mình.

Đề tài tập trung vào việc nghiên cứu các khía cạnh lý thuyết của Elasticsearch, đồng thời xây dựng một hệ thống tìm kiếm tài liệu thông minh nhằm hỗ trợ người dùng truy xuất thông tin một cách hiệu quả. Thông qua quá trình thực hiện, tôi không chỉ củng cố kiến thức chuyên môn mà còn rèn luyện kỹ năng phân tích, thiết kế và triển khai hệ thống trong lĩnh vực Công nghệ Thông tin.

Khóa luận này là kết quả của sự nỗ lực học hỏi, nghiên cứu và sự hỗ trợ quý báu từ Thầy, Cô, bạn bè và gia đình. Mặc dù đã cố gắng hết sức, khóa luận không thể tránh khỏi những thiếu sót. Tôi rất mong nhận được những ý kiến đóng góp từ quý Thầy, Cô và các bạn để nội dung được hoàn thiện hơn.

## LỜI CẢM ƠN

Trước tiên, tôi xin gửi lời cảm ơn chân thành đến quý Thầy Cô trong Khoa Công nghệ Thông tin, những người đã tận tình giảng dạy, truyền đạt kiến thức và kinh nghiệm quý báu trong suốt quá trình học tập tại trường. Đặc biệt, tôi xin bày tỏ lòng biết ơn sâu sắc đến Thầy Nguyễn Khắc Quốc, người đã trực tiếp hướng dẫn, hỗ trợ và định hướng tôi trong suốt quá trình thực hiện khóa luận. Những góp ý và chỉ bảo tận tình của thầy là nguồn động lực lớn lao giúp tôi hoàn thành công trình này.

Tôi cũng xin gửi lời cảm ơn đến gia đình, những người luôn ở bên, động viên và tạo điều kiện để tôi có thể tập trung học tập và nghiên cứu.

Cuối cùng, tôi xin chúc quý Thầy, Cô, gia đình và các bạn luôn dồi dào sức khỏe, hạnh phúc và thành công trong cuộc sống.

## NHẬN XÉT

(Của cơ quan thực tập, nếu có)

This image shows a full page of primary-ruled paper. It features multiple sets of horizontal dashed lines spaced evenly down the page, providing a guide for handwriting practice. The lines are black and set against a plain white background. There are no margins, text, or other markings on the page.

(Của giảng viên hướng dẫn trong đồ án, khoá luận của sinh viên)

**Giảng viên hướng dẫn**  
(Ký và ghi rõ họ tên)



Họ và tên sinh viên: ..... MSSV: .....  
 Ngành: ..... Khóa: .....  
 Tên đề tài: .....

Họ và tên Giáo viên hướng dẫn:.....  
 Chức danh: ..... Học vị: .....

### 1. Nội dung đề tài:

[illegible]

.....

.....

.....

.....

.....

.....

.....

.....

.....

4. Điểm mới đề tài:

.....

.....

.....

.....

5. Giá trị thực trên đề tài:

.....

.....

.....

.....

.....

.....

7. Đề nghị sửa chữa bổ sung:

.....

.....

.....

.....

.....

.....

8. Đánh giá:

.....

.....

.....

.....

Trà Vinh, ngày    tháng    năm 20...

Giảng viên hướng dẫn

(Ký & ghi rõ họ tên)

**NHẬN XÉT**  
(Của giảng viên chấm trong đồ án, khoá luận của sinh viên)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

**Giảng viên chấm**  
(Ký và ghi rõ họ tên)

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**  
(*Của cán bộ chấm đồ án, khóa luận*)

Họ và tên người nhận xét: .....  
Chức danh: ..... Học vị: .....  
Chuyên ngành: .....  
Cơ quan công tác: .....  
Tên sinh viên: .....

Tên đề tài đồ án, khóa luận tốt nghiệp:

.....  
.....

**I. Ý KIẾN NHẬN XÉT**

**1. Nội dung:**

.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....  
.....

**2. Điểm mới các kết quả của đồ án, khóa luận:**

.....  
.....  
.....

**3. Ứng dụng thực tế:**

.....  
.....  
.....  
.....  
.....

(Các câu hỏi của giáo viên phản biện)

### III. KẾT LUẬN

Người nhận xét  
(Ký & ghi rõ họ tên)

## DANH MỤC CÁC BẢNG, SƠ ĐỒ, HÌNH

Hình 2.1 Logo React.....	7
Hình 2.2 Logo Express.js.....	10
Hình 2.3 Cách hoạt động của RESTful API.....	12
Hình 2.4 Logo Node.js.....	13
Hình 2.5 Logo MongoDB.....	15
Hình 3.1 Use Case tổng quan.....	36
Hình 3.2 Use case đăng ký.....	37
Hình 3.3 Use case đăng nhập.....	37
Hình 3.4 Use case tìm kiếm tài liệu.....	38
Hình 3.5 Use case xem tài liệu.....	38
Hình 3.6 Use case tải lên tài liệu.....	38
Hình 3.7 Use case quản lý tài liệu.....	39
Hình 3.8 Use case quản lý người dùng.....	39
Hình 3.9 Use case lịch sử tìm kiếm.....	39
Hình 3.10 Lược đồ tuần tự.....	40
Hình 3.11 Lược đồ hoạt động.....	41
Hình 3.12 Node.js.....	45
Hình 4.1 Trang chủ.....	48
Hình 4.2 Giao diện tìm kiếm.....	49
Hình 4.3 Giao diện trả kết quả.....	50
Hình 4.4 Giao diện đăng nhập.....	51
Hình 4.5 Giao diện đăng ký.....	52
Hình 4.6 Giao diện xác thực OTP.....	53
Hình 4.7 Giao diện trang admin.....	54
Hình 4.8 Giao diện quản lý tài liệu.....	55
Hình 4.9 Giao diện quản lý tài khoản người dùng.....	56
Hình 4.10 Giao diện tải file.....	56
Hình 4.11 Trang lịch sử tìm kiếm.....	57

## KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT

API:	Application Programming Interface
BERT:	Bidirectional Encoder Representations from Transformers
CRUD:	Create, Read, Update, Delete
DOM:	Document Object Model
ELK:	Elasticsearch, Logstash, Kibana
HTTP:	HyperText Transfer Protocol
HTML:	HyperText Markup Language
JDK:	Java Development Kit
JSON:	JavaScript Object Notation
JSX:	JavaScript XML
JWT:	JSON Web Token
NLP:	Natural Language Processing
NPM:	Node Package Manager
OCR:	Optical Character Recognition
OTP:	One-Time Password
PDF:	Portable Document Format
NLP:	Natural Language Processing
RDBMS:	Relational Database Management System
REST:	Representational State Transfer
XML:	Extensible Markup Language

## CHƯƠNG 1: TỔNG QUAN

### 1.1 Lý do chọn đề tài

Trong thời đại công nghệ số phát triển mạnh mẽ, nhu cầu tìm kiếm và khai thác thông tin một cách nhanh chóng, chính xác đã trở nên cấp thiết trong nhiều lĩnh vực. Các hệ thống tìm kiếm truyền thống dần trở nên hạn chế khi phải xử lý khối lượng dữ liệu ngày càng lớn, đặc biệt là dữ liệu phi cấu trúc.

Elasticsearch là một công cụ tìm kiếm mã nguồn mở dựa trên Apache Lucene nổi bật với khả năng tìm kiếm toàn văn (full-text search), tốc độ xử lý nhanh và dễ dàng mở rộng. Việc kết hợp Elasticsearch với các công nghệ hiện đại như React, Node.js và Python cho phép xây dựng một hệ thống tìm kiếm tài liệu thông minh, thân thiện với người dùng và hiệu quả trong việc truy xuất dữ liệu.

Nhận thấy tiềm năng của công nghệ này cũng như tính thực tiễn cao trong các hệ thống quản lý tài liệu, tôi quyết định chọn đề tài: “Tìm hiểu công nghệ Elasticsearch và xây dựng hệ thống tìm kiếm tài liệu thông minh” cho khóa luận tốt nghiệp.

### 1.2 Mục tiêu của đề tài

Đề tài “Tìm hiểu công nghệ Elasticsearch và xây dựng hệ thống tìm kiếm tài liệu thông minh” được thực hiện nhằm đạt được các mục tiêu và mục đích cụ thể như sau:

Tìm hiểu và phân tích Elasticsearch: Nghiên cứu tổng quan về kiến trúc, nguyên lý hoạt động, cơ chế lưu trữ, đánh chỉ mục và các tính năng nổi bật của Elasticsearch, bao gồm tìm kiếm toàn văn (full-text search) và tìm kiếm ngữ nghĩa (semantic search). Qua đó, đánh giá khả năng của Elasticsearch trong việc xử lý khối lượng dữ liệu lớn và phức tạp, đặc biệt là dữ liệu phi cấu trúc.

Xây dựng hệ thống tìm kiếm tài liệu thông minh: Phát triển một hệ thống tích hợp các công nghệ hiện đại như ReactJS(frontend), Node.js(backend), Python (xử lý tài liệu PDF và sinh vector ngữ nghĩa) và Elasticsearch (lưu trữ và tìm kiếm dữ



liệu). Hệ thống hỗ trợ các chức năng tìm kiếm từ khóa, tìm kiếm ngữ nghĩa, lọc nâng cao và gợi ý kết quả dựa trên nội dung, đáp ứng nhu cầu tra cứu tài liệu học thuật hoặc hành chính nội bộ.

Cải thiện trải nghiệm người dùng: Tạo ra giao diện người dùng thân thiện, dễ sử dụng, với thời gian phản hồi nhanh (dưới 2 giây) và khả năng hiển thị kết quả tìm kiếm trực quan. Hệ thống cho phép người dùng xem thông tin mô tả tài liệu, mở file PDF trực tiếp và quản lý tài liệu (đối với quản trị viên).

Ứng dụng thực tế: Đề xuất các hướng ứng dụng hệ thống trong các lĩnh vực như thư viện số, cổng thông tin nội bộ, hoặc hệ thống quản lý văn bản của tổ chức/doanh nghiệp. Đồng thời, đưa ra các giải pháp cải tiến để nâng cao hiệu suất, bảo mật và khả năng mở rộng của hệ thống.

Đóng góp lý thuyết và thực tiễn: Củng cố kiến thức về Elasticsearch và các công nghệ liên quan, đồng thời cung cấp một giải pháp thực tiễn cho bài toán tìm kiếm tài liệu, góp phần vào việc ứng dụng công nghệ thông tin trong quản lý và khai thác thông tin.

### **1.3 Đối tượng nghiên cứu**

Đối tượng của đề tài là các tài liệu dưới định dạng PDF có nội dung dạng văn bản, chủ yếu thuộc lĩnh vực học thuật hoặc hành chính nội bộ. Hệ thống hướng đến việc phục vụ:

Cán bộ trong tổ chức hoặc doanh nghiệp cần truy xuất văn bản nội bộ.

Người dùng cuối có nhu cầu tìm kiếm thông tin từ khối lượng lớn tài liệu phi cấu trúc.

### **1.4 Phạm vi nghiên cứu**

Đề tài tập trung vào các nội dung chính:

Nghiên cứu và ứng dụng Elasticsearch vào bài toán tìm kiếm tài liệu.

Xây dựng hệ thống frontend sử dụng ReactJS với các chức năng cơ bản như tìm kiếm tài liệu, quản lý tài liệu, phân quyền người dùng.

Phát triển backend bằng Node.js xử lý xác thực, routing và giao tiếp với Elasticsearch.

Sử dụng Python để trích xuất nội dung từ file PDF và sinh vector ngữ nghĩa phục vụ tìm kiếm semantic.

Phạm vi tài liệu xử lý giới hạn ở định dạng PDF và ngôn ngữ tiếng Việt.

## **1.5 Phương pháp nghiên cứu**

### **1.5.1 Phương pháp nghiên cứu tài liệu**

Thu thập và phân tích các tài liệu, bài báo khoa học, sách chuyên ngành và tài liệu kỹ thuật từ các nguồn uy tín như trang web chính thức của Elastic, Python Software Foundation, MongoDB, và các công bố khoa học trên Hugging Face. Các tài liệu này được sử dụng để tìm hiểu về Elasticsearch (kiến trúc, cơ chế hoạt động, ưu điểm), các công nghệ liên quan (ReactJS, Node.js, MongoDB, Python) và các thư viện hỗ trợ (sentence-transformers, pytesseract, PyTorch).

Tổng hợp và đánh giá các nghiên cứu trước đó về ứng dụng Elasticsearch trong tìm kiếm tài liệu, xử lý ngôn ngữ tự nhiên (NLP), và trích xuất nội dung từ tài liệu PDF, từ đó xác định các điểm mạnh và hạn chế để áp dụng vào hệ thống.

### **1.5.2 Phương pháp phân tích và thiết kế hệ thống**

Phân tích yêu cầu chức năng và phi chức năng của hệ thống tìm kiếm tài liệu thông minh, bao gồm các chức năng như đăng ký, đăng nhập, tìm kiếm từ khóa, ngữ nghĩa, quản lý tài liệu và lịch sử tìm kiếm.

Hệ thống được xây dựng theo kiến trúc Client-Server, ban đầu tổ chức theo mô hình phân tầng (Layered Architecture) để đảm bảo tính rõ ràng và dễ bảo trì. Các thành phần chính bao gồm: giao diện người dùng (ReactJS), backend xử lý nghiệp vụ (Node.js), module xử lý dữ liệu (Python) và hệ quản trị dữ liệu tìm kiếm (Elasticsearch). Các sơ đồ use case, tuần tự và hoạt động được xây dựng để mô tả rõ luồng xử lý và sự tương tác giữa các thành phần trong hệ thống.

### 1.5.3 Phương pháp thực nghiệm

Triển khai hệ thống trên môi trường phát triển với các công cụ như Visual Studio Code, Anaconda, và các thư viện Python (sentence-transformers, elasticsearch, torch, python-dotenv). Các bước cài đặt môi trường, bao gồm Java Development Kit, Node.js, Elasticsearch, và các thư viện Python, được thực hiện và kiểm tra kỹ lưỡng.

Xây dựng giao diện bằng các công cụ như ReactJS. Áp dụng nguyên tắc responsive design, bố cục rõ ràng, màu sắc dễ chịu.

Thực hiện các thí nghiệm để kiểm tra hiệu quả của hệ thống, bao gồm:

Trích xuất nội dung từ tài liệu PDF bằng pytesseract.

Sinh vector ngữ nghĩa bằng mô hình SentenceTransformer (paraphrase-multilingual-MiniLM-L12-v2) và lưu trữ trong Elasticsearch.

Thực hiện tìm kiếm theo từ khóa và tìm kiếm ngữ nghĩa dựa trên các câu truy vấn.

## CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

### 2.1 Elasticsearch

#### 2.1.1 Giới thiệu chung

Elasticsearch là một công cụ tìm kiếm và phân tích dữ liệu thời gian thực, mã nguồn mở, được xây dựng trên nền tảng Apache Lucene – một thư viện tìm kiếm full-text nổi tiếng. Elasticsearch được phát triển lần đầu vào năm 2010 bởi Shay Banon và hiện được duy trì bởi Elastic NV.

Elasticsearch nổi bật nhờ khả năng tìm kiếm toàn văn bản (full-text search) cực kỳ nhanh chóng, hỗ trợ truy vấn phân tán (distributed search), lưu trữ có khả năng mở rộng cao, và tích hợp tốt với các hệ thống lớn. Đây là một thành phần trung tâm trong bộ sản phẩm Elastic Stack (ELK Stack) bao gồm Elasticsearch, Logstash, Kibana, và sau này bổ sung thêm Beats[7].

#### 2.1.2 Kiến trúc tổng quan của Elasticsearch

Elasticsearch được thiết kế theo kiến trúc phân tán, trong đó bao gồm các khái niệm chính sau:

Cluster: Tập hợp các node cùng hoạt động và chia sẻ dữ liệu.

Node: Một máy chủ (vật lý hoặc ảo) trong cluster, thực hiện các tác vụ như lưu trữ, xử lý truy vấn,...

Index: Tương đương với một cơ sở dữ liệu trong RDBMS. Mỗi index chứa nhiều tài liệu.

Document: Đơn vị dữ liệu cơ bản, tương tự như một dòng dữ liệu (record) trong bảng.

Shard và Replica:

Shard: Elasticsearch tự động chia nhỏ dữ liệu thành các shard để lưu trữ phân tán.

Replica: Bản sao của shard nhằm tăng tính sẵn sàng và hiệu suất đọc[7].

