

TRƯỜNG ĐẠI HỌC TRÀ VINH  
KHOA KỸ THUẬT VÀ CÔNG NGHỆ  
BỘ MÔN CÔNG NGHỆ THÔNG TIN



**BÁO CÁO**

**THỰC TẬP TỐT NGHIỆP**

Cán bộ hướng dẫn tại nơi thực tập:

**Họ và tên: Nhan Minh Phúc**

Giáo viên hướng dẫn:

**Họ và tên: Nguyễn Hoàng Duy Thiện**

Sinh viên báo cáo:

**Họ và tên: Nguyễn Đại Hoàng Phúc**

Mã số SV: 110121087

Lớp: DA21TTB

Trà Vinh, tháng 3 năm 2025

## LỜI CẢM ƠN

Trước tiên, em xin bày tỏ lòng biết ơn sâu sắc tới nhà trường đã tạo điều kiện thuận lợi để em có cơ hội trải nghiệm thực tập trước khi bước chân vào cuộc sống thực tế.

Em xin gửi lời cảm ơn chân thành đến Trung tâm Ngoại ngữ - Tin học Victory, nơi đã tạo ra môi trường thực tập chuyên nghiệp, giúp em mở rộng kiến thức và rèn luyện kỹ năng chuyên môn qua những trải nghiệm thực tế quý báu.

Em xin gửi lời tri ân sâu sắc đến thầy Nhan Minh Phúc – cán bộ hướng dẫn tại Trung tâm, người đã luôn tận tâm chỉ bảo, hướng dẫn và hỗ trợ em trong suốt quá trình thực tập. Sự nhiệt huyết và chuyên nghiệp của thầy chính là nguồn động lực giúp em tự tin vượt qua thử thách và hoàn thành tốt các nhiệm vụ được giao.

Em cũng không quên gửi lời tri ân sâu sắc tới thầy Nguyễn Hoàng Duy Thiện – giáo viên hướng dẫn của em, người đã tận tâm dìu dắt, chia sẻ kinh nghiệm và kiến thức bổ ích, góp phần định hướng cho quá trình học tập và thực tập của em. Sự tin tưởng và khích lệ của thầy đã truyền cho em sức mạnh để không ngừng nỗ lực và phấn đấu.

Em sẽ tiếp tục cố gắng học tập, rèn luyện để không phụ lòng mong đợi và sự tin yêu của quý thầy cô. Một lần nữa, em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến tất cả mọi người đã đồng hành và hỗ trợ em trong suốt hành trình thực tập này.

Nguyễn Đại Hoàng Phúc

## MỤC LỤC

NHẬT KÍ THỰC TẬP TỐT NGHIỆP.....	1
PHẦN 1 GIỚI THIỆU VỀ CƠ QUAN NƠI THỰC TẬP TỐT NGHIỆP .....	3
1.1 Giới thiệu chung: .....	3
1.2 Các lĩnh vực hoạt động chính.....	3
1.2.1 Đào tạo và Bồi dưỡng.....	3
1.2.2 Khảo thí và Đánh giá Năng lực .....	3
1.2.3 Cung cấp Dịch vụ Giáo dục và Tư vấn .....	3
1.2.4 Hợp tác Quốc tế.....	3
1.2.5 Nghiên cứu và Phát triển .....	4
1.3 Thế mạnh của Trung tâm:.....	4
1.4 Sơ đồ tổ chức và cơ cấu hoạt động.....	4
PHẦN 2 NỘI DUNG THỰC TẬP .....	6
2.1 Giới thiệu nhiệm vụ và mục tiêu dự án: .....	6
2.2 Phân tích yêu cầu hệ thống:.....	7
2.2.1. Yêu cầu chức năng: .....	7
2.2.2 Yêu cầu phi chức năng: .....	8
2.3 Nghiên cứu lý thuyết và lựa chọn công nghệ để ứng dụng: .....	8
2.3.1 Tổng quan các giải pháp:.....	8
2.3.2 Nghiên cứu về RAG (Retrieval Augmented Generation): .....	9
2.3.3 Công nghệ và thư viện được sử dụng:.....	11
2.4 Thiết kế hệ thống: .....	12
2.4.1 Kiến trúc tổng thể: .....	12
2.4.2 Thiết kế cơ sở dữ liệu (Supabase): .....	14
2.4.3 Thiết kế luồng xử lý RAG: .....	15
2.4.4 Thiết kế giao diện người dùng:.....	17
2.5 Các kĩ thuật được áp dụng trong hệ thống: .....	24
2.5.1 Xử lý và Nạp Tài liệu (Document Ingestion and Processing):.....	24
2.5.2 Lập Chỉ mục (Indexing Strategies) .....	25
2.5.3 Xử lý Truy vấn và Truy xuất (Query Processing and Retrieval) .....	26
2.5.4 Sắp xếp lại và Tối ưu hóa Ngữ cảnh: .....	26
2.5.5 Sinh Câu trả lời và Hậu xử lý:.....	27

2.5.6 Tối ưu hóa cho Tiếng Việt: .....	28
2.5.7 Kiến trúc và Quản lý Hệ thống:.....	28
2.6 Kết quả thực tế của hệ thống: .....	29
PHẦN 3 KẾT LUẬN .....	32
3.1 Tự nhận xét, đánh giá .....	32
3.2 Kết luận, kiến nghị .....	34

## DANH MỤC HÌNH ẢNH

Hình 1: Sơ đồ tổ chức của Trung tâm Tin học và Ngoại ngữ Victory .....	5
Hình 2: Hình ảnh minh họa quá trình hoạt động của RAG .....	10
Hình 3: Kiến trúc tổng thể hệ thống RAG hiện tại.....	13
Hình 4: Lược đồ cơ sở dữ liệu Supabase.....	15
Hình 5: Sơ đồ minh hoạt hệ thống Vertex AI RAG Engine của Google .....	16
Hình 6: Thiết kế giao diện Login (Đăng nhập hệ thống) .....	17
Hình 7: Thiết kế giao diện Register (Đăng ký tài khoản) .....	18
Hình 8: Thiết kế giao diện quên mật khẩu .....	18
Hình 9: Thiết kế giao diện chính của hệ thống và quản lý tài liệu .....	19
Hình 10: Thiết kế giao diện chính của hệ thống và quản lý lịch sử chat .....	19
Hình 11: Trang hiển thị thông tin cá nhân cơ bản và đổi mật khẩu .....	20
Hình 12: Giao diện mobile trang đăng nhập .....	20
Hình 13: Giao diện mobile trang đăng ký tài khoản .....	21
Hình 14: Giao diện mobile trang khôi phục mật khẩu .....	21
Hình 15: Giao diện mobile trang Chat chính .....	22
Hình 16: Giao diện mobile trang quản lý tài liệu .....	22
Hình 17: Giao diện mobile trang quản lý lịch sử chat.....	23
Hình 18: Giao diện mobile trang quản lý hồ sơ cá nhân .....	23
Hình 19: Kiểm tra độ chính xác của hệ thống RAG .....	30
Hình 20: Đối chiếu với tài liệu thật .....	30
Hình 21: Khả năng xử lý lỗi khi không trả lời được câu hỏi.....	31
Hình 22: Kết quả của khả năng xử lý lỗi khi không trả lời được câu hỏi .....	31

## NHẬT KÍ THỰC TẬP TỐT NGHIỆP

TT	NGÀY THÁNG	NỘI DUNG, CÔNG VIỆC THỰC HIỆN	GHI CHÚ
Tuần 1	Từ 17/02/2025 Đến 23/02/2025	<ul style="list-style-type: none"> <li>- Tìm hiểu tổng thể các công nghệ cơ bản: HTML, CSS, JavaScript cho frontend; Python cùng các framework Flask, Django cho backend.</li> <li>- Nắm bắt khái quát về thư viện AI (TensorFlow, PyTorch) và tìm hiểu quy trình tích hợp cơ chế RAG ngay từ ban đầu.</li> <li>- Khảo sát các dự án mã nguồn mở có tính năng upload file và tích hợp AI.</li> </ul>	Xây dựng nền tảng kiến thức ban đầu cho dự án.
Tuần 2	Từ 24/02/2025 Đến 02/03/2025	<ul style="list-style-type: none"> <li>- Nghiên cứu sâu các dự án đã triển khai, tổng hợp phương pháp và kinh nghiệm từ các dự án mở.</li> <li>- Xem video hướng dẫn về xây dựng giao diện upload file, xử lý dữ liệu backend và tích hợp mô hình AI.</li> </ul>	Nghiên cứu chuyên sâu các dự án đã có từ trước.
Tuần 3	Từ 03/03/2025 Đến 09/03/2025	<ul style="list-style-type: none"> <li>- Hoàn thành phiên bản tạm của hệ thống RAG với các chức năng: upload file, trích xuất nội dung, truy xuất thông tin và tạo phản hồi dựa trên dữ liệu tài liệu.</li> <li>- Thử nghiệm ban đầu hệ thống hoạt động ổn định, nhận diện các hạn chế về hiệu suất và độ chính xác của phản hồi.</li> </ul>	Ghi nhận những điểm cần cải thiện để tối ưu hệ thống.

Tuần 4	Từ 10/03/2025 Đến 16/03/2025	<ul style="list-style-type: none"> <li>- Bổ sung thêm các tính năng nhỏ nhằm nâng cao trải nghiệm người dùng: gợi ý các câu hỏi tương tự khi không tìm được đáp án, cải thiện tốc độ truy xuất dữ liệu.</li> <li>- Kiểm thử toàn bộ hệ thống và fix lỗi phát sinh trong quá trình chạy thử.</li> </ul>	Tăng cường tính ổn định và cải thiện hiệu suất hệ thống.
Tuần 5	Từ 17/03/2025 Đến 23/03/2025	<ul style="list-style-type: none"> <li>- Cải tiến giao diện người dùng, đảm bảo tính trực quan và responsive cho thiết bị di động.</li> <li>- Tiếp tục kiểm thử, fix lỗi và cập nhật tài liệu hướng dẫn sử dụng.</li> <li>- Thử tích hợp chức năng quản lý người dùng để phân chia dữ liệu riêng biệt cho từng cá nhân.</li> </ul>	Nâng cao trải nghiệm người dùng và chuẩn bị cho giai đoạn triển khai đầy đủ.
Tuần 6	Từ 24/03/2025 Đến 30/03/2025	Hoàn thiện toàn bộ hệ thống RAG, đảm bảo các chức năng upload, xử lý tài liệu và trả về phản hồi hoạt động ổn định và hiệu quả.	Kết thúc quá trình phát triển hệ thống với đầy đủ chức năng theo yêu cầu dự án.

## **PHẦN 1 GIỚI THIỆU VỀ CƠ QUAN NƠI THỰC TẬP TỐT NGHIỆP**

### **1.1 Giới thiệu chung:**

Trung tâm Ngoại ngữ – Tin học Victory thuộc Trường Đại học Trà Vinh được thành lập từ năm 2006. Với sứ mệnh cung cấp các chương trình đào tạo đa dạng, trung tâm đã phục vụ nhu cầu học tập của nhiều đối tượng như cán bộ công chức, viên chức, sinh viên, học sinh trong và ngoài tỉnh. Đặc biệt, từ năm 2014, trung tâm vinh dự trở thành Trung tâm Ủy quyền của Bộ phận Đánh giá Ngôn ngữ Cambridge (mã số VN370), mở ra cơ hội cho người học tại địa phương tiếp cận với các kỳ thi quốc tế, đánh giá năng lực ngôn ngữ ở tầm cao hơn.

### **1.2 Các lĩnh vực hoạt động chính**

#### **1.2.1 Đào tạo và Bồi dưỡng**

- Giảng dạy ngoại ngữ (Tiếng Anh, Hoa, Nhật).
- Đào tạo tiếng dân tộc (Tiếng Khmer).
- Đào tạo, bồi dưỡng và ứng dụng Công nghệ Thông tin (từ cơ bản đến nâng cao, tin học thiếu nhi, lớp Premium).
- Các khóa học bồi dưỡng ngắn hạn (chuyên môn, phương pháp giảng dạy, kỹ năng mềm, kỹ năng sống).
- Đào tạo tiếng Việt cho người nước ngoài.

#### **1.2.2 Khảo thí và Đánh giá Năng lực**

- Tổ chức thi chứng chỉ quốc tế (Cambridge, IELTS).
- Tổ chức thi đánh giá năng lực ngoại ngữ theo Khung NLNN 6 bậc.
- Thi, đánh giá và cấp chứng chỉ tiếng dân tộc Khmer.
- Liên kết tổ chức các kỳ thi theo tiêu chuẩn quốc tế.

#### **1.2.3 Cung cấp Dịch vụ Giáo dục và Tư vấn**

- Tư vấn khóa học, chương trình đào tạo.
- Dịch vụ liên quan (dịch thuật, hợp pháp lãnh sự, tư vấn du học).
- Thiết kế chương trình đào tạo, cung ứng dịch vụ lập trình, cài đặt phần mềm theo yêu cầu.
- Tổ chức hoạt động ngoại khóa, chuyến trải nghiệm.

#### **1.2.4 Hợp tác Quốc tế**

- Phát triển quan hệ hợp tác đào tạo và trao đổi học thuật.
- Tổ chức giao lưu, tập huấn, trao đổi giảng viên.



### **1.2.5 Nghiên cứu và Phát triển**

- Nghiên cứu xu hướng dạy và học, ứng dụng khoa học công nghệ vào quản lý và giảng dạy.
- Xây dựng và cập nhật chương trình đào tạo.

### **1.3 Thế mạnh của Trung tâm:**

- Đội ngũ giảng viên:
  - + Giáo viên giỏi, trẻ trung, năng động và có phương pháp giảng dạy sáng tạo.
  - + Sự có mặt của nguồn giáo viên nước ngoài giúp tạo môi trường giao tiếp, cải thiện phát âm và nâng cao kỹ năng ngoại ngữ cho học viên.
- Chương trình học đa dạng và cập nhật: Các chương trình được xây dựng linh hoạt, cập nhật thường xuyên nhằm đáp ứng kịp thời nhu cầu học tập của từng đối tượng.
- Đánh giá năng lực và cấp chứng chỉ: Trung tâm là đơn vị khảo thí ủy quyền duy nhất tại Trà Vinh kể từ năm 2014, tổ chức đánh giá năng lực và cấp chứng chỉ theo tiêu chuẩn quốc tế.
- Dịch vụ hỗ trợ chất lượng: Cung cấp dịch vụ nhanh chóng, tận tình và hiệu quả, luôn hướng đến việc xây dựng mối quan hệ bền vững với học viên và khách hàng.

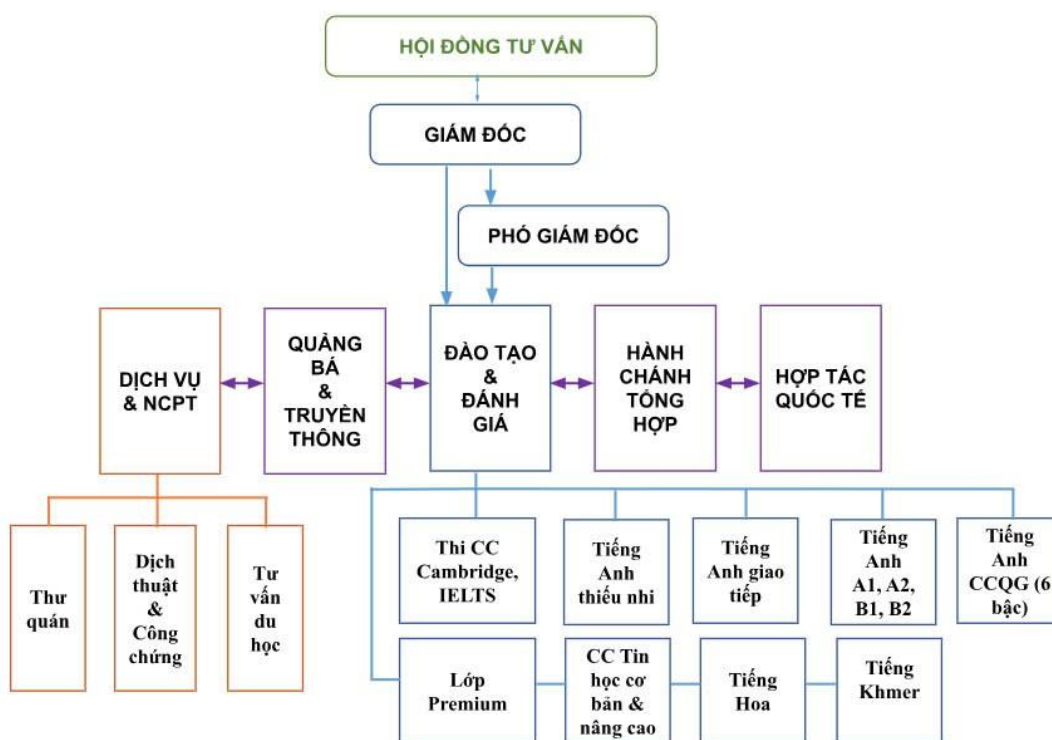
### **1.4 Sơ đồ tổ chức và cơ cấu hoạt động**

- Cấp lãnh đạo cao nhất: Bao gồm Hội đồng Tư vấn đóng vai trò cố vấn chiến lược, Giám đốc chịu trách nhiệm điều hành tổng thể và Phó Giám đốc hỗ trợ quản lý, trực tiếp chỉ đạo các khối chức năng.
- Các khối chức năng chính: Dưới sự điều hành của Ban Giám đốc, Trung tâm được chia thành các khối công việc chính, có sự phối hợp chặt chẽ với nhau:
  - + Khối Dịch vụ & NCPT (Nghiên cứu Phát triển): Cung cấp các dịch vụ hỗ trợ và phát triển liên quan.
  - + Khối Quảng bá & Truyền thông: Đảm nhận công tác marketing, xây dựng thương hiệu và thu hút học viên.
  - + Khối Đào tạo & Đánh giá: Là trung tâm của hoạt động nghiệp vụ, chịu trách nhiệm xây dựng, triển khai chương trình đào tạo và tổ chức các hoạt động kiểm tra, đánh giá năng lực.