### **2.1.3 Cơ chế hoạt động**

Elasticsearch sử dụng cơ chế ngược chỉ mục (inverted index) để thực hiện tìm kiếm toàn văn bản nhanh chóng. Quá trình hoạt động bao gồm:

Indexing: Dữ liệu được đánh chỉ mục bằng cách phân tích văn bản (text analysis), chia nhỏ (tokenize), loại bỏ từ dừng (stop words), chuẩn hóa (normalize), sau đó lưu lại trong inverted index.

Searching: Khi có truy vấn, Elasticsearch tìm kiếm trong inverted index và trả về kết quả phù hợp kèm theo điểm số độ phù hợp (relevance score)[7].

### **2.1.4 Các loại truy vấn trong Elasticsearch**

- Match query: Truy vấn toàn văn đơn giản.
- Term query: Truy vấn chính xác theo keyword.
- Range query: Tìm dữ liệu trong khoảng (ngày, số...).
- Bool query: Kết hợp nhiều điều kiện truy vấn với các toán tử must, should, must\_not.
- Fuzzy query: Tìm kiếm gần đúng, hỗ trợ sai chính tả.
- Wildcard query: Hỗ trợ ký tự đại diện (\*, ?)[7].

### **2.1.5 Ưu điểm của Elasticsearch**

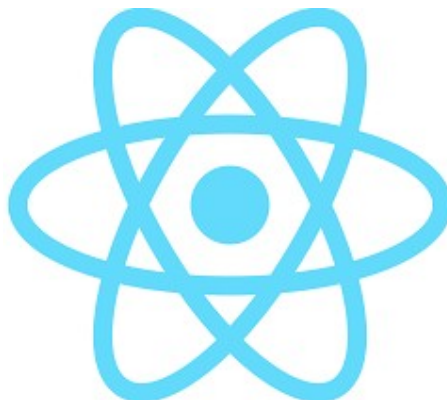
- Hiệu suất cao: Trả kết quả gần như tức thì.
- Khả năng mở rộng: Dễ dàng mở rộng theo chiều ngang.
- Realtime analytics: Phân tích dữ liệu theo thời gian thực.
- RESTful API: Giao tiếp đơn giản thông qua HTTP và JSON.
- Tích hợp tốt: Hỗ trợ các hệ sinh thái như Logstash, Kibana, Beats,...

### **2.1.6 Ứng dụng thực tế**

Elasticsearch có thể được ứng dụng vào hệ thống tìm kiếm, phân tích dữ liệu thời gian thực, chatbot và xử lý ngôn ngữ tự nhiên.

## 2.2 React

### 2.2.1 Khái niệm



*Hình 2.1 Logo React*

React là một thư viện JavaScript mã nguồn mở, được Facebook phát triển và ra mắt năm 2013, dùng để xây dựng các tương tác với thành phần trên website. Một trong những điểm nổi bật nhất của React đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client nữa.

React là một thư viện JavaScript chuyên giúp các nhà phát triển xây dựng giao diện người dùng. Trong lập trình ứng dụng front-end, lập trình viên thường sẽ phải làm việc chính trên 2 thành phần sau: Giao diện người dùng và xử lý tương tác của người dùng. Giao diện người dùng là tập hợp những thành phần có thể nhìn thấy được trên bất kỳ một ứng dụng nào, ví dụ như: menu, thanh tìm kiếm, những nút nhấn,...

### 2.2.2 Cấu trúc cơ bản của Component

React sử dụng kiến trúc dựa trên component, trong đó giao diện người dùng được chia thành các thành phần nhỏ, độc lập và có thể tái sử dụng. Mỗi component trong React là một lớp hoặc một hàm JavaScript, chứa logic và giao diện riêng biệt. Một component có thể nhỏ như một nút, hoặc lớn như toàn bộ trang. Tên component của React phải luôn bắt đầu bằng chữ in hoa. Các component có thể lồng vào nhau để tạo ra cấu trúc giao diện người dùng phức tạp [5].

### **2.2.3 JSX**

JSX là một cú pháp mở rộng của ngôn ngữ lập trình JavaScript, nó cho phép các lập trình viên viết mã HTML trực tiếp trong mã JavaScript. JSX giúp cho lập trình viên dễ dàng tạo ra các component hơn vì cú pháp của JSX gần giống với HTML.

Lợi ích khi sử dụng JSX để lập trình web là dễ sử dụng hơn JavaScript và cho phép trích dẫn HTML cũng như việc sử dụng các cú pháp thẻ HTML để render các subcomponent. JSX tối ưu hóa mã khi biên soạn, vì vậy nó chạy nhanh hơn so với mã JavaScript tương đương.

### **2.2.4 Virtual DOM**

DOM (Document Object Model) là cấu trúc mô phỏng của một trang web, dùng để truy cập và quản lý các thành phần HTML. Virtual DOM là bản sao của DOM trên trang web và React sử dụng bản copy đó để tìm kiếm đúng phần mà DOM thật cần cập nhật khi trạng thái ứng dụng thay đổi, React sẽ so sánh Virtual DOM với DOM thật và chỉ cập nhật đúng các thành phần cần thiết, giúp ứng dụng chạy nhanh và mượt mà hơn.

### **2.2.5 Luồng dữ liệu một chiều**

React không có những module chuyên dụng để xử lý data, vì vậy React chia nhỏ view thành các component nhỏ có mối quan hệ chặt chẽ với nhau. React hỗ trợ ràng buộc dữ liệu một chiều, giúp dữ liệu chỉ đi theo một hướng từ cha sang con, giúp dễ dàng kiểm soát luồng dữ liệu trong ứng dụng. Điều này làm cho việc quản lý và gỡ lỗi ứng dụng trở nên dễ dàng và rõ ràng hơn.

### **2.2.6 State và Props**

State là một đối tượng quản lý dữ liệu động trong component. Khi state thay đổi, React sẽ tự động cập nhật giao diện để phản ánh các thay đổi này. State thường được sử dụng trong các component stateful, nghĩa là các component có trạng thái thay đổi theo thời gian[5].

Props (viết tắt của properties) là các giá trị được truyền từ component cha xuống component con. Props giúp truyền dữ liệu và các hàm giữa các component và không thể thay đổi trong component con[5].

### **2.2.7 Ưu điểm của React**

React có cú pháp đơn giản và dễ hiểu, đặc biệt là đối với những người đã quen thuộc với JavaScript. JSX, một phần mở rộng của JavaScript, cho phép lập trình viên viết mã HTML trong JavaScript, làm cho quá trình phát triển trở nên trực quan và dễ dàng hơn. Điều này giúp các lập trình viên mới dễ dàng tiếp cận và bắt đầu sử dụng React nhanh chóng.

React cho phép xây dựng các thành phần giao diện người dùng nhỏ gọn và độc lập, gọi là component. Các component này có thể được tái sử dụng nhiều lần trong cùng một ứng dụng hoặc trong các dự án khác nhau, giúp tiết kiệm thời gian và công sức phát triển, đồng thời đảm bảo tính nhất quán của giao diện người dùng.

Cấu trúc cơ bản của component trong React giúp việc viết và quản lý các thành phần giao diện người dùng trở nên dễ dàng và hiệu quả hơn. Lập trình viên có thể tách biệt các phần của giao diện người dùng thành các component nhỏ hơn, dễ dàng kiểm thử và bảo trì. Việc viết các component này trở nên đơn giản nhờ JSX, cho phép kết hợp JavaScript và HTML một cách tự nhiên và dễ hiểu.

React sử dụng Virtual DOM để tối ưu hóa hiệu suất ứng dụng. Thay vì cập nhật toàn bộ DOM thật mỗi khi có thay đổi, React sẽ cập nhật Virtual DOM trước, sau đó so sánh với DOM thật để chỉ thay đổi những phần cần thiết. Điều này giúp giảm thiểu số lượng thao tác trên DOM thật, từ đó cải thiện tốc độ và hiệu suất của ứng dụng.

Cấu trúc dựa trên component của React giúp mã nguồn dễ dàng quản lý, bảo trì và mở rộng. Mỗi component có thể được phát triển, kiểm thử và bảo trì độc lập, giúp việc sửa lỗi và thêm tính năng mới trở nên đơn giản hơn. Điều này cũng giúp các nhóm phát triển có thể làm việc song song trên các phần khác nhau của ứng dụng mà không gây ra xung đột.



Với React Native, một framework dựa trên React, lập trình viên có thể phát triển các ứng dụng di động cho cả iOS và Android sử dụng cùng một codebase. Điều này giúp tiết kiệm thời gian và nguồn lực, đồng thời đảm bảo tính nhất quán của ứng dụng trên các nền tảng khác nhau.

## **2.3 Express.js**

### **2.3.1 Khái niệm**



*Hình 2.2 Logo Express.js*

Express.js là một framework web phổ biến được sử dụng để phát triển các ứng dụng web và API qua Node.js. Nền tảng được xem là một phương thức xử lý các yêu cầu HTTP, quản lý các tuyến đường, xử lý phần mềm trung gian và nhiều tính năng khác để phát triển hiệu quả ứng dụng web[6].

Express.js tập trung vào công việc tối ưu hóa việc xây dựng web ứng dụng bằng cách cung cấp một cấu trúc hoạt động và chỉ định rõ ràng việc xử lý yêu cầu và phản hồi. Nền tảng cũng hỗ trợ tích hợp các phần mềm trung gian bên ngoài để mở rộng chức năng của ứng dụng[6].

### **2.3.2 Đặc điểm của Express.js**

Định tuyến (Routing): Express.js cho phép định tuyến các yêu cầu HTTP tới các endpoint cụ thể dựa trên đường dẫn và phương thức yêu cầu. Điều này giúp tách biệt logic xử lý cho từng phần của ứng dụng và quản lý các route một cách linh hoạt.

Trích xuất dữ liệu từ yêu cầu (Request parsing): Express.js cung cấp các công cụ để trích xuất dữ liệu từ yêu cầu HTTP như tham số đường dẫn, tham số truy vấn, thân yêu cầu, header và các loại dữ liệu khác.

Phản hồi yêu cầu (Response handling): Express.js cho phép phản hồi yêu cầu với headers và status code tương ứng, cũng như trả về dữ liệu dưới nhiều định dạng như JSON, HTML hoặc phản hồi khác.

Middleware: Trung gian (middleware) trong Express.js có thể được sử dụng để xử lý các yêu cầu trước khi chúng đến các endpoint cụ thể, giúp thực hiện các thao tác như xác thực người dùng, xử lý lỗi, nén dữ liệu và nhiều tác vụ khác.

RESTful API Development: Express.js thường được sử dụng để xây dựng các RESTful API với sự hỗ trợ tốt cho việc tạo, lấy, cập nhật và xóa dữ liệu thông qua các phương thức HTTP.

### **2.3.3 Ưu điểm của Express.js**

Dễ sử dụng: Với cú pháp đơn giản và dễ hiểu, Express.js giúp việc phát triển ứng dụng trở nên nhanh chóng và hiệu quả, giảm thiểu độ phức tạp trong việc quản lý yêu cầu và phản hồi.

Hiệu suất cao: Vì được xây dựng trên Node.js, Express.js tận dụng các đặc tính của Node.js như xử lý bất đồng bộ và hiệu suất cao, giúp ứng dụng web hoặc API chạy nhanh.

Phát triển nhanh chóng: Express.js cung cấp nhiều công cụ và tính năng hỗ trợ việc phát triển web nhanh chóng, như hỗ trợ các template engine, middleware và khả năng xử lý lỗi dễ dàng.

Cộng đồng lớn: Express.js có một cộng đồng phát triển rộng lớn và rất nhiều tài liệu hỗ trợ, giúp người dùng dễ dàng tìm kiếm giải pháp và nhận sự trợ giúp khi gặp vấn đề.

Mở rộng linh hoạt: Express.js cho phép dễ dàng mở rộng ứng dụng bằng cách sử dụng các module hoặc middleware từ bên thứ ba, tăng tính linh hoạt của ứng dụng.

## 2.4 RESTful API

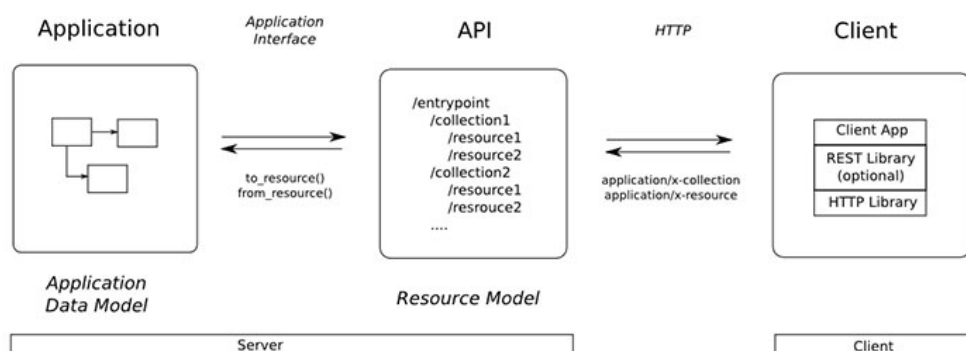
### 2.4.1 Khái niệm

RESTful API là một loại API tuân theo các nguyên tắc của kiến trúc REST (Representational State Transfer). Đây là một tiêu chuẩn thiết kế API cho các ứng dụng web, giúp quản lý các tài nguyên một cách hiệu quả. RESTful API tập trung vào tài nguyên hệ thống. Các trạng thái của tài nguyên được định dạng và truyền tải qua giao thức HTTP. [4].

API (Application Programming Interface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu cho ứng dụng dưới những kiểu dữ liệu phổ biến như JSON hay XML.

REST (REpresentational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE,... đến một URL để xử lý dữ liệu.

### 2.4.2 Cách hoạt động của RESTful API



Hình 2.3 Cách hoạt động của RESTful API

REST hoạt động dựa trên nguyên lý của kiến trúc REST chủ yếu dựa vào giao thức HTTP để giao tiếp với tài nguyên của hệ thống thông qua phương thức HTTP. Các phương thức HTTP thường được sử dụng:

- GET (SELECT): Trả về một sản phẩm hoặc một danh sách sản phẩm.
- POST (CREATE): Tạo mới một sản phẩm.
- PUT (UPDATE): Cập nhật thông tin cho sản phẩm.
- DELETE (DELETE): Xóa một sản phẩm.

Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

## 2.5 Node.js

### 2.5.1 Khái niệm



*Hình 2.4 Logo Node.js*

Được phát hành vào năm 2009, Node.js, hay còn được biết với tên gọi chính thức là Node.js, là một nền tảng JavaScript phía máy chủ (server-side) cho phép chạy mã JavaScript ngoài trình duyệt, chủ yếu dùng để phát triển ứng dụng web, ứng dụng mạng, và các dịch vụ web. Node.js được xây dựng trên V8 JavaScript engine của Google Chrome, giúp mã JavaScript chạy nhanh và hiệu quả. Node.js cho phép các lập trình viên tạo cả ứng dụng front-end và back-end bằng JavaScript[3].

### **2.5.2 Đặc điểm của Node.js**

Nguồn mở (Open-source): Mã nguồn của Node.js được công bố công khai, điều này có nghĩa là bất kỳ ai cũng có thể truy cập, sử dụng, và đóng góp vào mã nguồn. Node.js được duy trì bởi cộng đồng lập trình viên trên toàn thế giới, và hướng dẫn đóng góp của Node.js hướng dẫn bạn cách để bạn có thể góp phần phát triển nó[3].

Đa nền tảng (Cross-platform): Node.js không phụ thuộc vào bất kỳ hệ điều hành nào cụ thể nào, nghĩa là nó có thể chạy trên Linux, macOS hoặc Windows. Điều này làm cho Node.js trở thành một lựa chọn linh hoạt cho các nhà phát triển muốn xây dựng các ứng dụng có thể hoạt động trên nhiều nền tảng khác nhau mà không cần thay đổi mã nguồn[3].

Môi trường thực thi JavaScript (JavaScript runtime environment): Để mã JavaScript có thể được thực thi, nó cần một môi trường chạy phù hợp. Trong khi trình duyệt như Chrome và Firefox cung cấp một môi trường thực thi cho JavaScript, Node.js mở rộng khả năng này ra ngoài trình duyệt. Node.js cho phép chạy JavaScript trên máy chủ, hoặc trong bất kỳ môi trường máy tính nào khác, không chỉ trong trình duyệt.

Dựa trên V8 JavaScript Engine: Node.js được xây dựng dựa trên V8, động cơ JavaScript được phát triển bởi Google cho trình duyệt Chrome. Điều này giúp Node.js có khả năng thực thi JavaScript nhanh và hiệu quả, đồng thời hỗ trợ các tính năng mới nhất của ngôn ngữ JavaScript.

### **2.5.3 Ưu điểm của Node.js**

Hiệu suất cao: Node.js được xây dựng trên động cơ JavaScript V8 của Google Chrome, cho phép biên dịch mã JavaScript thành mã máy nhanh chóng. Nhờ đó, thời gian thực thi của Node.js rất nhanh, làm tăng hiệu suất của các ứng dụng.’

Hệ sinh thái phong phú: Với hơn 50,000 gói có sẵn trong Node Package Manager (NPM), các nhà phát triển có thể dễ dàng tìm và sử dụng các thư viện

theo nhu cầu của họ mà không cần phải viết lại từ đầu, tiết kiệm đáng kể thời gian và công sức.

Xử lý bất đồng bộ và không chặn (Asynchronous and Non-blocking): Node.js hoạt động một cách bất đồng bộ và không chặn các hoạt động I/O, nghĩa là nó không cần chờ đợi API trả về dữ liệu trước khi tiếp tục xử lý yêu cầu tiếp theo. Điều này làm cho Node.js trở nên lý tưởng cho việc xây dựng các ứng dụng web thời gian thực và xử lý dữ liệu lớn.

Tính nhất quán trong mã nguồn: Node.js cho phép sử dụng cùng một ngôn ngữ lập trình (JavaScript) cho cả phía máy chủ và máy khách. Điều này không chỉ giúp giảm thiểu sự không đồng bộ giữa client và server mà còn làm cho việc bảo trì và quản lý mã nguồn trở nên dễ dàng hơn.

## **2.6 MongoDB**

### **2.6.1 Khái niệm**



*Hình 2.5 Logo MongoDB*

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL mã nguồn mở, rất phổ biến hiện nay. Thay vì lưu trữ dữ liệu trong các bảng và dòng như các hệ quản trị SQL truyền thống, MongoDB lưu dữ liệu dưới dạng các tài liệu (documents) trong các bộ sưu tập (collections). Các dữ liệu được lưu trữ trong tài liệu kiểu JSON nên tốc độ truy vấn sẽ rất nhanh.

MongoDB lần đầu ra đời bởi MongoDB Inc., tại thời điểm đó là thế hệ 10, vào tháng Mười năm 2007, nó là một phần của sản phẩm PaaS (Platform as a

Service) tương tự như Windows Azure và Google App Engine. Sau đó nó đã được chuyển thành nguồn mở từ năm 2009 [2].

MongoDB là một nền tảng cơ sở dữ liệu tài liệu NoSQL linh hoạt, có thể mở rộng được thiết kế để khắc phục cách tiếp cận cơ sở dữ liệu quan hệ và những hạn chế của các giải pháp NoSQL khác. Các sản phẩm MongoDB nổi tiếng với khả năng mở rộng theo chiều ngang và cân bằng tải, điều này đã mang lại cho các nhà phát triển ứng dụng một mức độ linh hoạt và khả năng mở rộng chưa từng có [10].

Tính đến năm 2020, MongoDB đã được tải xuống hơn 30 triệu lần với hơn 730.000 đăng ký MongoDB University. Có trình điều khiển cho hơn 10 ngôn ngữ, với hàng chục ngôn ngữ khác được cộng đồng thêm vào. Hơn hết, MongoDB hoàn toàn miễn phí để sử dụng. Ứng dụng này cung cấp cho các nhà phát triển một số khả năng hữu ích, cho dù bạn cần chạy riêng tư trên trang web hay trên đám mây công cộng [10].

MongoDB có thể chạy trong các môi trường sau:

MongoDB Atlas: Đây đủ dịch vụ được quản lý để triển khai MongoDB trên đám mây.

MongoDB Enterprise: Các Phiên bản MongoDB tự quản lý, dựa trên đăng ký.

MongoDB Community Edition: Các phiên bản MongoDB có sẵn, sử dụng miễn phí và tự quản lý.

## **2.6.2 Những tính năng nổi bật**

Truy vấn ad-hoc cho phân tích thời gian thực: MongoDB hỗ trợ các truy vấn ad-hoc linh hoạt giúp tăng hiệu quả khi xử lý khối lượng dữ liệu lớn với các biến động không dự đoán trước, giúp tối ưu hóa hiệu suất.

Lập chỉ mục (Indexing) để cải thiện tốc độ truy vấn: Lập chỉ mục phù hợp tăng tốc độ truy vấn và cải thiện hiệu suất. MongoDB cho phép tạo chỉ mục theo yêu cầu trên các trường khác nhau, bao gồm cả các trường lồng trong mảng.

Sao chép để tăng tính ổn định và khả dụng dữ liệu: Sao chép dữ liệu trên nhiều máy chủ giúp tăng tính ổn định và khả năng khôi phục khi gặp sự cố.

Phân mảnh: Phân mảnh dữ liệu trên các máy chủ khác nhau giúp tăng khả năng mở rộng, giảm tải và hỗ trợ ứng dụng có số lượng người dùng lớn.

Cân bằng tải: Với tính năng sao chép và phân mảnh, MongoDB có thể xử lý hàng triệu yêu cầu đồng thời mà vẫn đảm bảo độ chính xác và tính nhất quán của dữ liệu.

### **2.6.3 Ưu điểm của MongoDB**

Dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ.

Dữ liệu được ghi đệm lên RAM, hạn chế truy cập vào ổ cứng nên tốc độ đọc và ghi cao.

### **2.6.4 Nhược điểm của MongoDB**

Không ứng dụng được cho các mô hình giao dịch nào có yêu cầu độ chính xác cao do không có ràng buộc.

Không có cơ chế giao dịch để phục vụ các ứng dụng ngân hàng.

Dữ liệu lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn.

Mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

## **2.7 Python**

### **2.7.1 Giới thiệu**

Python được tạo ra bởi Guido van Rossum vào cuối những năm 1980 tại Hà Lan, lần đầu tiên phát hành vào năm 1991 (phiên bản 0.9.0). Tên “Python” lấy cảm hứng từ chương trình hài “Monty Python's Flying Circus” của Anh, không liên quan đến loài rắn. Hiện tại, Python được duy trì bởi Python Software Foundation



(PSF) và đã phát triển qua các phiên bản lớn như Python 2 (đã ngừng hỗ trợ vào 2020) và Python 3 (phiên bản hiện hành, mới nhất là 3.12 vào năm 2025)[8].

Đặc điểm chính:

Cú pháp đơn giản, dễ đọc: Python được thiết kế để dễ học và sử dụng, với cú pháp gần gũi với ngôn ngữ tự nhiên.

Ngôn ngữ lập trình cấp cao: Python hỗ trợ lập trình hướng đối tượng (OOP), lập trình hàm (functional programming) và lập trình thủ tục.

Đa nền tảng: Chạy trên nhiều hệ điều hành như Windows, macOS, Linux.

Mã nguồn mở: Miễn phí, cộng đồng phát triển rộng lớn, cho phép tùy chỉnh và đóng góp.

Loại ngôn ngữ:

Python là ngôn ngữ thông dịch (interpreted), không cần biên dịch trước như C/C++.

Sử dụng trình thông dịch CPython (triển khai chính) hoặc các triển khai khác như PyPy, Jython.

### **2.7.2 Ưu điểm của Python**

Dễ học và sử dụng: Cú pháp rõ ràng, ít phức tạp, phù hợp cho người mới bắt đầu.

Thư viện phong phú: Python có hệ sinh thái thư viện đa dạng, như NumPy, Pandas (xử lý dữ liệu), TensorFlow, PyTorch (AI), Flask, Django (web).

Cộng đồng lớn: Hỗ trợ mạnh mẽ từ cộng đồng lập trình viên, tài liệu phong phú, diễn đàn như Stack Overflow.

Ứng dụng đa dạng: Được sử dụng trong nhiều lĩnh vực như trí tuệ nhân tạo (AI), học máy (machine learning), phát triển web, tự động hóa, phân tích dữ liệu, khoa học, IoT,...

Khả năng tích hợp: Dễ dàng tích hợp với các ngôn ngữ khác như C/C++, Java, hoặc các công nghệ như API, cơ sở dữ liệu.

### **2.7.3 Nhược điểm của Python**

Hiệu suất chậm hơn: Vì là ngôn ngữ thông dịch, Python chậm hơn so với các ngôn ngữ biên dịch như C/C++.

Không tối ưu cho ứng dụng di động: Python ít được sử dụng để phát triển ứng dụng di động so với Java, Swift.

Quản lý bộ nhớ: Python sử dụng cơ chế garbage collection, có thể gây khó khăn trong các ứng dụng yêu cầu quản lý bộ nhớ chi tiết.

Hạn chế trong lập trình hệ thống: Ít phù hợp cho các tác vụ cấp thấp như viết driver hoặc hệ điều hành.

### **2.7.4 Ứng dụng Python trong xử lý tài liệu PDF và tìm kiếm ngữ nghĩa**

Trong đồ án này, Python được sử dụng để xử lý các tài liệu PDF và thực hiện phân tích nội dung nhờ vào các thư viện hỗ trợ mạnh mẽ.

Cụ thể, các tác vụ chính được thực hiện bằng Python bao gồm:

Trích xuất nội dung từ tài liệu PDF bằng thư viện OCR.

Sinh vector embedding từ nội dung văn bản sử dụng mô hình SentenceTransformer (mô hình paraphrase-multilingual-MiniLM-L12-v2).

Kết nối và ghi dữ liệu vào Elasticsearch, hệ quản trị dữ liệu tìm kiếm, hỗ trợ tìm kiếm ngữ nghĩa thông minh.

Python được chọn cho hệ thống này nhờ:

Tính năng mạnh mẽ về xử lý ngôn ngữ tự nhiên (Natural Language Processing).

Hệ sinh thái phong phú với nhiều thư viện chuyên dụng.

Tương thích tốt với các hệ thống backend như Elasticsearch và Node.js thông qua API hoặc script ngoại.

### 2.7.5 Thư viện OCR

Nhận dạng ký tự quang học (Optical Character Recognition - OCR) là công nghệ cho phép chuyển đổi hình ảnh chứa văn bản (như tài liệu PDF quét, ảnh chụp) thành văn bản có thể chỉnh sửa và tìm kiếm được. Trong Python, pytesseract là một thư viện mã nguồn mở phổ biến, được sử dụng để giao tiếp với công cụ OCR Tesseract, do Google phát triển và duy trì. Tesseract ban đầu được phát hành vào năm 1985 bởi HP, sau đó trở thành mã nguồn mở vào năm 2005 và được Google cải tiến đáng kể [13].

pytesseract cung cấp giao diện Python để gọi Tesseract, giúp trích xuất văn bản từ hình ảnh hoặc tệp PDF chứa hình ảnh, hỗ trợ nhiều ngôn ngữ, bao gồm tiếng Việt. Thư viện này phù hợp để xử lý các tài liệu PDF quét trong hệ thống tìm kiếm tài liệu, nơi nội dung văn bản không có sẵn trực tiếp.

#### **Các tính năng chính:**

- Trích xuất văn bản từ hình ảnh: pytesseract có thể nhận dạng và trích xuất văn bản từ các định dạng hình ảnh như PNG, JPEG, hoặc các trang PDF được chuyển đổi thành hình ảnh.
- Hỗ trợ đa ngôn ngữ: Tesseract hỗ trợ hơn 100 ngôn ngữ, bao gồm tiếng Việt, thông qua các gói dữ liệu ngôn ngữ được huấn luyện trước.
- Xử lý bố cục phức tạp: Có khả năng nhận dạng văn bản trong các tài liệu có bố cục phức tạp như bảng, cột, hoặc văn bản in nghiêng, dù hiệu quả phụ thuộc vào chất lượng hình ảnh.
- Tùy chỉnh cấu hình: Cho phép điều chỉnh các tham số OCR (như chế độ phân đoạn trang, loại ký tự) để tối ưu hóa kết quả nhận dạng.
- Tích hợp với Python: Dễ dàng kết hợp với các thư viện như PIL (Python Imaging Library) hoặc fitz (PyMuPDF) để xử lý hình ảnh hoặc PDF trước khi OCR.

Ứng dụng trong hệ thống:

- Trong hệ thống tìm kiếm tài liệu thông minh, pytesseract được sử dụng để xử lý các tệp PDF quét không chứa văn bản có thể trích xuất trực tiếp.
- Trích xuất văn bản từ PDF quét: Kết hợp với fitz để chuyển các trang PDF thành hình ảnh, sau đó sử dụng pytesseract để nhận dạng và trích xuất văn bản.
- Chuẩn bị dữ liệu cho Elasticsearch: Văn bản trích xuất được gửi đến Elasticsearch để lập chỉ mục, hỗ trợ tìm kiếm từ khóa và ngữ nghĩa.
- Hỗ trợ tài liệu đa dạng: Đảm bảo hệ thống có thể xử lý cả PDF văn bản và PDF hình ảnh, tăng tính linh hoạt khi làm việc với các tài liệu học thuật hoặc hành chính.

#### **Ưu điểm:**

- Mã nguồn mở và miễn phí: pytesseract và Tesseract đều miễn phí, phù hợp cho các dự án học thuật hoặc triển khai quy mô nhỏ.
- Hỗ trợ tiếng Việt: Có gói dữ liệu ngôn ngữ tiếng Việt (vie), giúp xử lý tài liệu tiếng Việt hiệu quả.
- Dễ tích hợp: Cú pháp Python đơn giản, dễ dàng sử dụng trong pipeline xử lý dữ liệu cùng các thư viện như fitz hoặc sentence-transformers.
- Cộng đồng lớn: Tesseract có cộng đồng phát triển rộng, với nhiều tài liệu và hướng dẫn trên GitHub và các diễn đàn.

#### **Nhược điểm:**

- Phụ thuộc vào chất lượng hình ảnh: Độ chính xác của OCR giảm nếu hình ảnh mờ, nghiêng, hoặc có nhiễu.
- Hiệu suất chậm với tài liệu lớn: Xử lý các PDF nhiều trang hoặc hình ảnh độ phân giải cao có thể mất nhiều thời gian.
- Yêu cầu cài đặt bổ sung: Cần cài đặt Tesseract trên hệ thống và cấu hình đường dẫn đúng cho pytesseract.

- Hạn chế với bố cục phức tạp: Nhận dạng văn bản trong bảng hoặc tài liệu có định dạng đặc biệt có thể không chính xác nếu không tiền xử lý tốt.

### **2.7.6 Thư viện sentence-transformers**

Sentence-Transformers là một thư viện Python mã nguồn mở, được phát triển dựa trên Transformers của Hugging Face, chuyên dùng để tạo và sử dụng các biểu diễn vector (embeddings) cho văn bản, đặc biệt là các câu hoặc đoạn văn ngắn. Thư viện này cung cấp các mô hình được huấn luyện trước và hỗ trợ tinh chỉnh (fine-tuning) để phục vụ các tác vụ xử lý ngôn ngữ tự nhiên (NLP)[9].

#### **Các tính năng chính:**

Tạo embeddings văn bản: Chuyển đổi câu hoặc đoạn văn thành các vector số để sử dụng trong các tác vụ như tìm kiếm ngữ nghĩa (semantic search), so sánh độ tương đồng văn bản (semantic textual similarity), hoặc khai thác câu tương đồng (paraphrase mining).

Hỗ trợ mô hình tiên tiến: Cung cấp hơn 10.000 mô hình được huấn luyện trước từ Hugging Face, bao gồm BERT, RoBERTa, DistilBERT,... với hiệu suất cao trên bảng xếp hạng MTEB (Massive Text Embedding Benchmark)[9].

Cross-Encoder: Tính toán điểm tương đồng giữa các cặp câu bằng các mô hình Cross-Encoder.

Tinh chỉnh mô hình: Dễ dàng huấn luyện hoặc tinh chỉnh mô hình cho các tác vụ cụ thể với hơn 20 hàm mất mát (loss functions) cho embeddings và 10 hàm cho Cross-Encoder.

Hỗ trợ đa ngôn ngữ: Hỗ trợ xử lý văn bản trong nhiều ngôn ngữ.

#### **Ưu điểm:**

- Dễ sử dụng, tích hợp tốt với hệ sinh thái Hugging Face.
- Hỗ trợ nhiều mô hình mạnh mẽ và đa dạng cho các tác vụ NLP.