- + Khối Hành chánh Tổng hợp: Phụ trách công tác hành chính, nhân sự, tài chính và hậu cần.
- + Khối Hợp tác Quốc tế: Chuyên trách việc phát triển và duy trì các mối quan hệ hợp tác với đối tác nước ngoài.

- **Các bộ phận chuyên môn và dịch vụ trực thuộc:**

- + Trực thuộc Khối Dịch vụ & NCPT: Gồm có Thư quán, bộ phận Dịch thuật & Công chứng, và bộ phận Tư vấn du học.
- + Trực thuộc Khối Đào tạo & Đánh giá: Phụ trách tổ chức các hoạt động đào tạo và khảo thí đa dạng, bao gồm:
  - Tổ chức thi chứng chỉ Cambridge, IELTS.
  - Các Lớp Premium theo yêu cầu.
  - Các lớp Tiếng Anh thiếu nhi, Tiếng Anh giao tiếp.
  - Đào tạo Chứng chỉ Tin học cơ bản & nâng cao.
  - Các lớp Tiếng Hoa, Tiếng Khmer.
  - Đào tạo và thi Tiếng Anh theo khung năng lực 6 bậc (A1-B2, CCQG).



Hình 1: Sơ đồ tổ chức của Trung tâm Tin học và Ngoại ngữ Victory

## PHẦN 2 NỘI DUNG THỰC TẬP

### 2.1 Giới thiệu nhiệm vụ và mục tiêu dự án:

Trong suốt quá trình thực tập tốt nghiệp kéo dài 6 tuần, dưới sự hướng dẫn trực tiếp của thầy Nhan Minh Phúc - Cán bộ hướng dẫn tại Trung tâm Ngoại ngữ - Tin học Victory, em đã có cơ hội được áp dụng kiến thức đã học vào một dự án thực tế, đồng thời học hỏi thêm nhiều kỹ năng và kinh nghiệm quý báu. Nội dung thực tập của em tập trung vào nghiên cứu và phát triển một hệ thống ứng dụng Trí tuệ nhân tạo (AI) với nhiệm vụ chính là “Xây dựng hệ thống Chatbot truy vấn tài liệu tích hợp công nghệ RAG”.

Báo cáo này trình bày chi tiết quá trình thực hiện dự án trong suốt 6 tuần, bao gồm các nhiệm vụ cụ thể được giao, công nghệ và phương pháp nghiên cứu cũng như áp dụng, các thách thức gặp phải cùng giải pháp khắc phục, và những kiến thức, kỹ năng quý báu đã tích lũy được. Trọng tâm của dự án là xây dựng một hệ thống RAG (Retrieval-Augmented Generation) hiệu quả, tích hợp với nền tảng Supabase để quản lý người dùng và dữ liệu, nhằm mang lại trải nghiệm hỏi đáp thông minh và cá nhân hóa cho người dùng.

Mục tiêu cụ thể của dự án bao gồm:

1. Xây dựng giao diện web thân thiện: Cung cấp giao diện trực quan, dễ sử dụng cho việc tải lên tài liệu, quản lý tài liệu, và tương tác hỏi đáp. Giao diện cần đảm bảo tính đáp ứng (responsive) trên các thiết bị khác nhau.
2. Xử lý và lưu trữ tài liệu hiệu quả: Hệ thống phải có khả năng trích xuất nội dung văn bản từ các định dạng file khác nhau, xử lý và lưu trữ nội dung này một cách có cấu trúc để phục vụ cho việc truy vấn.
3. Triển khai cơ chế hỏi đáp thông minh: Áp dụng các kỹ thuật AI tiên tiến để hiểu câu hỏi của người dùng và tìm kiếm, tổng hợp thông tin từ các tài liệu đã tải lên để đưa ra câu trả lời chính xác và phù hợp. Đặc biệt, câu trả lời phải bám sát nội dung tài liệu gốc và có khả năng chỉ dẫn nguồn tham khảo.
4. Quản lý người dùng và dữ liệu: Tích hợp hệ thống xác thực người dùng, đảm bảo mỗi người dùng chỉ có thể truy cập và hỏi đáp trên tài liệu của chính mình. Lưu trữ lịch sử trò chuyện để người dùng có thể xem lại.
5. Tối ưu hóa và đảm bảo chất lượng: Hệ thống cần hoạt động ổn định, có hiệu suất chấp nhận được và cung cấp câu trả lời với độ chính xác cao.

## **2.2 Phân tích yêu cầu hệ thống:**

### **2.2.1. Yêu cầu chức năng:**

- Quản lý Người dùng:
  - + Đăng ký tài khoản mới bằng email và mật khẩu.
  - + Đăng nhập vào hệ thống bằng email/mật khẩu hoặc tài khoản Google (OAuth).
  - + Đăng xuất khỏi hệ thống.
  - + Người dùng có thể thay đổi mật khẩu sau khi đăng nhập.
  - + Người dùng có thể yêu cầu đặt lại mật khẩu qua email nếu quên.
- Quản lý Tài liệu:
  - + Người dùng có thể tải lên một hoặc nhiều tệp tài liệu cùng lúc (hỗ trợ PDF, DOCX, TXT).
  - + Hệ thống hiển thị danh sách các tài liệu đã tải lên của người dùng đang đăng nhập.
  - + Người dùng có thể xóa các tài liệu đã tải lên.
  - + Hệ thống xử lý (trích xuất text, chunking, embedding, indexing) tài liệu sau khi tải lên.
- Hỏi đáp Tài liệu (RAG):
  - + Người dùng có thể nhập câu hỏi vào giao diện chat.
  - + Hệ thống tiếp nhận câu hỏi, thực hiện truy xuất thông tin liên quan chỉ từ các tài liệu của người dùng đó.
  - + Hệ thống sử dụng LLM (Gemini) để tạo câu trả lời dựa trên thông tin truy xuất và câu hỏi.
  - + Hệ thống hiển thị câu trả lời cho người dùng trong giao diện chat.
  - + Câu trả lời phải kèm theo thông tin nguồn (tên tài liệu, số trang ước tính).
  - + Nếu không tìm thấy thông tin liên quan, hệ thống trả lời "Không có thông tin liên quan trong tài liệu." và gợi ý các câu hỏi tương tự.
- Quản lý Lịch sử Chat:
  - + Hệ thống tự động lưu lại các cặp câu hỏi và câu trả lời vào lịch sử chat của người dùng.
  - + Người dùng có thể xem lại danh sách các cuộc trò chuyện trước đó.
  - + Người dùng có thể chọn một cuộc trò chuyện cũ để xem lại nội dung.

- + Người dùng có thể tạo một cuộc trò chuyện mới.
- + Người dùng có thể xóa các cuộc trò chuyện cũ.

### **2.2.2 Yêu cầu phi chức năng:**

- Hiệu năng:
  - + Thời gian phản hồi cho một câu hỏi nên ở mức chấp nhận được (ví dụ: dưới 10-15 giây cho các truy vấn thông thường).
  - + Thời gian xử lý file upload (embedding, indexing) không nên quá lâu, đặc biệt với các file kích thước trung bình.
- Độ chính xác: Câu trả lời phải chính xác, bám sát nội dung tài liệu và giảm thiểu tối đa thông tin sai lệch.
- Tính khả dụng: Hệ thống web phải truy cập được thông qua trình duyệt web phổ biến. Giao diện phải hoạt động ổn định.
- Tính bảo mật:
  - + Dữ liệu người dùng (thông tin tài khoản, tài liệu, lịch sử chat) phải được bảo mật.
  - + Người dùng chỉ có thể truy cập dữ liệu của chính mình.
  - + Mật khẩu người dùng phải được lưu trữ an toàn (Supabase Auth xử lý).
- Tính dễ sử dụng (Usability): Giao diện phải trực quan, dễ hiểu, dễ thao tác cho người dùng không chuyên về kỹ thuật.
- Tính đáp ứng (Responsiveness): Giao diện phải hiển thị tốt trên các kích thước màn hình khác nhau (desktop, tablet, mobile).
- Khả năng bảo trì và mở rộng: Mã nguồn cần được tổ chức tốt, module hóa để dễ dàng sửa lỗi, nâng cấp và thêm tính năng mới.

## **2.3 Nghiên cứu lý thuyết và lựa chọn công nghệ để ứng dụng:**

### **2.3.1 Tổng quan các giải pháp:**

Trước khi đi sâu vào RAG, cần xem xét các phương pháp xây dựng hệ thống Hỏi đáp (Question Answering - QA) phổ biến khác để hiểu rõ hơn về ưu thế của giải pháp được lựa chọn:

- QA dựa trên Truy xuất Thông tin (IR-based QA): Các hệ thống này thường tìm kiếm các đoạn văn bản chứa từ khóa trong câu hỏi và trả về đoạn văn bản đó, ít có khả năng tổng hợp hay diễn giải sâu.

- QA dựa trên Cơ sở Tri thức (KB-QA): Yêu cầu dữ liệu phải được cấu trúc hóa dưới dạng đồ thị tri thức (Knowledge Base). Việc xây dựng và bảo trì KB tốn nhiều công sức và khó áp dụng cho các tài liệu văn bản phi cấu trúc.
- QA dựa trên Huấn luyện tinh chỉnh LLM (Fine-tuning based QA): Phương pháp này huấn luyện lại một phần hoặc toàn bộ LLM trên bộ dữ liệu hỏi-đáp cụ thể. Mặc dù có thể cho kết quả tốt, nhưng đòi hỏi tài nguyên tính toán lớn, tốn thời gian và khó cập nhật kiến thức mới khi tài liệu thay đổi thường xuyên. LLM thuần túy cũng dễ gặp hiện tượng "ảo giác" (hallucination) - tự bịa thông tin.

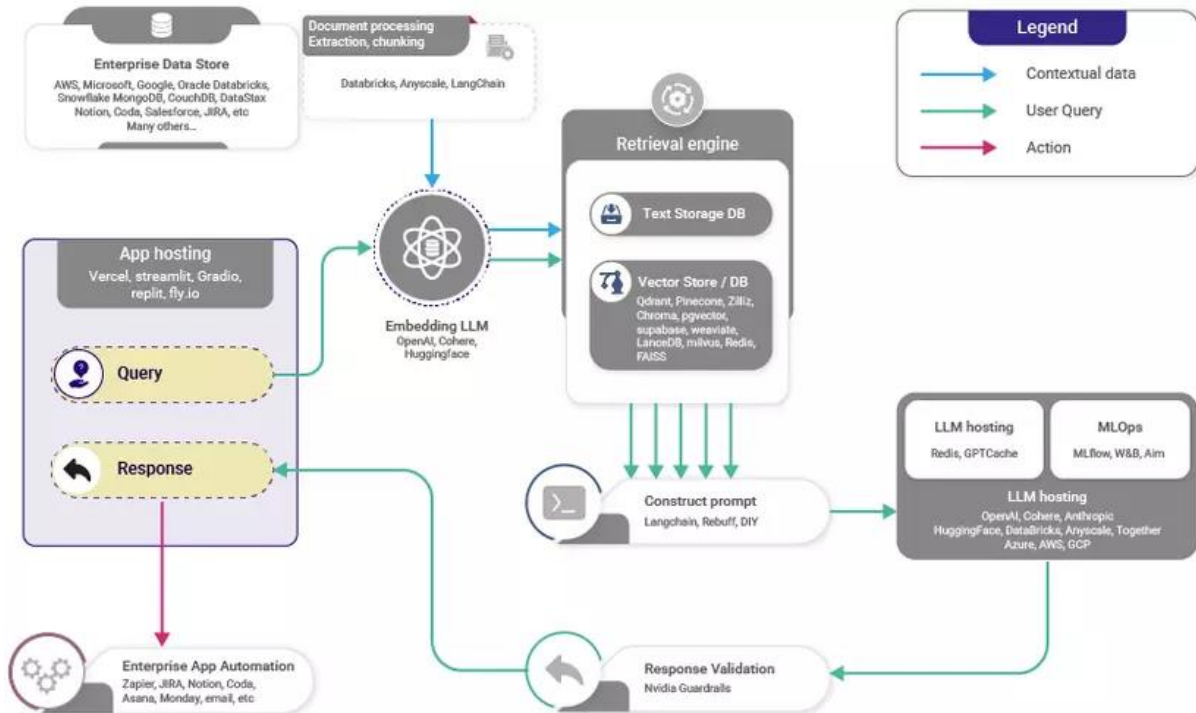
So với các phương pháp trên, Retrieval-Augmented Generation (RAG) nổi lên như một giải pháp cân bằng và hiệu quả cho bài toán hỏi đáp dựa trên tài liệu. RAG khắc phục hạn chế "ảo giác" của LLM bằng cách buộc mô hình phải dựa vào thông tin được truy xuất từ tài liệu gốc. Việc cập nhật kiến thức đơn giản chỉ là cập nhật kho tài liệu và chỉ mục mà không cần fine-tuning LLM tốn kém. Khả năng trích dẫn nguồn giúp tăng độ tin cậy. Do đó, RAG là lựa chọn phù hợp với yêu cầu về tính chính xác, linh hoạt và khả năng cập nhật của dự án này.

### 2.3.2 Nghiên cứu về RAG (Retrieval Augmented Generation):

- Retrieval-Augmented Generation (RAG) là một kỹ thuật tiên tiến trong lĩnh vực trí tuệ nhân tạo (AI), kết hợp giữa khả năng truy xuất thông tin và mô hình ngôn ngữ lớn (LLM) để tạo ra nội dung chính xác và phù hợp hơn. Phương pháp này giúp các mô hình AI không chỉ dựa vào dữ liệu đã được huấn luyện mà còn có thể truy cập và sử dụng thông tin từ các nguồn bên ngoài, nâng cao độ tin cậy và tính cập nhật của kết quả.
- Tóm tắt ngắn gọn quá trình của RAG như sau:
  1. **Create Vector database:** Đầu tiên, convert toàn bộ dữ liệu tri thức thành các vector và lưu trữ chúng vào một vector database.
  2. **User input:** User cung cấp 1 câu truy vấn (query) bằng ngôn ngữ tự nhiên nhằm tìm kiếm câu trả lời hoặc để hoàn thành câu truy vấn đó.
  3. **Information retrieval:** Cơ chế retrieval quét toàn bộ vector trong database để xác định các phân đoạn tri thức (chính là paragraphs) nào có ngữ nghĩa tương đồng với câu truy vấn của người dùng. Các paragraphs

này sau đó được vào LLM để làm tăng context cho quá trình sinh ra câu trả lời.

4. **Combining data:** Các paragraphs được lấy sau quá trình retrieval từ database được kết hợp với câu query ban đầu của user tạo thành 1 câu prompt.
5. **Generate text:** Câu prompt được bổ sung thêm context sau đó được đưa qua LLM để sinh ra câu phản hồi cuối cùng theo context bổ sung.



Hình 2: Hình ảnh minh họa quá trình hoạt động của RAG

#### - Ưu điểm của RAG

- + Tận dụng nguồn dữ liệu bên ngoài: RAG cho phép các mô hình LLM sử dụng thông tin từ kho dữ liệu bên ngoài, giúp bổ sung tri thức và cung cấp các câu trả lời chính xác, phong phú cho người dùng.
- + Không cần huấn luyện lại mô hình: Do không đòi hỏi việc tái huấn luyện toàn bộ mô hình, RAG giúp tiết kiệm đáng kể thời gian và tài nguyên tính toán, đặc biệt hữu ích khi cần triển khai nhanh chóng.
- + Hiệu quả với dữ liệu hạn chế được gán nhãn: Trong nhiều tình huống chỉ có sẵn dữ liệu unlabeled với số lượng lớn, RAG vẫn có thể hoạt động tốt bằng cách truy xuất và sử dụng thông tin hữu ích từ nguồn dữ liệu có cấu trúc tốt.

+ Phù hợp cho ứng dụng thời gian thực: RAG lý tưởng cho các ứng dụng như trợ lý ảo hay chatbot (ví dụ: Siri, Alexa) vì khả năng truy cập và cập nhật thông tin theo thời gian thực từ nhiều nguồn khác nhau, đáp ứng nhanh chóng các yêu cầu cụ thể từ người dùng.