- Tài liệu chi tiết tại [www.sbert.net](http://www.sbert.net) và mã nguồn trên GitHub.
- Linh hoạt trong việc tinh chỉnh mô hình cho các ứng dụng cụ thể.

**Nhược điểm:**

- Yêu cầu tài nguyên tính toán lớn khi sử dụng các mô hình phức tạp hoặc huấn luyện trên GPU.
- Cần kiến thức về NLP để tinh chỉnh mô hình hiệu quả.

**Ứng dụng:**

- Sentence-Transformers được sử dụng trong các ứng dụng như:
- Tìm kiếm ngữ nghĩa trong cơ sở dữ liệu văn bản.
- Phân loại văn bản, phát hiện câu tương đồng hoặc đối sánh.
- Xây dựng chatbot, hệ thống khuyến nghị, hoặc phân tích cảm xúc.

### **2.7.7 Thư viện elasticsearch**

Thư viện elasticsearch là một thư viện Python mã nguồn mở, được phát triển bởi Elastic, cung cấp giao diện Python để tương tác với Elasticsearch – công cụ tìm kiếm và phân tích dữ liệu mạnh mẽ. Thư viện này cho phép lập trình viên thực hiện các thao tác như tạo, quản lý chỉ mục (index), thêm, xóa, cập nhật tài liệu, và thực hiện các truy vấn tìm kiếm thông qua API Python một cách dễ dàng[7].

**Các tính năng chính:**

- Kết nối và quản lý Elasticsearch: Hỗ trợ kết nối đến cluster Elasticsearch, cấu hình xác thực và quản lý các node trong cluster.
- Quản lý chỉ mục: Tạo, xóa, cập nhật chỉ mục và ánh xạ (mapping) để định nghĩa cấu trúc dữ liệu.
- Thao tác với tài liệu: Thêm, cập nhật, xóa tài liệu trong chỉ mục, hỗ trợ cả thao tác đơn lẻ và hàng loạt (bulk).

- Truy vấn tìm kiếm: Hỗ trợ các loại truy vấn như match, term, bool, script\_score (cho tìm kiếm ngữ nghĩa), và các truy vấn nâng cao khác.
- Tích hợp dễ dàng: Tương thích tốt với các thư viện Python khác như pandas, numpy, hoặc sentence-transformers để xử lý dữ liệu trước khi đưa vào Elasticsearch.
- Hỗ trợ bất đồng bộ: Cung cấp phiên bản elasticsearch-async cho các ứng dụng yêu cầu xử lý bất đồng bộ.

### **Ứng dụng trong hệ thống:**

- Trong hệ thống tìm kiếm tài liệu thông minh, thư viện elasticsearch được sử dụng để:
  - Kết nối với Elasticsearch và cấu hình chỉ mục để lưu trữ nội dung tài liệu PDF và vector ngữ nghĩa.
  - Thực hiện các truy vấn tìm kiếm từ khóa (keyword search) và tìm kiếm ngữ nghĩa (semantic search) thông qua API Python.
  - Ghi dữ liệu (nội dung văn bản và vector embedding) vào Elasticsearch sau khi được xử lý bởi các script Python.

### **Ưu điểm:**

- Dễ sử dụng, cú pháp Python trực quan, phù hợp với các lập trình viên quen thuộc với Python.
- Hỗ trợ đầy đủ các tính năng của Elasticsearch thông qua API RESTful.
- Tài liệu tham khảo phong phú tại trang chính thức của Elastic.

### **Nhược điểm:**

- Yêu cầu cấu hình chính xác thông tin kết nối (host, port, authentication) để tránh lỗi kết nối.
- Hiệu suất có thể bị ảnh hưởng nếu xử lý khối lượng dữ liệu lớn mà không tối ưu hóa truy vấn hoặc cấu hình cluster.

### 2.7.8 Thư viện torch

PyTorch, thường được gọi là torch trong Python, là một thư viện học máy mã nguồn mở được phát triển bởi Facebook AI Research (FAIR). Ra mắt lần đầu vào năm 2016, PyTorch đã nhanh chóng trở thành một trong những framework học máy phổ biến nhất nhờ tính linh hoạt, dễ sử dụng và hiệu suất cao. Thư viện này được xây dựng dựa trên ngôn ngữ lập trình Python và tích hợp chặt chẽ với các thư viện khoa học như NumPy, cung cấp các công cụ mạnh mẽ để phát triển các mô hình học sâu (deep learning) và xử lý dữ liệu [11].

PyTorch hỗ trợ cả nghiên cứu và triển khai thực tế, cho phép lập trình viên xây dựng các mô hình từ đơn giản đến phức tạp, như mạng nơ-ron (neural networks), mô hình xử lý ngôn ngữ tự nhiên (NLP), hay thị giác máy tính (computer vision). Thư viện này nổi bật với cơ chế tính toán động (dynamic computation graph), giúp dễ dàng điều chỉnh mô hình trong quá trình phát triển.

#### **Các tính năng chính:**

Tính toán động (Dynamic Computation Graph): PyTorch sử dụng cơ chế “eager execution”, cho phép định nghĩa và thực thi đồ thị tính toán (computation graph) ngay trong quá trình chạy, giúp dễ dàng gỡ lỗi và thử nghiệm mô hình.

Hỗ trợ GPU: PyTorch tích hợp với CUDA, cho phép tận dụng sức mạnh của GPU để tăng tốc độ huấn luyện và suy luận mô hình, đặc biệt với các tác vụ yêu cầu tính toán lớn.

Hệ sinh thái phong phú: PyTorch đi kèm với các thư viện bổ trợ như torchvision (xử lý hình ảnh), torchaudio (xử lý âm thanh), và torchtext (xử lý văn bản), hỗ trợ đa dạng các ứng dụng học máy.

Tích hợp với Python: PyTorch được thiết kế để hoạt động mượt mà với Python, cho phép sử dụng các công cụ như NumPy, Pandas, hoặc Matplotlib để xử lý dữ liệu và trực quan hóa kết quả.



Tinh chỉnh mô hình dễ dàng: PyTorch cung cấp API trực quan để xây dựng, huấn luyện, và tinh chỉnh (fine-tuning) các mô hình học sâu, phù hợp cho cả người mới bắt đầu và chuyên gia.

### **Ứng dụng trong hệ thống:**

Trong hệ thống tìm kiếm tài liệu thông minh, PyTorch được sử dụng để hỗ trợ quá trình sinh vector ngữ nghĩa (semantic embeddings) thông qua các mô hình học sâu, đặc biệt khi kết hợp với thư viện sentence-transformers. Cụ thể:

Tạo embeddings: PyTorch cung cấp nền tảng tính toán để chạy các mô hình như BERT hoặc MiniLM, giúp chuyển đổi văn bản thành các vector số biểu diễn ý nghĩa ngữ cảnh.

Tìm kiếm ngữ nghĩa: Các vector được sinh ra từ mô hình chạy trên PyTorch được lưu trữ trong Elasticsearch, hỗ trợ truy vấn tìm kiếm ngữ nghĩa dựa trên độ tương đồng cosine.

Tích hợp với Python: PyTorch tương thích tốt với các thư viện Python khác như sentence-transformers và elasticsearch, tạo ra một pipeline xử lý dữ liệu hiệu quả từ trích xuất văn bản đến tìm kiếm thông minh.

### **Ưu điểm:**

- Dễ sử dụng: Cú pháp Pythonic và API trực quan giúp PyTorch dễ học và sử dụng, đặc biệt với những người đã quen thuộc với Python.
- Linh hoạt: Cơ chế tính toán động cho phép thay đổi mô hình trong thời gian thực, lý tưởng cho nghiên cứu và thử nghiệm.
- Hiệu suất cao: Tối ưu hóa trên GPU và CPU, giúp xử lý các tác vụ học sâu với tốc độ nhanh.
- Cộng đồng lớn: PyTorch có cộng đồng phát triển rộng rãi, với nhiều tài liệu, hướng dẫn, và mô hình được chia sẻ trên các nền tảng như Hugging Face.
- Tích hợp tốt: Hỗ trợ tích hợp với các công cụ học máy khác và các hệ thống như Elasticsearch, Flask, hoặc Django.

**Nhược điểm:**

Yêu cầu tài nguyên lớn: Các mô hình học sâu trên PyTorch thường đòi hỏi GPU mạnh mẽ để huấn luyện hiệu quả, làm tăng chi phí phần cứng.

Học khúc ban đầu: Người mới có thể gặp khó khăn khi làm quen với các khái niệm như tensor hoặc autograd.

Triển khai phức tạp hơn so với một số framework: Mặc dù PyTorch đã cải thiện với TorchServe, việc triển khai mô hình vào sản xuất có thể phức tạp hơn so với TensorFlow trong một số trường hợp

### **2.7.9 Thư viện python-dotenv**

Thư viện python-dotenv là một thư viện Python mã nguồn mở, được phát triển để quản lý các biến môi trường trong các ứng dụng Python. Thư viện này cho phép đọc các cặp khóa-giá trị từ tệp .env và tải chúng vào biến môi trường của hệ thống, giúp quản lý cấu hình nhạy cảm (như thông tin đăng nhập, khóa API) một cách an toàn và tiện lợi[12].

**Các tính năng chính:**

Đọc tệp .env: Tự động tải các biến môi trường từ tệp .env vào os.environ.

Quản lý cấu hình an toàn: Lưu trữ thông tin nhạy cảm (như mật khẩu, URL cơ sở dữ liệu, khóa API) trong tệp .env thay vì mã nguồn.

Hỗ trợ đa dạng định dạng: Hỗ trợ các giá trị chuỗi, số, và boolean trong tệp .env.

Tích hợp dễ dàng: Dễ dàng sử dụng trong các dự án Python, đặc biệt khi kết hợp với các framework như Flask, Django, hoặc các script xử lý dữ liệu.

Hỗ trợ tệp tùy chỉnh: Cho phép chỉ định tên hoặc đường dẫn tệp cấu hình khác ngoài .env.

**Ứng dụng trong hệ thống:**

Trong hệ thống tìm kiếm tài liệu thông minh, python-dotenv được sử dụng để:

Quản lý các thông tin cấu hình như địa chỉ host, cổng của Elasticsearch, hoặc khóa API của các dịch vụ liên quan.

Đảm bảo các thông tin nhạy cảm (như thông tin xác thực Elasticsearch) không bị hard-code trong mã nguồn, tăng tính bảo mật.

Tăng tính di động của mã nguồn khi triển khai trên các môi trường khác nhau (local, staging, production).

**Ưu điểm:**

- Đơn giản, dễ tích hợp vào các dự án Python.
- Tăng cường bảo mật bằng cách tách biệt thông tin cấu hình khỏi mã nguồn.
- Hỗ trợ tốt cho việc triển khai ứng dụng trên nhiều môi trường.
- Tài liệu rõ ràng tại [github.com/theskumar/python-dotenv](https://github.com/theskumar/python-dotenv).

**Nhược điểm:**

Chỉ hỗ trợ các cấu hình đơn giản dưới dạng cặp khóa-giá trị, không phù hợp cho các cấu hình phức tạp.

Yêu cầu người dùng cẩn thận trong việc bảo vệ tệp .env để tránh rò rỉ thông tin.

## CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

### 3.1 Mô tả

Hệ thống được xây dựng nhằm hỗ trợ người dùng tìm kiếm và tra cứu các tài liệu học thuật hoặc văn bản nội bộ dưới định dạng PDF, thông qua hai phương pháp chính: tìm kiếm theo từ khóa (keyword search) và tìm kiếm ngữ nghĩa (semantic search). Đây là giải pháp kết hợp giữa các công nghệ hiện đại như ReactJS, Node.js, Python và Elasticsearch, giúp tăng hiệu quả trong việc truy xuất thông tin từ các tài liệu phi cấu trúc.

#### Kiến trúc tổng thể

Frontend (Giao diện người dùng):

Sử dụng ReactJS kết hợp Tailwind CSS để cung cấp giao diện tìm kiếm, đăng ký/đăng nhập, và quản lý tài liệu (nếu người dùng có quyền). Người dùng có thể tương tác dễ dàng, tìm kiếm theo thời gian thực và xem thông tin tài liệu ngay trên giao diện.

Backend (Máy chủ):

Được phát triển bằng Node.js và ExpressJS, backend xử lý các yêu cầu API, xác thực người dùng bằng JWT, phân quyền (user và admin), quản lý tải lên tài liệu và kết nối đến Elasticsearch. Ngoài ra, backend còn thực thi các script xử lý tài liệu thông qua mô-đun Python.

Xử lý tài liệu (Python module):

Mã Python thực hiện việc trích xuất nội dung từ các file PDF (dùng thư viện fitz hoặc pytesseract đối với tài liệu scan), sinh vector ngữ nghĩa từ văn bản bằng mô hình SentenceTransformer, sau đó gửi dữ liệu lên Elasticsearch để đánh chỉ mục phục vụ truy vấn ngữ nghĩa.

Cơ sở dữ liệu tìm kiếm (Elasticsearch):

Là nơi lưu trữ nội dung tài liệu và các vector embedding. Elasticsearch cho phép truy vấn nhanh chóng theo từ khóa hoặc theo độ tương đồng ngữ nghĩa (cosine similarity), trả về kết quả phù hợp với truy vấn người dùng.

Luồng xử lý hệ thống:

Đăng nhập và Đăng ký: Người dùng có thể đăng ký tài khoản bằng email, sau đó nhận mã OTP để xác minh và kích hoạt tài khoản. Khi đăng nhập thành công, hệ thống sẽ tự động phân quyền truy cập theo vai trò, bao gồm quản trị viên (admin) và người dùng thông thường (user).

Tải lên tài liệu (Chỉ dành cho Admin): Người dùng có quyền admin được phép tải lên một hoặc nhiều tệp PDF thông qua giao diện web. Sau khi tiếp nhận, backend sẽ lưu trữ các tệp vào thư mục cục bộ và gọi script Python để thực hiện chuỗi xử lý bao gồm: trích xuất nội dung văn bản từ tài liệu, sinh vector ngữ nghĩa từ nội dung và tiến hành đánh chỉ mục (index) cả nội dung lẫn vector vào Elasticsearch.

Người dùng nhập truy vấn, có thể là từ khóa hoặc câu truy vấn ngữ nghĩa, thông qua giao diện frontend. Frontend gửi yêu cầu tìm kiếm đến backend, nơi truy vấn được phân loại để xử lý. Với tìm kiếm từ khóa, backend sử dụng multi\_match query để truy vấn Elasticsearch. Đối với tìm kiếm ngữ nghĩa, backend gọi script Python để tạo vector từ truy vấn, sau đó thực hiện script\_score query trên Elasticsearch. Kết quả tìm kiếm được trả về, sắp xếp theo độ phù hợp, và hiển thị mượt mà trên giao diện người dùng.

Xem và quản lý tài liệu: Sau khi tìm kiếm, người dùng có thể xem các thông tin mô tả liên quan đến tài liệu như tiêu đề và đoạn trích nội dung. Đối với admin, hệ thống cung cấp thêm các chức năng như xóa, cập nhật hoặc xác minh lại nội dung của các tài liệu đã được tải lên.

Lịch sử tìm kiếm: Mỗi truy vấn của người dùng được ghi lại (nếu đã đăng nhập), phục vụ mục đích thống kê và quản trị.

### 3.2 Yêu cầu chức năng

#### **Đăng ký, đăng nhập người dùng:**

- Cho phép người dùng đăng ký và đăng nhập bằng email, mật khẩu.
- Quản lý phân quyền theo vai trò (user, admin).

#### **Tải lên tài liệu (Admin):**

- Cho phép người dùng có vai trò admin tải lên nhiều file PDF.
- Hệ thống tự động xử lý file và đưa vào kho dữ liệu tìm kiếm.

#### **Tìm kiếm tài liệu:**

- Hỗ trợ tìm kiếm theo từ khóa (keyword search).
- Hỗ trợ tìm kiếm ngữ nghĩa (semantic search) sử dụng embedding.

#### **Xem nội dung tài liệu:**

- Cho phép người dùng xem thông tin mô tả của tài liệu đã tìm thấy.

#### **Giao diện web thân thiện:**

- Tương tác thời gian thực, hiển thị kết quả mượt mà.

### 3.3 Yêu cầu phi chức năng

#### **Hiệu năng:**

- Tìm kiếm tài liệu phải trả kết quả trong vòng dưới 2 giây.
- Tải lên tài liệu không ảnh hưởng đến trải nghiệm người dùng khác.