+ Giải pháp tối ưu trong bối cảnh dữ liệu chưa được tổ chức: Khi có nhiều dữ liệu chưa được gán nhãn hoặc phân loại theo cách hữu ích, RAG giúp chuyển hóa thông tin đó thành các câu trả lời mạch lạc, chính xác cho các ứng dụng đòi hỏi truy vấn thông tin cụ thể như hướng dẫn sử dụng sản phẩm, tin tức,...

### **2.3.3 Công nghệ và thư viện được sử dụng:**

- Ngôn ngữ Backend: Python 3.x - Phổ biến, hệ sinh thái thư viện phong phú cho AI và web.
- Web Framework: Flask - Nhẹ, linh hoạt, dễ học và triển khai, phù hợp với quy mô dự án.
- Frontend: HTML, Tailwind CSS, Flowbite, JavaScript - Tạo giao diện hiện đại, responsive và tương tác.
- Text Extraction:
  - + PyPDF2: Thư viện phổ biến để đọc và trích xuất văn bản từ file PDF.
  - + python-docx: Thư viện để làm việc với file DOCX.
- NLP & RAG Core:
  - + NLTK: Thư viện cơ bản cho các tác vụ NLP như `sent_tokenize`, `word_tokenize`.
  - + Sentence Transformers: Thư viện mạnh mẽ và dễ sử dụng để tạo embedding văn bản chất lượng cao. Mô hình `all-mpnet-base-v2` được chọn vì cân bằng giữa hiệu năng và tốc độ.
  - + FAISS (faiss-cpu): Thư viện hiệu quả cao cho tìm kiếm tương đồng vector, phiên bản CPU phù hợp để triển khai mà không cần GPU. `IndexFlatL2` được sử dụng cho tìm kiếm chính xác trên tập dữ liệu vừa phải.
  - + Scikit-learn: Cung cấp `TfidfVectorizer` và `cosine_similarity` cho việc triển khai tìm kiếm từ khóa TF-IDF.
  - + `rank_bm25`: Thư viện triển khai thuật toán reranking BM25 Okapi.

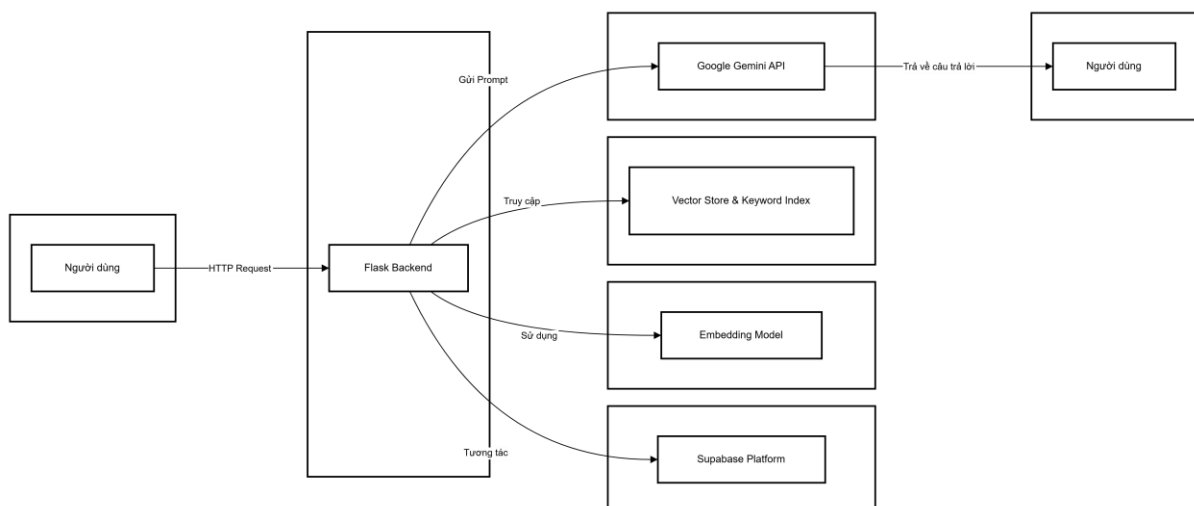


- + Underthesea, Pyvi: Các thư viện hỗ trợ xử lý tiếng Việt (chuẩn hóa, tách từ) cho các bước Query Transformation và Reranking.
- Large Language Model (LLM): Google Gemini API (gemini-2.0-flash) - Mô hình mạnh mẽ, có API dễ sử dụng, cung cấp khả năng sinh ngôn ngữ tốt và có bậc miễn phí. Thư viện google-generativeai được sử dụng để tương tác.
- Backend-as-a-Service (BaaS): Supabase - Cung cấp giải pháp toàn diện mã nguồn mở cho:
  - + Authentication: Quản lý đăng ký, đăng nhập (email/password, OAuth), JWT session.
  - + Database: Cơ sở dữ liệu PostgreSQL mạnh mẽ.
  - + API tự động: Tự động tạo RESTful API cho các bảng dữ liệu.
  - + Thư viện supabase-py: Python client để tương tác với Supabase API.
- Quản lý Biến môi trường: python-dotenv - Tải các biến môi trường (API keys, URL) từ file .env.
- Quản lý Phụ thuộc: pip và file requirements.txt.
- (Tùy chọn) Tunneling: pyngrok - Được sử dụng trong quá trình phát triển để đưa ứng dụng Flask local ra internet (không cần thiết cho sản phẩm cuối).

## 2.4 Thiết kế hệ thống:

### 2.4.1 Kiến trúc tổng thể:

Hệ thống được thiết kế theo kiến trúc client-server, tích hợp các dịch vụ đám mây và thư viện xử lý cục bộ để thực hiện quy trình RAG (Retrieval-Augmented Generation). Các thành phần chính bao gồm:



Hình 3: Kiến trúc tổng thể hệ thống RAG hiện tại

- Client (Trình duyệt): Giao diện người dùng được xây dựng bằng HTML, CSS và JavaScript (sử dụng Tailwind CSS và Flowbite). Người dùng tương tác với hệ thống thông qua trình duyệt để tải lên tài liệu, đặt câu hỏi và nhận câu trả lời. Giao diện này giao tiếp với backend thông qua các HTTP request (form submissions và AJAX).
- Web Server (Flask Backend): Là trung tâm xử lý của hệ thống, được xây dựng bằng Flask (Python). Backend đảm nhiệm các vai trò:
  - + Xử lý các HTTP request từ client (ví dụ: /upload, /query, /api/chat/...).
  - + Quản lý phiên làm việc (session) của người dùng.
  - + Tương tác với Supabase Platform để:
    - o Xác thực người dùng (đăng ký, đăng nhập, quản lý session).
    - o Lưu trữ và truy xuất siêu dữ liệu (metadata) về các tệp tin đã tải lên, lịch sử trò chuyện (chats, messages).
  - + Thực hiện luồng Ingestion: Nhận file từ client, trích xuất văn bản (extract\_text...), phân đoạn (chunking) văn bản (create\_...\_chunks), mã hóa chunk thành vector bằng Embedding Model, và lập chỉ mục vector vào Vector Store (FAISS) cùng chỉ mục từ khóa vào TF-IDF Index.
  - + Thực hiện luồng Query Retrieval: Nhận câu hỏi từ client, xử lý và mã hóa câu hỏi, thực hiện tìm kiếm kết hợp (Hybrid Search) trên FAISS và TF-IDF, xếp hạng lại (reranking) kết quả, xây dựng ngữ cảnh (context) tối ưu.
  - + Gọi LLM Service (Google Gemini API): Gửi prompt (bao gồm câu hỏi và ngữ cảnh đã xây dựng) đến Gemini API để sinh câu trả lời.
  - + Hậu xử lý câu trả lời từ LLM (sửa lỗi tiếng Việt, thêm nguồn) trước khi gửi về client.
- Supabase Platform: Dịch vụ Backend-as-a-Service (BaaS) cung cấp:
  - + Auth: Hệ thống xác thực và quản lý người dùng (email/password, OAuth với Google).
  - + Database (PostgreSQL): Lưu trữ dữ liệu có cấu trúc như thông tin người dùng, danh sách tệp tin, lịch sử trò chuyện (chats, messages).

- Embedding Model (Sentence Transformers): Thư viện Python (sentence-transformers) chạy trên Flask backend, chịu trách nhiệm chuyển đổi văn bản (chunks và câu hỏi) thành các vector embedding ngữ nghĩa. Mô hình cụ thể được sử dụng là 'all-mpnet-base-v2'.
- Vector Store & Keyword Index (Local Storage):
  - + FAISS Index: Chỉ mục vector được tạo và quản lý bởi thư viện faiss-cpu trên Flask backend. File index (faiss\_index.bin, vectors.pkl) được lưu trữ cục bộ trong thư mục riêng của mỗi người dùng trên server.
  - + TF-IDF Index: Chỉ mục từ khóa (tfidf\_vectorizer.pkl, tfidf\_matrix.pkl) được tạo và quản lý bởi scikit-learn trên Flask backend, cũng được lưu trữ cục bộ trong thư mục người dùng.
- LLM Service (Google Gemini API): Dịch vụ Mô hình Ngôn ngữ Lớn bên ngoài. Flask backend gửi yêu cầu chứa prompt (câu hỏi + ngữ cảnh) đến API này và nhận về câu trả lời do AI tạo ra.

#### 2.4.2 Thiết kế cơ sở dữ liệu (Supabase):

Cơ sở dữ liệu được thiết kế trên Supabase PostgreSQL để lưu trữ thông tin người dùng (thông qua Supabase Auth), thông tin tài liệu và lịch sử chat. Script supabase\_modules/setup\_database.sql định nghĩa schema chi tiết:

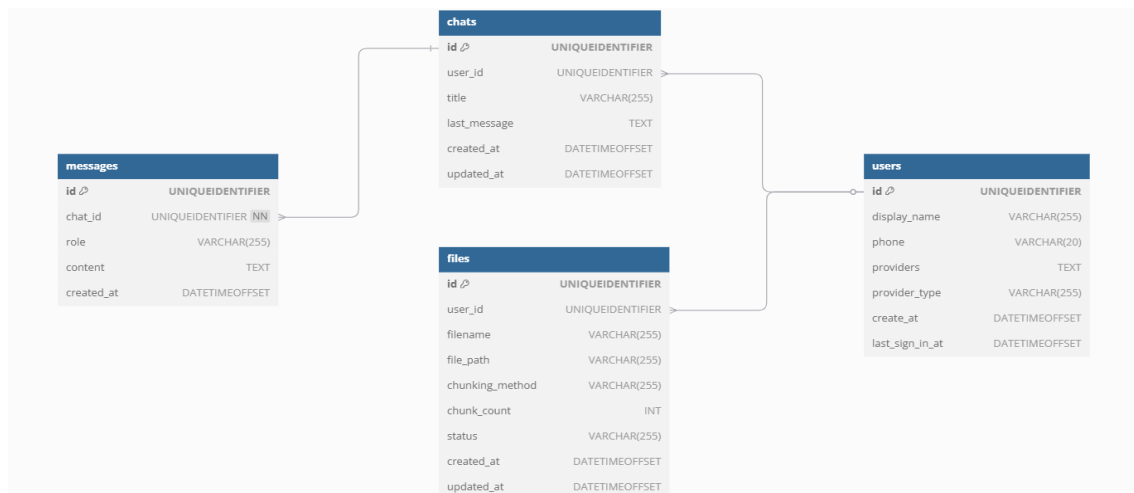
- **Bảng files:**
  - + id (UUID, PK): Khóa chính duy nhất cho mỗi file.
  - + user\_id (UUID, FK references auth.users): Liên kết đến người dùng sở hữu file.
  - + filename (TEXT): Tên gốc của file được tải lên.
  - + file\_path (TEXT): Đường dẫn tương đối đến file vật lý trên server (<user\_id>/<filename>).
  - + chunking\_method (TEXT): Phương pháp chunking đã sử dụng cho file này.
  - + chunk\_count (INTEGER): Số lượng chunks được tạo ra từ file.
  - + status (TEXT): Trạng thái xử lý file ('active', 'processing', 'error').
  - + created\_at, updated\_at (TIMESTAMPZ): Dấu thời gian.
- **Bảng chats:**
  - + id (UUID, PK): Khóa chính duy nhất cho mỗi cuộc trò chuyện.

- + user\_id (UUID, FK references auth.users): Liên kết đến người dùng sở hữu cuộc trò chuyện.
- + title (TEXT): Tiêu đề của cuộc trò chuyện (có thể do người dùng đặt hoặc tự động tạo).
- + last\_message (TEXT): Đoạn trích của tin nhắn cuối cùng (để hiển thị trong danh sách chat).
- + created\_at, updated\_at (TIMESTAMPZ): Dấu thời gian.

- **Bảng messages:**

- + id (UUID, PK): Khóa chính duy nhất cho mỗi tin nhắn.
- + chat\_id (UUID, FK references chats): Liên kết đến cuộc trò chuyện chứa tin nhắn.
- + role (TEXT): Vai trò của người gửi ('user' hoặc 'assistant').
- + content (TEXT): Nội dung của tin nhắn.
- + created\_at (TIMESTAMPZ): Dấu thời gian.

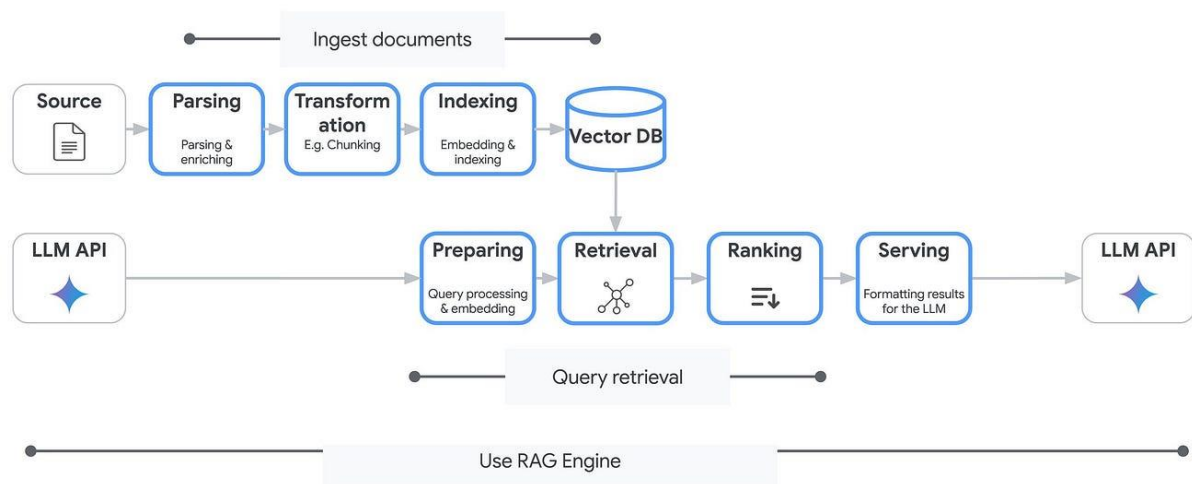
- **Bảng users** (Sử dụng Authentication được tích hợp sẵn của supabase)



Hình 4: Lược đồ cơ sở dữ liệu Supabase

### 2.4.3 Thiết kế luồng xử lý RAG:

Hệ thống hiện tại được em tham khảo thiết kế dựa theo sơ đồ hoạt động của công cụ RAG Vertex AI của Google.



Hình 5: Sơ đồ minh hoạt hệ thống Vertex AI RAG Engine của Google

Hệ thống được chia thành 2 luồng xử lý chính như sau tương ứng với sơ đồ trên:

### 1. Ingest documents (Nạp tài liệu):

- **Source:** Người dùng tải lên file (PDF, DOCX, TXT)
- **Parsing:** Các hàm `extract_text_pdf`, `extract_text_docx`, `extract_text_txt` trích xuất văn bản và thêm thông tin trang
- **Transformation:** Các hàm `create_hybrid_chunks`,... chia văn bản thành các đoạn (chunks).
- **Indexing:** Hàm `add_document` mã hóa (embedding) các chunk bằng SentenceTransformer và lưu vào index FAISS (đồng thời cũng tạo index TF-IDF).
- **Vector DB:** Index FAISS và ma trận TF-IDF được lưu trữ cục bộ (trong thư mục uploads hoặc thư mục riêng của người dùng).

### 2. Query retrieval (Truy xuất theo truy vấn):

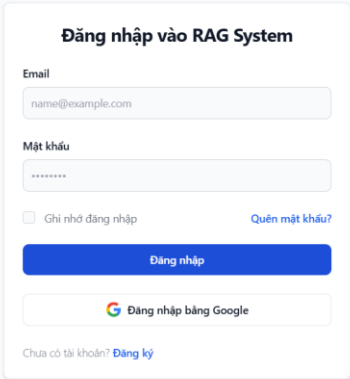
- **LLM API (Input):** Đại diện cho truy vấn người dùng nhập vào thông qua giao diện tương tác với LLM.
- **Preparing:** Câu hỏi được xử lý trong hàm `answer_question` (tách câu hỏi `split_multiple_questions`, phân loại, mở rộng nếu cần) và sau đó được mã hóa (embedding) bằng `embedder.encode`.
- **Retrieval:** Hệ thống sử dụng tìm kiếm kết hợp (Hybrid Search) trên FAISS (ngữ nghĩa) và TF-IDF (từ khóa) để tìm các chunk liên quan.