#### **Bảo mật:**

- Mã hóa mật khẩu người dùng (sử dụng bcrypt).
- Xác thực và phân quyền sử dụng JWT (JSON Web Token).

### 3.4 Kiến trúc hệ thống

Hệ thống được thiết kế dựa trên mô hình Client-Server, kết hợp định hướng microservice đơn giản, đảm bảo tính linh hoạt và khả năng mở rộng trong tương

lai. Các thành phần chính được tách biệt rõ ràng theo chức năng, giao tiếp với nhau thông qua giao thức HTTP và API RESTful.

Frontend (ReactJS + Tailwind CSS): Cung cấp giao diện người dùng hiện đại, cho phép thực hiện các chức năng như đăng ký, đăng nhập, tìm kiếm tài liệu, xem thông tin mô tả và quản lý tài liệu (nếu có quyền). Frontend tương tác với backend thông qua các API được định nghĩa rõ ràng.

Backend (Node.js + ExpressJS): Chịu trách nhiệm xử lý các yêu cầu từ frontend, xác thực người dùng thông qua JWT, phân quyền (user/admin), quản lý tài liệu và gọi các mô-đun xử lý dữ liệu bằng Python. Đây là trung tâm điều phối giữa giao diện người dùng và các dịch vụ phía sau.

Xử lý tài liệu (Python module): Thực thi việc trích xuất nội dung từ các file PDF và sinh vector ngữ nghĩa bằng mô hình SentenceTransformer. Dữ liệu sau xử lý sẽ được gửi lên Elasticsearch để lưu trữ và phục vụ cho truy vấn ngữ nghĩa.

Elasticsearch: Đóng vai trò là hệ quản trị dữ liệu tìm kiếm. Nó lưu trữ nội dung tài liệu và các vector embedding, đồng thời cung cấp cơ chế tìm kiếm theo từ khóa (multi\_match) và tìm kiếm ngữ nghĩa (script\_score).

### **3.5 Luồng xử lý tìm kiếm**

Người dùng nhập truy vấn vào thanh tìm kiếm trên giao diện React (truy vấn có thể là từ khóa hoặc một câu đầy đủ).

Frontend gửi truy vấn đến backend thông qua API: GET /api/search?query=...&type=....

Backend (Node.js) tiếp nhận truy vấn và xác định loại tìm kiếm:

- Nếu là từ khóa (keyword search), backend gọi trực tiếp đến Elasticsearch với truy vấn multi\_match.
- Nếu là ngữ nghĩa (semantic search), backend gọi một script Python để sinh vector ngữ nghĩa từ truy vấn. Vector này sau đó được gửi vào

Elasticsearch sử dụng truy vấn `script_score`, tính toán độ tương đồng cosine.

Elasticsearch xử lý truy vấn và trả về danh sách tài liệu phù hợp, sắp xếp theo mức độ tương đồng.

Backend nhận kết quả, đóng gói và trả lại cho frontend.

Frontend hiển thị danh sách tài liệu cho người dùng, kèm theo thông tin mô tả như tiêu đề, đoạn trích nội dung và liên kết mở PDF.

### **3.6 Mô tả về các chức năng của hệ thống**

Hệ thống tìm kiếm tài liệu thông minh được xây dựng nhằm phục vụ nhu cầu tra cứu thông tin của người dùng cuối lẫn quản trị viên, với các chức năng được thiết kế cụ thể như sau:

1. Đăng ký và xác minh tài khoản: Người dùng có thể đăng ký tài khoản bằng email và mật khẩu. Hệ thống sẽ gửi mã OTP đến email để xác minh danh tính trước khi kích hoạt tài khoản. Sau khi xác minh thành công, người dùng có thể đăng nhập để sử dụng hệ thống.

2. Đăng nhập và phân quyền: Người dùng đăng nhập vào hệ thống để truy cập các chức năng tương ứng với vai trò của họ. Hệ thống phân quyền thành hai nhóm là người dùng thông thường (user) và quản trị viên (admin), mỗi nhóm có quyền truy cập vào các chức năng khác nhau.

3. Tìm kiếm tài liệu:

Tìm kiếm từ khoá: Người dùng nhập chuỗi từ khoá để tìm tài liệu theo tiêu đề hoặc nội dung.

Tìm kiếm ngữ nghĩa: Hệ thống sử dụng mô hình học máy (SentenceTransformer) để sinh vector từ câu truy vấn và tìm các tài liệu có nội dung tương tự về mặt ngữ nghĩa. Kết quả được hiển thị trực quan, cho phép mở tài liệu trực tiếp.



4. Ghi lịch sử tìm kiếm: Mỗi lượt tìm kiếm của người dùng đã đăng nhập đều được lưu lại với thông tin người dùng, từ khoá, thời gian và trường tìm kiếm (tiêu đề, nội dung). Quản trị viên có thể xem toàn bộ lịch sử này và lọc theo email hoặc ngày giúp phục vụ công tác quản lý.

5. Tải lên và quản lý tài liệu (Admin): Quản trị viên có thể tải lên nhiều tài liệu PDF một cách dễ dàng. Sau khi upload, hệ thống gọi script Python để trích xuất nội dung và sinh vector ngữ nghĩa, sau đó đưa lên Elasticsearch. Hệ thống cho phép quản trị viên xóa tài liệu hoặc tìm kiếm lại để xác minh nội dung.

#### 6. Quản lý tài khoản người dùng (Admin)

Quản trị viên có thể:

- Xem danh sách tất cả người dùng.
- Tìm kiếm theo email hoặc vai trò.
- Chỉnh sửa hoặc xóa tài khoản nếu cần thiết.
- Đây là phần quan trọng để bảo trì hệ thống và loại bỏ tài khoản sai phạm hoặc không xác minh.

### 3.7 Mô tả về actor

#### 1. Người dùng (User)

Người dùng là đối tượng chính sử dụng hệ thống để tra cứu tài liệu. Vai trò của họ bao gồm:

- Đăng ký tài khoản: Tạo tài khoản mới và xác minh bằng mã OTP gửi qua email.
- Đăng nhập: Truy cập vào hệ thống với vai trò người dùng.
- Tìm kiếm tài liệu: Sử dụng chức năng tìm kiếm từ khóa hoặc tìm kiếm ngữ nghĩa.
- Xem chi tiết tài liệu: Bao gồm tiêu đề, đoạn trích, hoặc xem nội dung tài liệu PDF.

- Giao diện trực quan: Sử dụng giao diện thân thiện, dễ thao tác, phù hợp với mọi đối tượng người dùng. Người dùng không có quyền quản lý tài liệu hoặc tài khoản người dùng khác, giúp đảm bảo tính bảo mật và phân quyền.

## 2. Quản trị viên (Admin)

Quản trị viên là người có quyền cao nhất trong hệ thống, ngoài các chức năng như người dùng còn có thêm:

Quản lý tài liệu:

- Tải lên nhiều file PDF.
- Xoá các tài liệu không còn giá trị.
- Kiểm tra dữ liệu sau khi index lên Elasticsearch.

Quản lý người dùng:

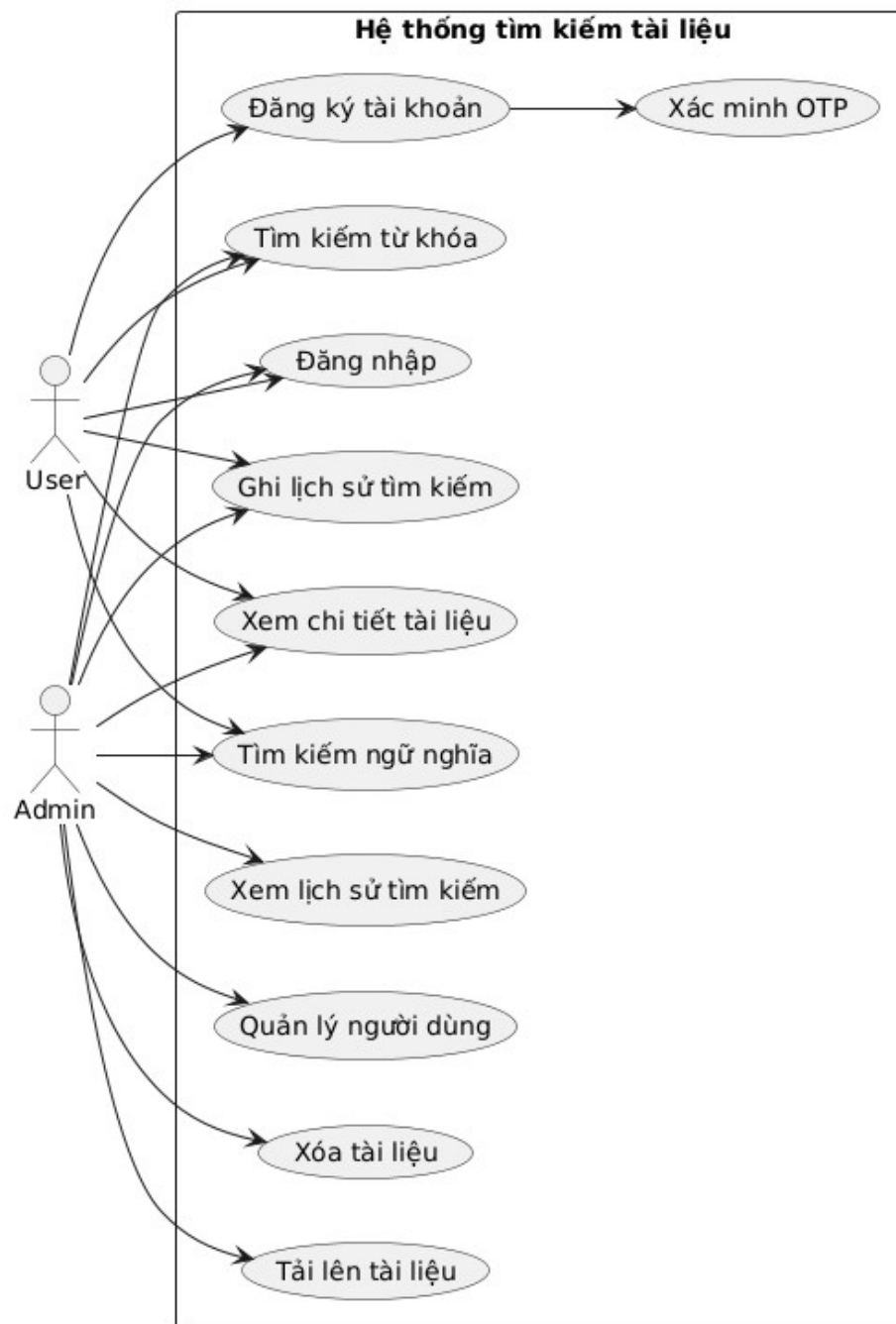
- Xem danh sách người dùng đã đăng ký.
- Lọc theo vai trò hoặc email.
- Chỉnh sửa hoặc xoá tài khoản bất kỳ.

Xem lịch sử tìm kiếm:

- Theo dõi tất cả truy vấn đã thực hiện.
- Lọc truy vấn theo thời gian, từ khóa hoặc email người dùng.

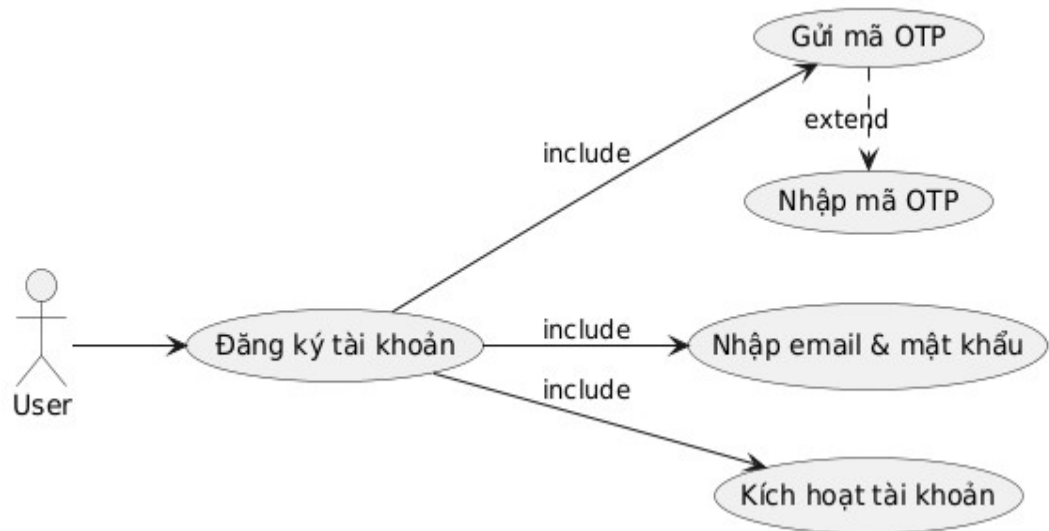
### 3.8 Lược đồ use case

Use case tổng quan:



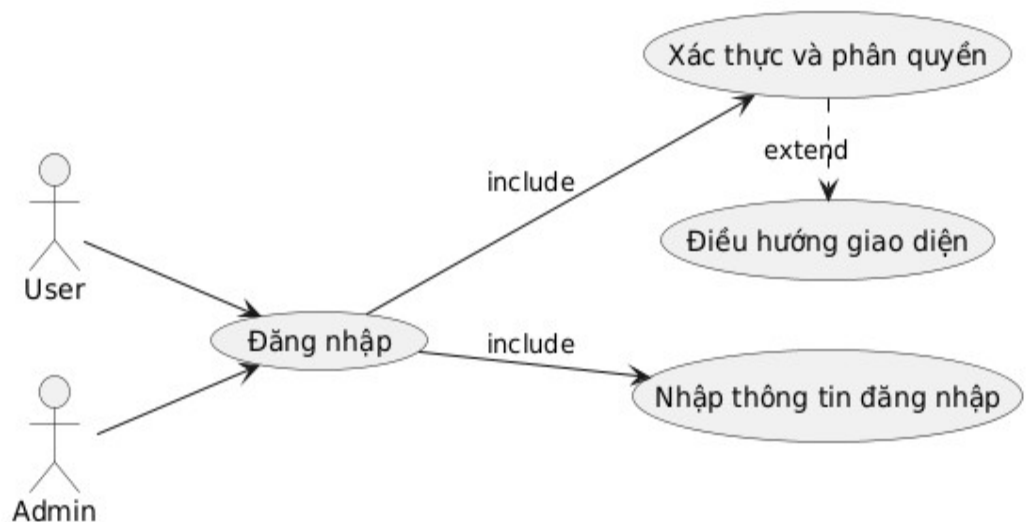
Hình 3.1 Use Case tổng quan

Use case đăng ký:



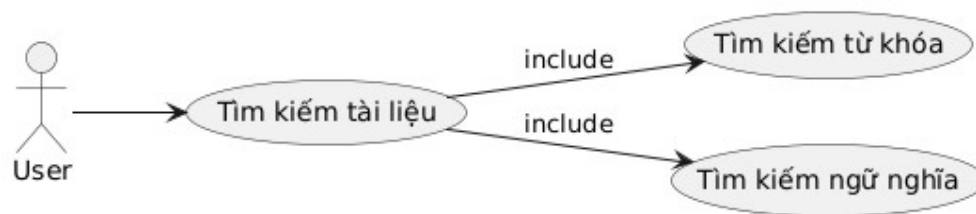
Hình 3.2 Use case đăng ký

Use case đăng nhập:



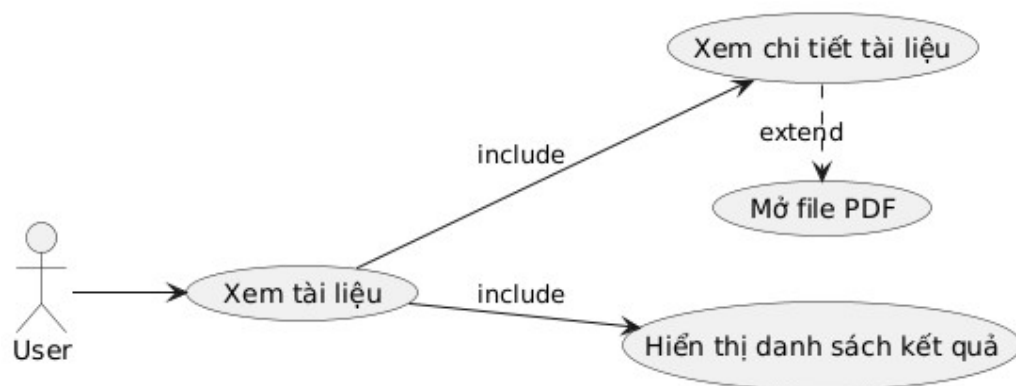
Hình 3.3 Use case đăng nhập

Use case tìm kiếm tài liệu:



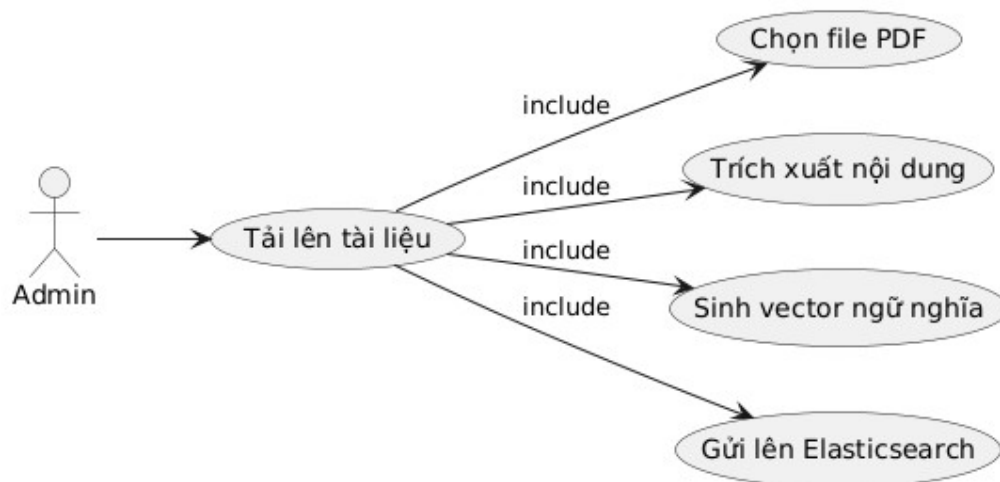
Hình 3.4 Use case tìm kiếm tài liệu

Use case xem tài liệu:



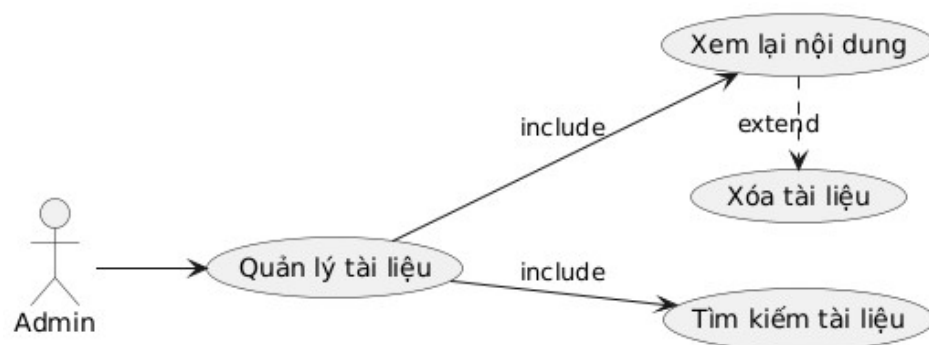
Hình 3.5 Use case xem tài liệu

Use case tải lên tài liệu:



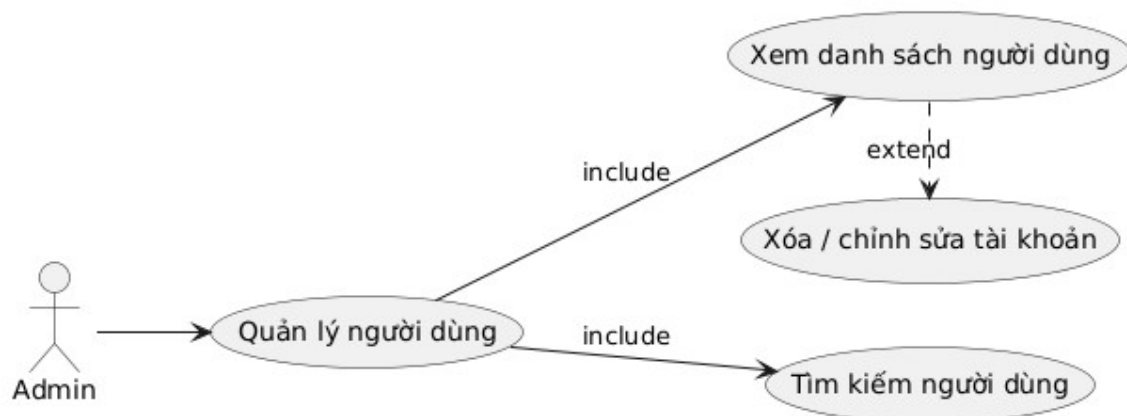
Hình 3.6 Use case tải lên tài liệu

Use case quản lý tài liệu:



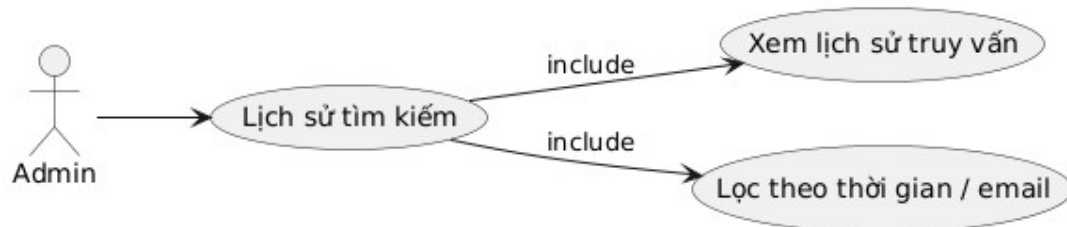
Hình 3.7 Use case quản lý tài liệu

Use case quản lý người dùng:



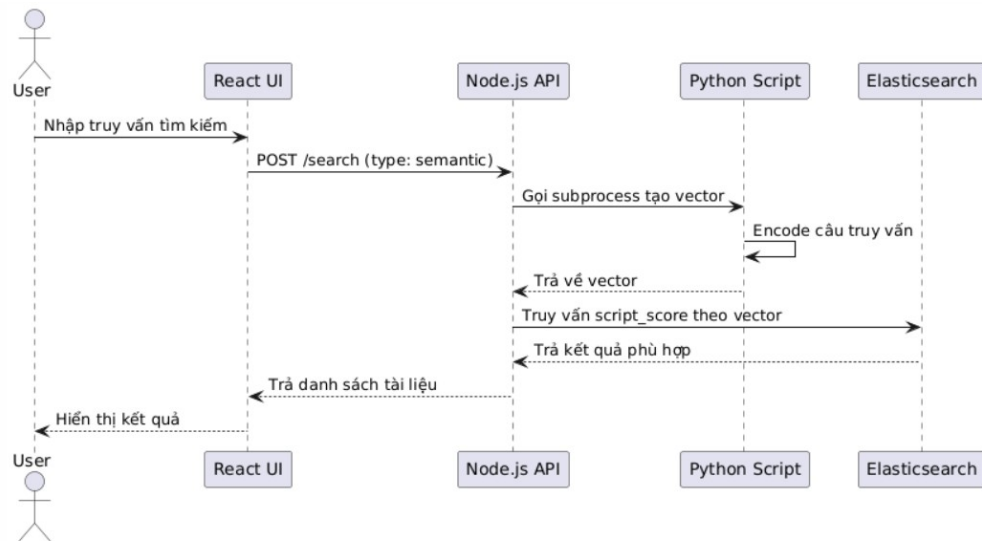
Hình 3.8 Use case quản lý người dùng

Use case lịch sử tìm kiếm:



Hình 3.9 Use case lịch sử tìm kiếm

### 3.9 Lược đồ tuần tự - Quá trình tìm kiếm tài liệu



Hình 3.10 Lược đồ tuần tự

Lược đồ tuần tự mô tả chi tiết trình tự các bước diễn ra trong quá trình người dùng tìm kiếm tài liệu, đặc biệt với tìm kiếm ngữ nghĩa (semantic search):

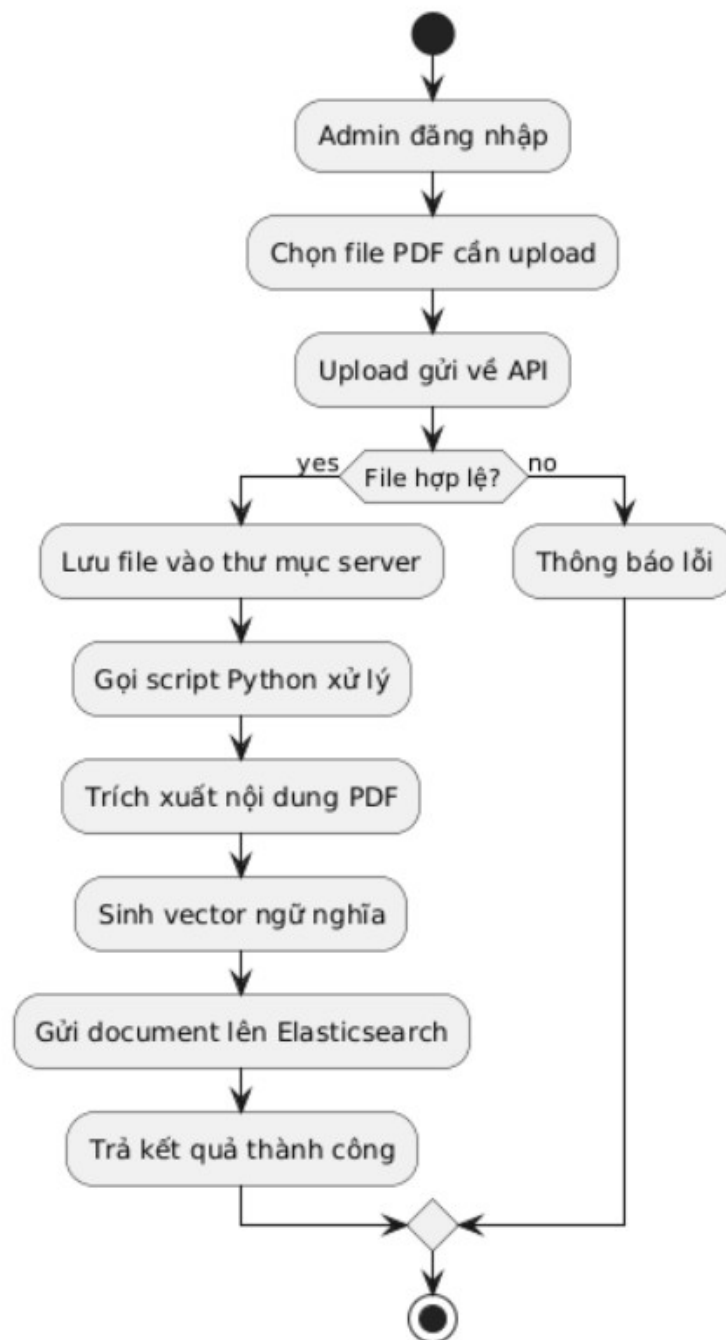
Mô tả quy trình:

- Người dùng nhập truy vấn (từ khóa/câu) vào giao diện tìm kiếm (React).
- Frontend gửi yêu cầu POST /search đến backend Node.js.

Backend kiểm tra kiểu tìm kiếm:

- Nếu là ngữ nghĩa, backend gọi script Python để tạo vector embedding.
- Script Python xử lý và trả lại vector biểu diễn truy vấn.
- Backend gửi truy vấn vector đến Elasticsearch với cơ chế script\_score để tính độ tương đồng cosine.
- Elasticsearch trả về danh sách tài liệu phù hợp.
- Backend gửi kết quả về lại frontend.
- React hiển thị danh sách kết quả cho người dùng.

### 3.10 Lược đồ hoạt động - Luồng xử lý upload tài liệu



Hình 3.11 Lược đồ hoạt động



Lược đồ hoạt động mô tả quy trình khi admin tải lên tài liệu PDF để hệ thống xử lý và đưa vào kho tìm kiếm.

Mô tả quy trình:

- Admin đăng nhập vào hệ thống.
- Chọn file PDF cần upload.
- Frontend gửi file lên backend qua API /upload.
- Backend lưu file vào thư mục tài liệu.
- Backend gọi script Python để:
  - Trích xuất nội dung văn bản từ PDF.
  - Sinh vector ngữ nghĩa từ nội dung.
  - Dữ liệu được gửi đến Elasticsearch để đánh chỉ mục.
- Backend thông báo kết quả thành công.

### **3.11 Các bước cài đặt Elasticsearch**

#### **Cài đặt Java Development Kit:**

Elasticsearch yêu cầu môi trường Java để chạy. Các bước cài đặt JDK trên Windows như sau:

Bước 1: Tải JDK bằng cách truy cập trang web chính thức của Oracle (<https://www.oracle.com/java/technologies/javase-downloads.html>)

Tải phiên bản JDK 11 hoặc 17 phù hợp với hệ điều hành Windows (64-bit).

Bước 2: Chạy file cài đặt (.exe) đã tải về và làm theo hướng dẫn trên màn hình. Ghi nhớ thư mục cài đặt JDK (mặc định thường là C:\Program Files\Java\jdk-<version>).

Bước 3: Tiến hành thiết lập biến môi trường bằng cách nhấn chuột phải vào This PC > Properties > Advanced system settings > Environment Variables.

Trong phần System Variables, nhấn New và điền các thông tin sau cho biến:

Variable name: JAVA\_HOME

Variable value: Đường dẫn đến thư mục JDK (C:\Program Files\Java\jdk-17).

Trong phần System variables, tìm biến có tên là Path, nhấn Edit sau đó nhấn New và thêm dòng %JAVA\_HOME%\bin.

Bước 4: Sau khi cài đặt mở Command Prompt (nhấn Win + R, gõ cmd và nhấn Enter).

Gõ lệnh java -version để kiểm tra. Nếu cài đặt thành công, thông tin phiên bản Java sẽ hiển thị.

### **Tải và cài đặt Elasticsearch:**

Bước 1: Tải Elasticsearch:

Truy cập trang web chính thức của Elastic  
(<https://www.elastic.co/downloads/elasticsearch>).

Tải phiên bản Elasticsearch mới nhất dưới dạng file ZIP cho Windows.

Bước 2: Giải nén file ZIP vào một thư mục, ví dụ: C:\elasticsearch-<version> (nên tránh thư mục có dấu cách hoặc ký tự đặc biệt để hạn chế lỗi).

Bước 3: Cấu hình Elasticsearch bằng cách mở thư mục vừa giải nén, vào thư mục config và chỉnh sửa file elasticsearch.yml bằng trình soạn thảo văn bản (như Notepad hoặc Visual Studio Code).

Ví dụ về thiết lập các thông số cơ bản trong file elasticsearch.yml:

cluster.name: my-cluster

node.name: node-1

network.host: 127.0.0.1

http.port: 9200

Bước 4: Cài đặt Elasticsearch như một dịch vụ của Windows. Để chạy Elasticsearch như một dịch vụ Windows, vào thư mục bin và chạy lệnh `elasticsearch-service.bat install`.

#### **Khởi động Elasticsearch:**

Sau khi cài đặt và cấu hình Elasticsearch, để tiện lợi cho việc khởi động và quản lý, ta có thể thiết lập để Elasticsearch chạy như một dịch vụ Windows. Các bước thực hiện như sau:

Bước 1: Thêm Elasticsearch vào biến môi trường Path

Nhấn chuột phải vào This PC > chọn Properties.

Nhấn Advanced system settings > chọn Environment Variables....

Tại khu vực System variables, tìm đến biến Path, chọn Edit....

Nhấn New, rồi dán vào đường dẫn tới thư mục bin của Elasticsearch, ví dụ:

`C:\elasticsearch-8.13.0\bin`

Nhấn OK liên tục để lưu thay đổi.

Bước 2: Khởi động dịch vụ Elasticsearch, mở Command Prompt và chạy lệnh **`elasticsearch-service.bat start`** để khởi động Elasticsearch.

Có thể cấu hình để Elasticsearch tự động chạy bằng cách:

Nhấn Windows + R, gõ `services.msc`, nhấn Enter. Trong danh sách dịch vụ, tìm dịch vụ có tên Elasticsearch hoặc `elasticsearch-service-x64`. Nhấn chuột phải > chọn Start để khởi động dịch vụ. (Tuỳ chọn) Đặt chế độ khởi động tự động:

Chuột phải > Properties > mục Startup type > chọn Automatic.