- **Ranking:** Kết quả từ retrieval được xếp hạng lại (rerank) bằng hàm rerank\_results\_for\_vietnamese dựa trên nhiều yếu tố (BM25, từ khóa, vị trí, độ dài, granularity) để tăng độ chính xác.
- **Serving:** Hàm build\_optimized\_context chọn lọc, sắp xếp các chunk đã rerank, thêm trích dẫn và định dạng thành ngữ cảnh (context) phù hợp với giới hạn token của LLM.
- **LLM API (Output):** Ngữ cảnh và câu hỏi được gửi đến API Gemini để tạo câu trả lời và trả về cho người dùng.

#### 2.4.4 Thiết kế giao diện người dùng:

Giao diện người dùng được thiết kế với mục tiêu đơn giản, trực quan và dễ sử dụng, sử dụng Tailwind CSS.

- **Trang Xác thực (login.html, register.html, forgot\_password.html):** Các form đơn giản để người dùng nhập thông tin, có hiển thị thông báo lỗi/thành công (flash messages). Hỗ trợ đăng nhập/đăng ký bằng Google.



Hình 6: Thiết kế giao diện Login (Đăng nhập hệ thống)

Hình 7: Thiết kế giao diện Register (Đăng ký tài khoản)

Hình 8: Thiết kế giao diện quên mật khẩu

- **Layout chính (index.html):**

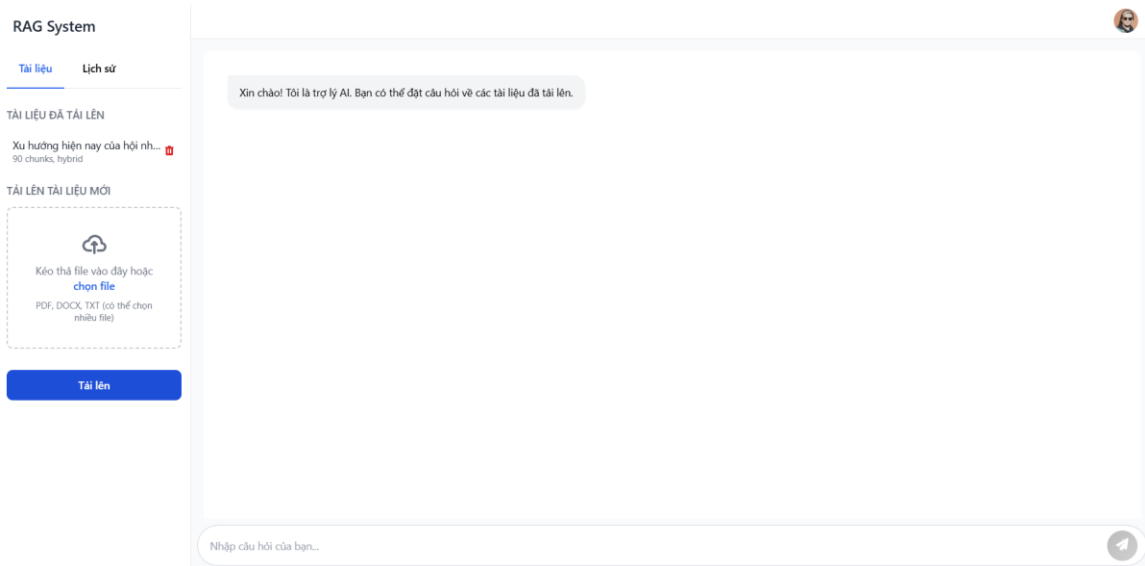
Navigation Bar: Hiển thị logo/tên ứng dụng, menu người dùng (hồ sơ, đăng xuất) hoặc nút đăng nhập/đăng ký.

Sidebar:

- + Có thể thu gọn/mở rộng trên desktop.
- + Chứa 2 tab chính: "Tài liệu" và "Lịch sử".
- + Tab "Tài liệu": Hiển thị danh sách file đã upload (tên, số chunks, phương pháp chunking), nút xóa file, khu vực upload file (drag-and-drop hoặc chọn file).
- + Tab "Lịch sử": Hiển thị danh sách các cuộc trò chuyện đã lưu (tiêu đề, thời gian), nút tạo chat mới, nút xóa chat.

### Khu vực Chat chính:

- + Hiển thị nội dung cuộc trò chuyện hiện tại, phân biệt tin nhắn user và bot.
- + Ô nhập câu hỏi cố định ở dưới cùng.
- + Hiệu ứng "đang soạn tin" (typing indicator).
- + Khu vực hiển thị câu hỏi gợi ý (khi không tìm thấy câu trả lời).



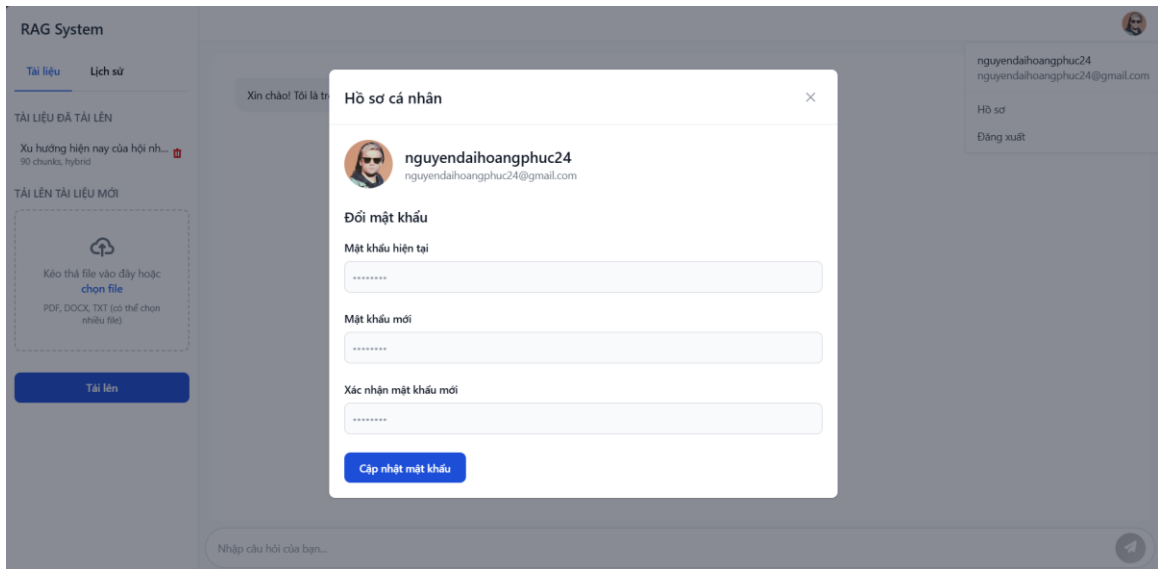
Hình 9: Thiết kế giao diện chính của hệ thống và quản lý tài liệu



Hình 10: Thiết kế giao diện chính của hệ thống và quản lý lịch sử chat

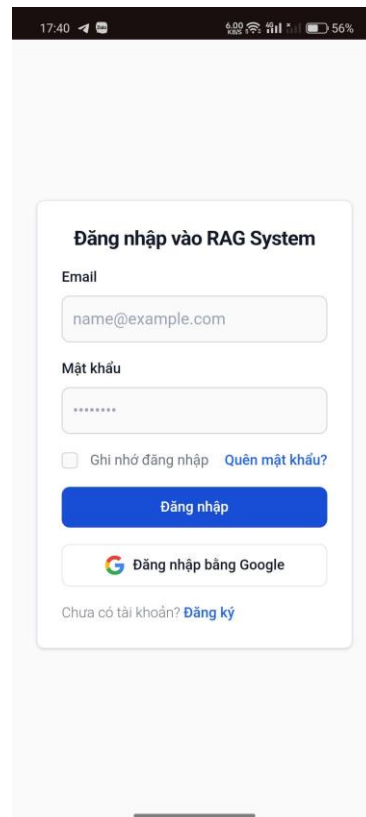
- **Trang hồ sơ (profile.html):** Hiển thị thông tin cơ bản của người dùng, form đổi mật khẩu.