### **3.12 Cài đặt môi trường phát triển**

#### **Cài đặt Node.js:**

Bước 1: Truy cập trang chính thức của Node.js tại địa chỉ <https://Node.js.org/en>.

Bước 2: Tải phiên bản Node.js LTS (Long-Term Support).



*Hình 3.12 Node.js*

Bước 3: Bước 3: Chạy file cài đặt và làm theo các bước hướng dẫn trên màn hình.

Bước 4: Sau khi cài đặt, kiểm tra xem Node.js đã được cài đặt vào máy tính hay chưa bằng cách mở terminal và gõ lệnh `node -v` và `npm -v`. Nếu xuất hiện phiên bản của Node.js và npm thì đã cài đặt thành công.

### **Cài đặt React:**

Bước 1: Mở terminal và di chuyển đến thư mục muốn tạo dự án bằng lệnh `cd` đường-dẫn-thư-mục.

Bước 2: Sau khi di chuyển vào đường dẫn thư mục mong muốn thì gõ lệnh `npm create vite@latest tên-dự-án`. Sau đó chọn Framework React và ngôn ngữ JavaScript.

Bước 3: Di chuyển vào thư mục dự án bằng lệnh `cd tên-dự-án` và cài đặt các gói phụ thuộc bằng cách gõ lệnh `npm install`.

Bước 4: Chạy server phát triển bằng cách gõ lệnh `npm-run-dev`.

### **3.13 Cài đặt các thư viện Python**

Trong hệ thống tìm kiếm tài liệu thông minh, Python đóng vai trò quan trọng trong việc xử lý nội dung tài liệu PDF, sinh vector ngữ nghĩa và giao tiếp với

Elasticsearch. Toàn bộ các thao tác này được thực hiện thông qua một script Python chạy trong môi trường ảo được thiết lập sẵn.

### **Thiết lập môi trường ảo:**

Để đảm bảo tính ổn định và tái sử dụng cao khi chạy các chương trình Python xử lý PDF và tìm kiếm ngữ nghĩa, hệ thống sử dụng môi trường ảo tên là `nlp_env` được quản lý bởi Miniconda. Dưới đây là các bước chi tiết để cài đặt và cấu hình môi trường này trên máy tính Windows:

#### **Bước 1: Tải và cài đặt Miniconda**

Miniconda là một phiên bản nhẹ hơn của Anaconda, dùng để quản lý môi trường và gói Python.

Truy cập trang chính thức: <https://docs.conda.io/en/latest/miniconda.html>

Chọn bản phù hợp với Windows (64-bit), nhấn tải về và tiến hành cài đặt theo mặc định.

#### **Bước 2: Mở Anaconda Prompt**

Sau khi cài xong Miniconda, mở Anaconda Prompt từ menu Start (không phải CMD thường) để chạy các lệnh cấu hình.

#### **Bước 3: Tạo môi trường ảo mới**

Tạo môi trường tên là `nlp_env` với phiên bản Python mong muốn bằng lệnh **`conda create -n nlp_env python=3.10`**

#### **Bước 4: Kích hoạt môi trường**

Sau khi tạo thành công, kích hoạt môi trường bằng lệnh **`conda activate nlp_env`**

#### **Bước 5: Cài đặt các thư viện cần thiết**

Sau khi vào môi trường `nlp_env`, tiến hành cài đặt các thư viện cần dùng bằng lệnh:

**`pip install pytesseract Pillow`**

**pip install elasticsearch**

**pip install sentence-transformers**

**pip install torch**

**pip install python-dotenv**

Bước 6: Kiểm tra xem các thư viện đã cài đặt

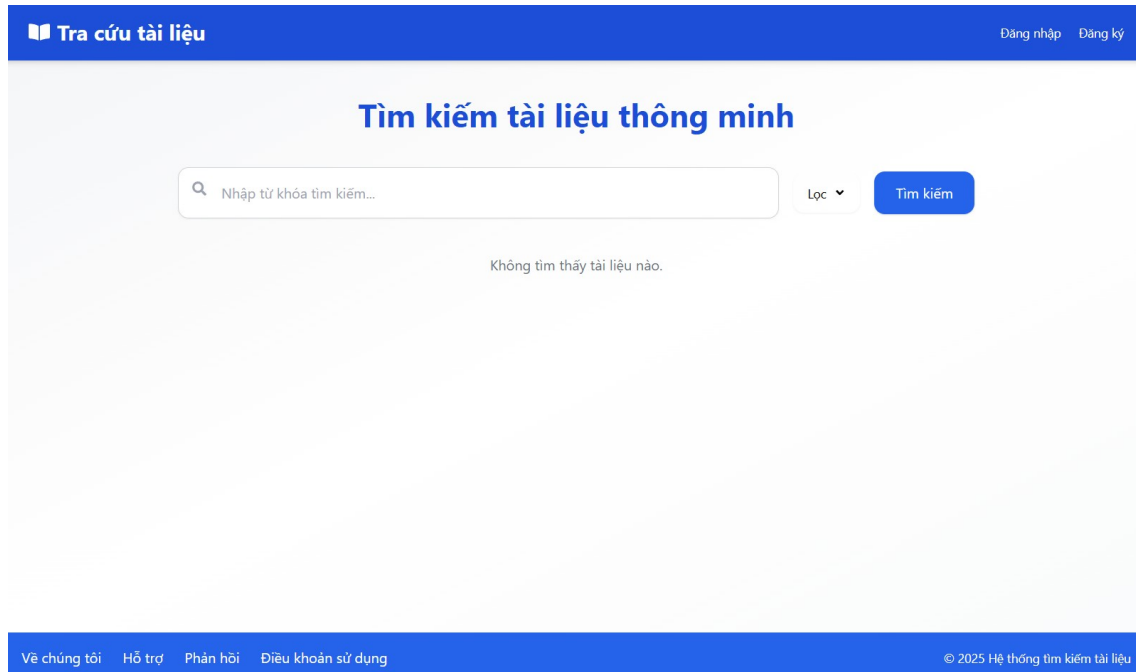
Sau khi cài đặt các thư viện cần thiết trong môi trường `nlp_env`, cần thực hiện kiểm tra để đảm bảo rằng các gói đã được cài đặt đúng và đầy đủ.

Trong Anaconda Prompt (khi đang ở môi trường `nlp_env`), nhập lệnh **pip list**. Lệnh này sẽ hiển thị danh sách tất cả các thư viện Python đã được cài đặt trong môi trường. Kiểm tra xem các thư viện sau có nằm trong danh sách hay không.

## CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

### 4.1 Giao diện trang chủ

Giao diện trang chủ của hệ thống được thiết kế đơn giản, trực quan. Trên thanh điều hướng có hai nút chính: Đăng nhập và Đăng ký, giúp người dùng dễ dàng truy cập vào hệ thống. Trung tâm trang là thanh tìm kiếm, cho phép người dùng nhập từ khóa để tra cứu tài liệu một cách nhanh chóng.



Hình 4.1 Trang chủ

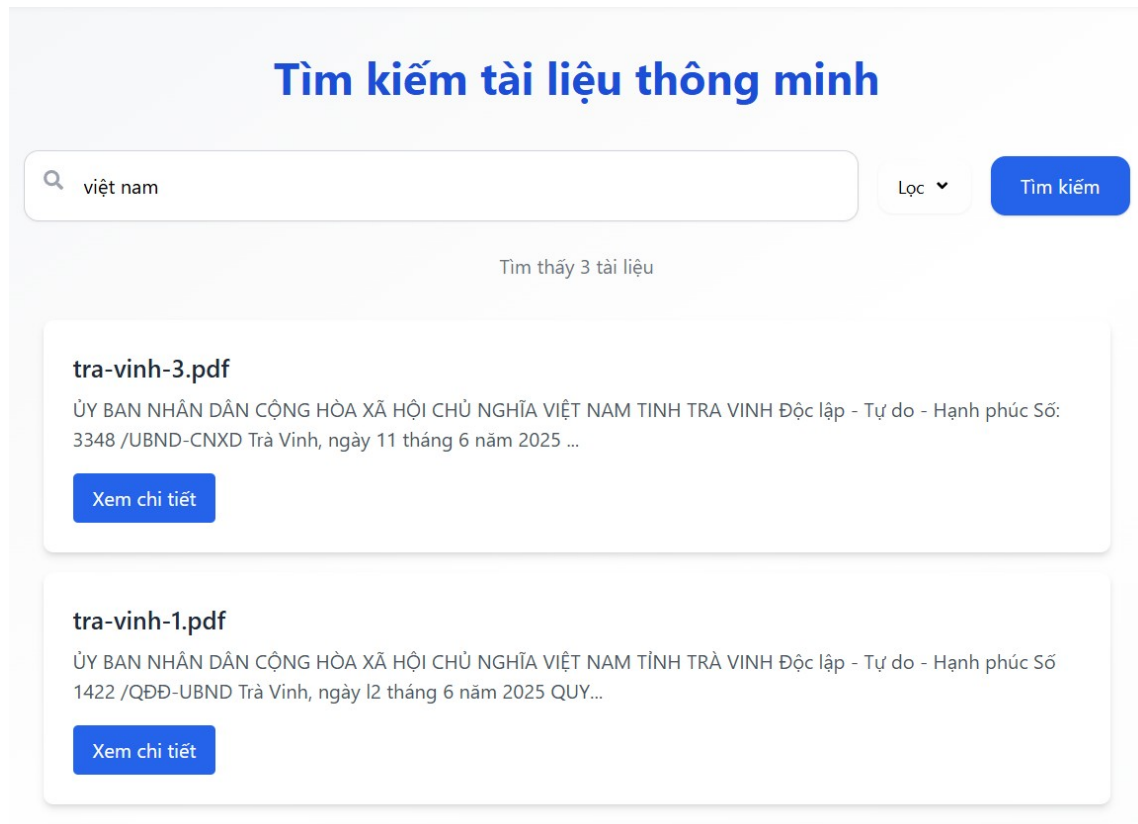
## 4.2 Giao diện tìm kiếm

Thanh tìm kiếm tài liệu thông minh được thiết kế với giao diện hiện đại, thân thiện, hỗ trợ người dùng tìm kiếm hiệu quả. Người dùng có thể nhập từ khóa tìm kiếm vào ô tìm kiếm và lựa chọn bộ lọc để thu hẹp kết quả. Đặc biệt, thanh tìm kiếm tích hợp chức năng lọc theo ngày ban hành của văn bản, cho phép chọn khoảng thời gian từ “Ngày ban hành” với hai trường ngày (mm/dd/yyyy) để xác định phạm vi thời gian mong muốn. Nút tìm kiếm sẽ kích hoạt quá trình tra cứu.

Hình 4.2 Giao diện tìm kiếm



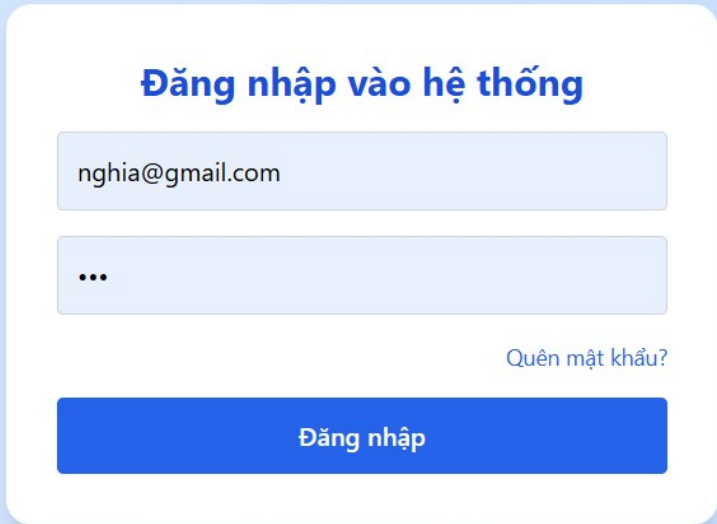
Sau khi thực hiện thao tác tìm kiếm, hệ thống sẽ hiển thị danh sách các tài liệu phù hợp. Mỗi mục trong danh sách bao gồm tiêu đề và đoạn trích nội dung, kèm theo nút “Xem chi tiết” để người dùng có thể truy cập và đọc toàn bộ nội dung tài liệu.



Hình 4.3 Giao diện trả kết quả

### 4.3 Giao diện đăng nhập

Giao diện đăng nhập bao gồm hai trường nhập liệu là email và mật khẩu, cùng với các nút chức năng như “Đăng nhập” và “Quên mật khẩu”. Thiết kế đơn giản, tập trung vào tính dễ sử dụng và bảo mật thông tin người dùng.

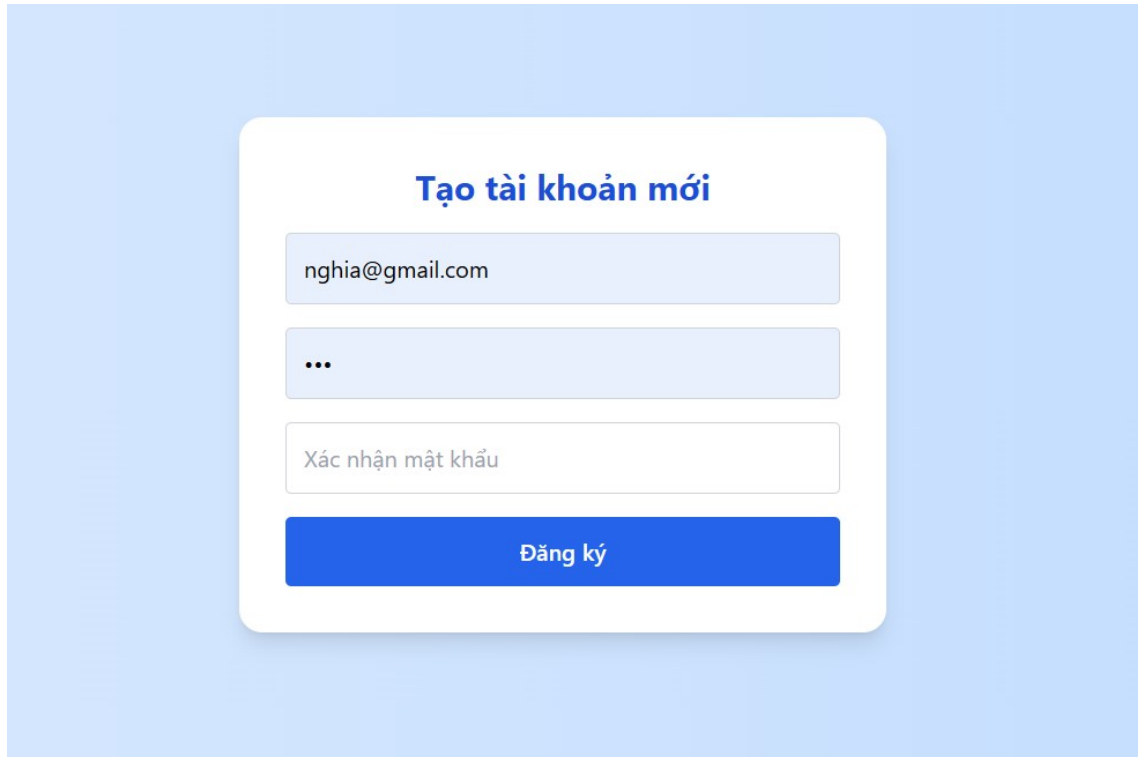


The image shows a login form titled "Đăng nhập vào hệ thống" (Login to the system). It features two input fields: the first contains the email address "nghia@gmail.com", and the second contains three dots "...". To the right of the password field is a link that says "Quên mật khẩu?" (Forgot password?). Below the input fields is a large blue button labeled "Đăng nhập" (Login). The entire form is centered on a light blue background.

Hình 4.4 Giao diện đăng nhập

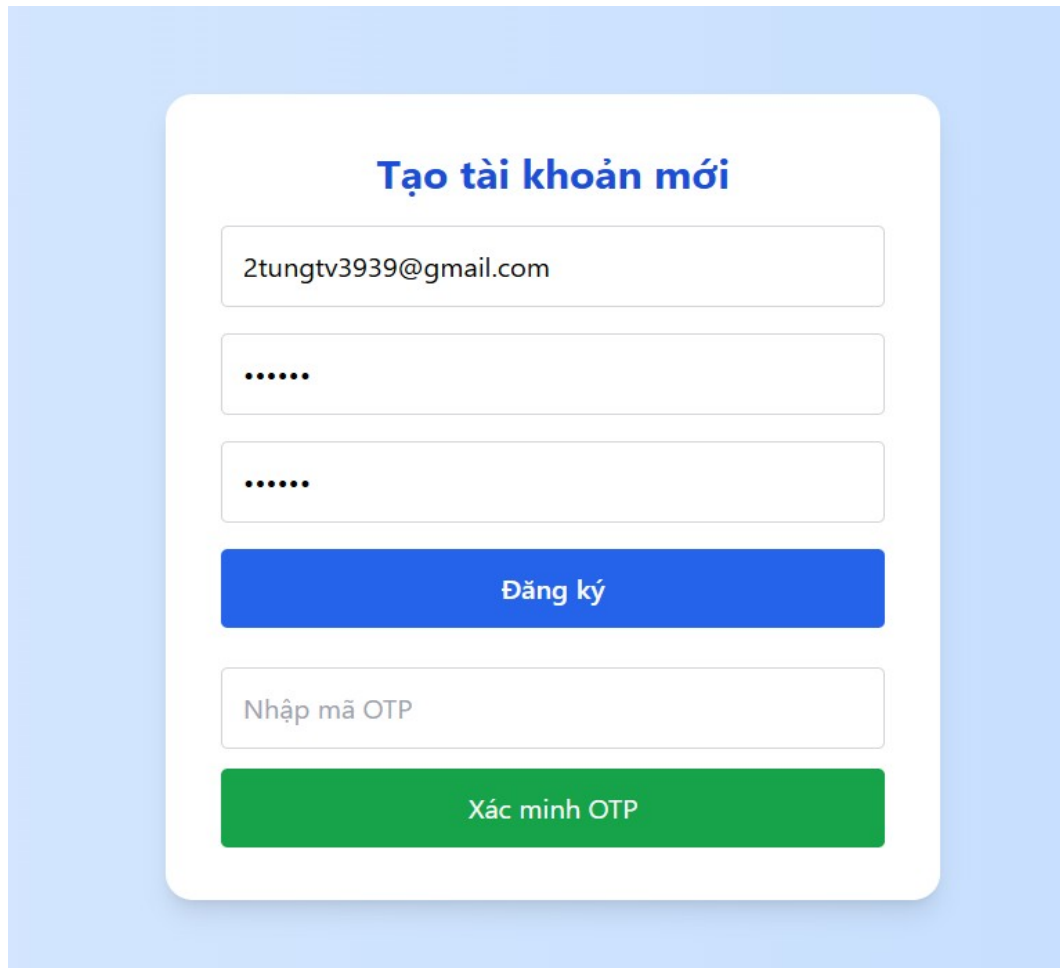
#### 4.4 Giao diện đăng ký

Người dùng tạo tài khoản bằng cách nhập email, nhập và xác nhận lại mật khẩu. Sau khi bấm nút “Đăng ký”, hệ thống sẽ gửi một mã OTP đến email để xác minh. Chỉ sau khi xác thực thành công thì tài khoản mới được kích hoạt và chuyển hướng đến trang đăng nhập.

The image shows a registration form titled "Tạo tài khoản mới" (Create new account) in blue text. The form is set against a light blue background. It contains three input fields: the first contains the email "nghia@gmail.com", the second contains three dots "...", and the third is labeled "Xác nhận mật khẩu" (Confirm password). Below these fields is a prominent blue button with the white text "Đăng ký" (Register).

*Hình 4.5 Giao diện đăng ký*

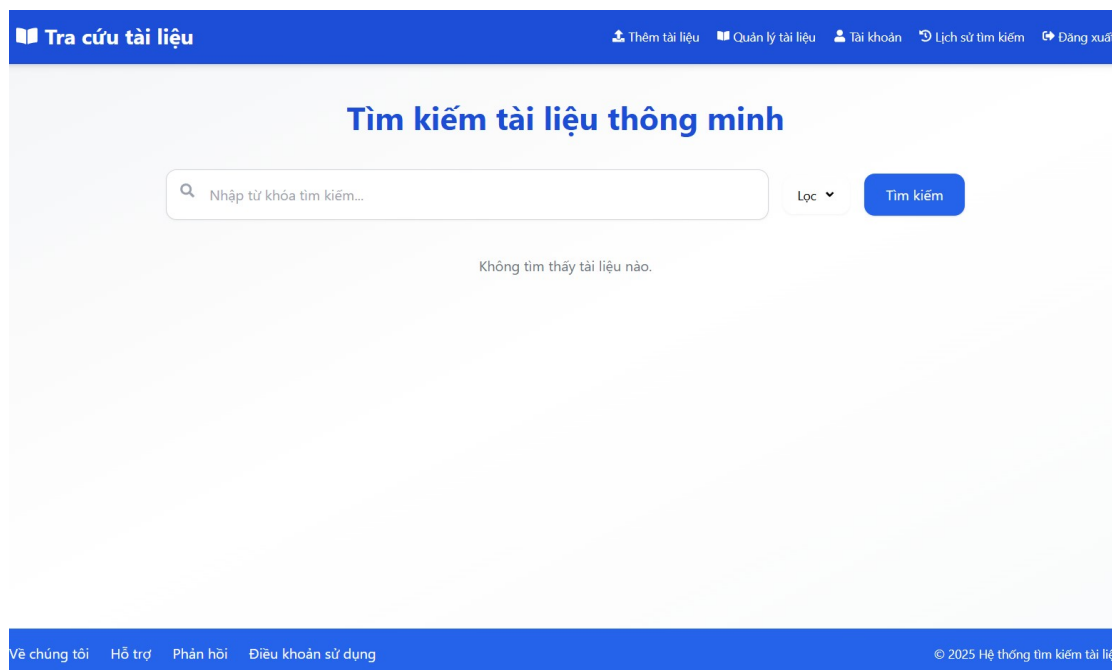
Người dùng tạo tài khoản bằng cách nhập email, nhập và xác nhận lại mật khẩu. Sau khi bấm nút “Đăng ký”, hệ thống sẽ gửi một mã OTP đến email để xác minh. Chỉ sau khi xác thực thành công thì tài khoản mới được kích hoạt và chuyển hướng đến trang đăng nhập.



*Hình 4.6 Giao diện xác thực OTP*

## 4.5 Giao diện trang admin

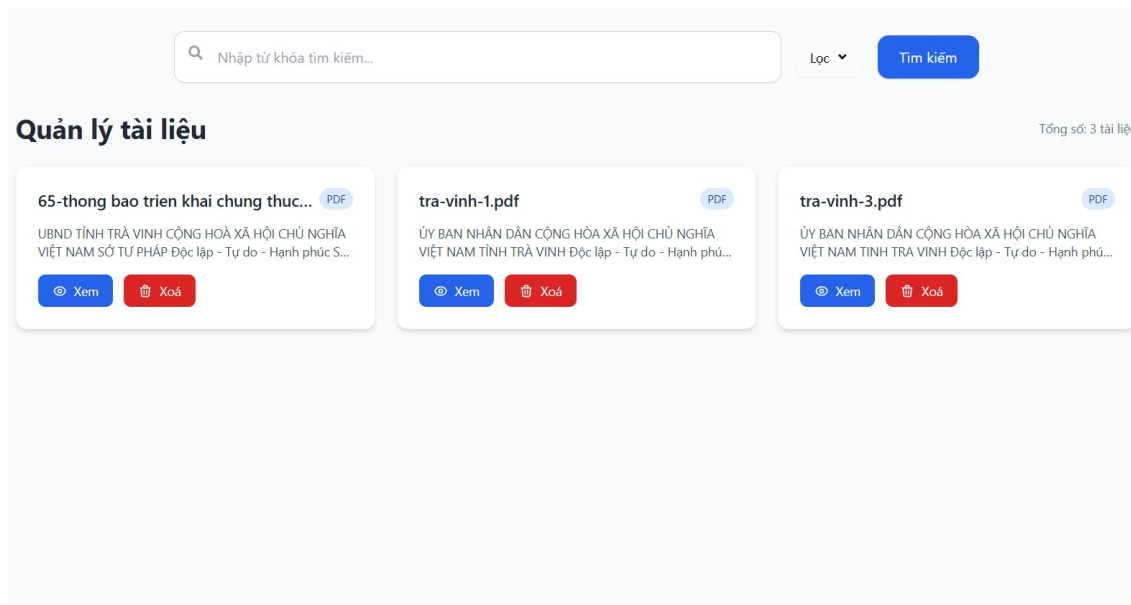
Dành riêng cho người quản trị hệ thống, giao diện này có thanh điều hướng bao gồm các nút: Thêm tài liệu, Quản lý tài liệu, Quản lý người dùng, Lịch sử tìm kiếm và Đăng xuất. Đây là trung tâm điều hành toàn bộ hoạt động và dữ liệu của hệ thống.



Hình 4.7 Giao diện trang admin

## 4.6 Giao diện quản lý tài liệu

Giao diện quản lý tài liệu hiển thị danh sách tất cả các tài liệu trong hệ thống. Mỗi tài liệu đều có hai nút chức năng: “Xem tài liệu” và “Xoá tài liệu”. Ngoài ra, có thanh tìm kiếm để quản trị viên dễ dàng tìm kiếm theo từ khóa.



Hình 4.8 Giao diện quản lý tài liệu


## 4.7 Giao diện quản lý tài khoản người dùng

Trang này hiển thị danh sách tất cả người dùng đã đăng ký trong hệ thống. Quản trị viên có thể tìm kiếm người dùng theo email, lọc theo vai trò (user/admin), đồng thời có thể chỉnh sửa hoặc xoá tài khoản khi cần thiết.


### Quản lý tài khoản người dùng

Tất cả vai trò Lọc Đặt lại

Email: frametv921@gmail.com  
Vai trò: admin  
Đã xác minh: ✓

Xoá  Chỉnh sửa

Email: 2tungtv3939@gmail.com  
Vai trò: user  
Đã xác minh: ✓

Xoá  Chỉnh sửa

Hình 4.9 Giao diện quản lý tài khoản người dùng

### 4.8 Giao diện tải file

Quản trị viên có thể tải lên một hoặc nhiều file PDF thông qua giao diện này. Sau khi upload, hệ thống sẽ xử lý nội dung, sinh vector ngữ nghĩa và lưu trữ vào Elasticsearch để phục vụ chức năng tìm kiếm.

#### Tải lên tài liệu PDF

Bạn có thể chọn một hoặc nhiều tệp PDF để tải lên hệ thống. Sau khi được xử lý, nội dung tài liệu sẽ được phân tích và lưu trữ để phục vụ cho chức năng tìm kiếm thông minh.

Choose Files No file chosen

Tải lên

Hình 4.10 Giao diện tải file

## 4.9 Lịch sử tìm kiếm

Chức năng này cho phép quản trị viên theo dõi toàn bộ lịch sử tìm kiếm của người dùng. Có thể lọc lịch sử theo email, từ khóa, thời gian (từ ngày nào đến ngày nào), và xuất dữ liệu dưới dạng file Excel để phân tích hoặc lưu trữ.

### Lịch sử tìm kiếm người dùng

Lọc theo email

mm/dd/yyyy

mm/dd/yyyy

Lọc

Xuất Excel

Email	Từ khóa	Trường	Thời gian
frametv921@gmail.com	trà vinh	all	15:23:55 16/6/2025
frametv921@gmail.com	trà vinh	all	13:56:21 16/6/2025
frametv921@gmail.com	trà vinh	all	13:56:16 16/6/2025
frametv921@gmail.com	trà vinh	all	13:56:14 16/6/2025
frametv921@gmail.com	trà vinh	all	13:18:00 16/6/2025
frametv921@gmail.com	y tế	all	13:15:09 16/6/2025
frametv921@gmail.com	việt nam	all	14:24:12 14/6/2025

Hình 4.11 Trang lịch sử tìm kiếm



## CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1 Kết luận

Khóa luận “Tìm hiểu công nghệ Elasticsearch và xây dựng hệ thống tìm kiếm tài liệu thông minh” đã đạt được các mục tiêu đề ra. Cụ thể:

Hoàn thành nghiên cứu lý thuyết về Elasticsearch, bao gồm kiến trúc, cơ chế hoạt động, và các tính năng nổi bật như tìm kiếm toàn văn và tìm kiếm ngữ nghĩa.

Xây dựng thành công hệ thống tìm kiếm tài liệu thông minh với các chức năng chính: đăng ký, đăng nhập, tìm kiếm từ khóa và ngữ nghĩa, quản lý tài liệu, quản lý người dùng, và theo dõi lịch sử tìm kiếm.

Tích hợp các công nghệ hiện đại như ReactJS (frontend), Node.js (backend), Python (xử lý PDF và vector ngữ nghĩa), và Elasticsearch (tìm kiếm và lưu trữ dữ liệu).

Hệ thống đáp ứng tốt các yêu cầu chức năng và phi chức năng, đảm bảo hiệu năng, bảo mật, và khả năng mở rộng. Mặc dù đã đạt được nhiều kết quả tích cực, hệ thống vẫn còn một số hạn chế như khả năng xử lý tài liệu PDF phức tạp (bảng, hình ảnh) hoặc tối ưu hóa hiệu suất khi khối lượng dữ liệu lớn. Những hạn chế này sẽ được xem xét trong các hướng phát triển tiếp theo.

### 5.2 Hướng phát triển

Để nâng cao chất lượng và mở rộng ứng dụng của hệ thống, các hướng phát triển trong tương lai bao gồm:

Mở rộng định dạng tài liệu: Hỗ trợ các định dạng tài liệu khác như DOCX, TXT để đa dạng hóa kho dữ liệu.

Tối ưu hóa hiệu suất: Tăng cường cấu hình Elasticsearch (shard, replica) và sử dụng caching để cải thiện tốc độ tìm kiếm với khối lượng dữ liệu lớn.

Tích hợp AI nâng cao: Áp dụng các mô hình NLP tiên tiến hơn (như BERT, GPT) để cải thiện tìm kiếm ngữ nghĩa và hỗ trợ trả lời câu hỏi tự nhiên từ tài liệu.

## DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Amazon Web Service, "Javascript (JS) là gì?," Amazon Web Service, 3 2006. [Online]. Available: <https://aws.amazon.com/vi/what-is/javascript/>. [Accessed 2025].
- [2] Top Dev, "MongoDB là gì? Định nghĩa đầy đủ và chi tiết nhất về MongoDB," Top Dev, 2015. [Online]. Available: <https://topdev.vn/blog/mongodb-la-gi/>. [Accessed 2025].
- [3] Top Dev, "Node.js là gì? Tổng hợp kiến thức NodeJS," Top Dev, 2015. [Online]. Available: <https://topdev.vn/blog/node-js-la-gi/>. [Accessed 2025].
- [4] Top Dev, "RESTful API là gì? Cách thiết kế RESTful API," Top Dev, 2015. [Online]. Available: <https://topdev.vn/blog/restful-api-la-gi/>. [Accessed 2025].
- [5] P. M. Khoa, "React là gì? Lộ trình trở thành lập trình viên ReactJS," Top Dev, 10 1 2015. [Online]. Available: <https://topdev.vn/blog/react-la-gi-lo-trinh/>. [Accessed 2025].
- [6] N. Liên, "Expressjs là gì? Cách sử dụng Expressjs trong lập trình đạt kết quả tốt nhất," FPT Shop, 8 3 2012. [Online]. Available: <https://fptshop.com.vn/tin-tuc/danh-gia/express-js-la-gi-174976>. [Accessed 2025].
- [7] Elastic, "Elasticsearch: The official distributed search & analytics engine," Elastic, 2023. [Online]. Available: <https://www.elastic.co/elasticsearch/>. [Accessed 2025].
- [8] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks.," in *2019 Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*., Hong Kong, China, 2019.
- [9] MongoDB, "MongoDB Features," MongoDB, 11 2 2009. [Online]. Available: <https://www.mongodb.com/resources/products/fundamentals/features>. [Accessed 2025].
- [10] PyTorch Team, "PyTorch: An open source machine learning framework," Facebook AI Research, 2023. [Online]. Available: <https://pytorch.org/>. [Accessed 2025].
- [11] S. Kumar, "python-dotenv: Read key-value pairs from .env file," 2023. [Online]. Available: <https://github.com/theskumar/python-dotenv>. [Accessed 2025].
- [12] Google, "Tesseract OCR," Google, 2023. [Online]. Available: <https://github.com/tesseract-ocr/tesseract>. [Accessed 2025].
- [13] G. v. Rossum, "Python programming language.," Python Software Foundation, 2023. [Online]. Available: <https://www.python.org/>. [Accessed 2025].

## PHỤ LỤC