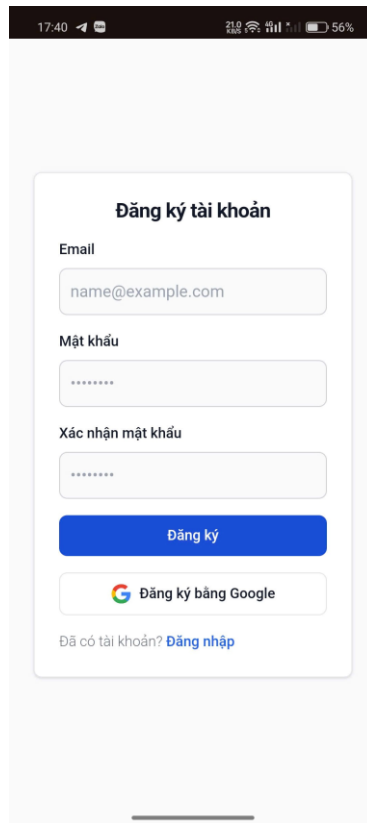


Hình 11: Trang hiển thị thông tin cá nhân cơ bản và đổi mật khẩu

Bên cạnh đó giao diện cũng được Responsive cho mobile như sau:



Hình 12: Giao diện mobile trang đăng nhập



17:40 56%

### Đăng ký tài khoản

Email

name@example.com

Mật khẩu

\*\*\*\*\*

Xác nhận mật khẩu

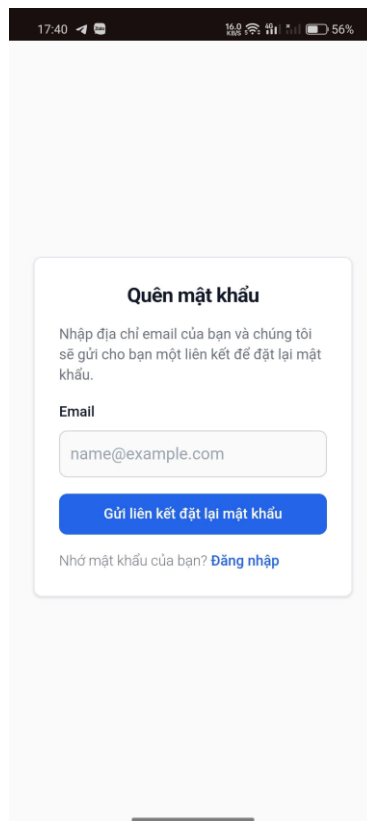
\*\*\*\*\*

Đăng ký

Đăng ký bằng Google

Đã có tài khoản? [Đăng nhập](#)

Hình 13: Giao diện mobile trang đăng ký tài khoản



17:40 56%

### Quên mật khẩu

Nhập địa chỉ email của bạn và chúng tôi sẽ gửi cho bạn một liên kết để đặt lại mật khẩu.

Email

name@example.com

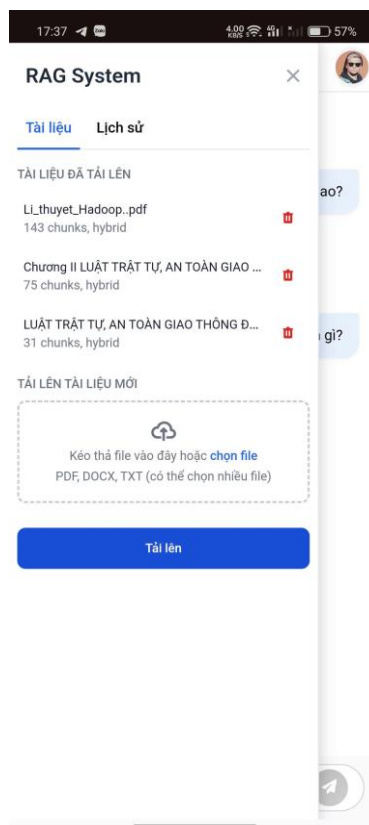
Gửi liên kết đặt lại mật khẩu

Nhớ mật khẩu của bạn? [Đăng nhập](#)

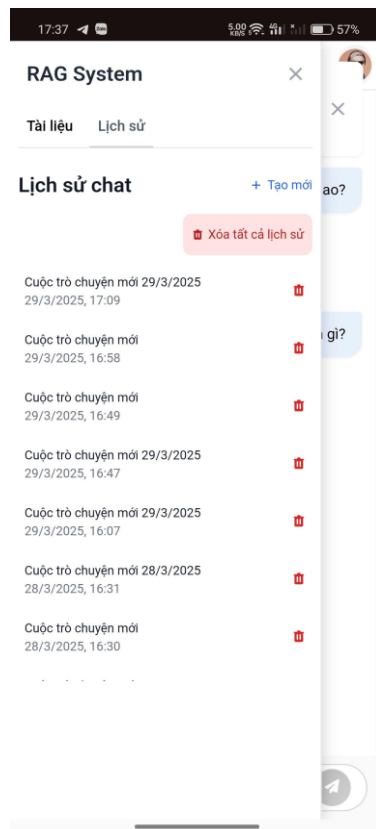
Hình 14: Giao diện mobile trang khôi phục mật khẩu



Hình 15: Giao diện mobile trang Chat chính



Hình 16: Giao diện mobile trang quản lý tài liệu



Hình 17: Giao diện mobile trang quản lý lịch sử chat



Hình 18: Giao diện mobile trang quản lý hồ sơ cá nhân

## 2.5 Các kĩ thuật được áp dụng trong hệ thống:

### 2.5.1 Xử lý và Nạp Tài liệu (Document Ingestion and Processing):

- Trích xuất văn bản (Parsing): Hệ thống có khả năng xử lý nhiều định dạng tài liệu đầu vào phổ biến như PDF (PyPDF2), DOCX (python-docx), và TXT. Các hàm `extract_text_pdf`, `extract_text_docx`, `extract_text_txt` được sử dụng để trích xuất nội dung văn bản thô.
- Phân đoạn Văn bản (Chunking): Đây là bước quan trọng để chia nhỏ tài liệu lớn thành các đoạn văn bản (chunks) có kích thước phù hợp cho việc lập chỉ mục và truy xuất. Hệ thống triển khai đa dạng các phương pháp chunking, cho phép lựa chọn hoặc kết hợp để tối ưu hóa hiệu quả:
  - + `create_sentence_windows`: Chia theo cửa sổ câu trượt.
  - + `create_paragraph_chunks`: Chia theo đoạn văn vật lý (dựa trên dấu xuống dòng).
  - + `create_semantic_chunks`: Cố gắng chia dựa trên ranh giới ngữ nghĩa, ưu tiên giữ các điều khoản luật hoặc các câu có liên quan gần nhau.
  - + `create_token_chunks`: Chia dựa trên số lượng token cố định, phù hợp với giới hạn của mô hình ngôn ngữ.
  - + `create_adaptive_chunks`: Tự động điều chỉnh kích thước chunk dựa trên cấu trúc (tiêu đề, mục) và mật độ nội dung.
  - + `create_hierarchical_chunks`: Phân tích cấu trúc phân cấp (chương, điều, mục) và tạo chunk kèm thông tin ngữ cảnh phân cấp.
  - + `create_contextual_chunks`: Giữ ngữ cảnh bằng cách nhóm các đoạn văn liên quan.
  - + `create_multi_granularity_chunks`: Tạo chunks ở nhiều cấp độ chi tiết (tài liệu, phần, đoạn, câu) để phục vụ các loại câu hỏi khác nhau.
  - + `create_hybrid_chunks`: Phương pháp kết hợp nhiều kỹ thuật trên để tạo bộ chunks tối ưu nhất.
- Việc triển khai đa dạng các chiến lược chunking là có chủ đích. Mỗi loại tài liệu (văn bản luật, giáo trình kỹ thuật, thông báo ngắn) có cấu trúc và đặc điểm riêng. Ví dụ, `create_semantic_chunks` phù hợp với văn bản luật để cố gắng giữ trọn vẹn các điều khoản, trong khi `create_paragraph_chunks` lại hiệu quả với các đoạn văn xuôi thông thường. `create_token_chunks` đảm bảo

kích thước đồng đều nhưng có thể cắt ngang ngữ nghĩa. Phương pháp `create_hybrid_chunks` được thiết kế để kết hợp ưu điểm của nhiều kỹ thuật như:

- + Multi-granularity chunks: Tạo chunks ở nhiều mức độ chi tiết khác nhau (tài liệu, phần, đoạn văn, câu) để tối ưu cho nhiều loại câu hỏi.
- + Contextual chunks: Chia văn bản dựa trên ngữ cảnh, đảm bảo giữ nguyên mối quan hệ ngữ nghĩa giữa các phần.
- + Paragraph chunks: Chia theo ranh giới đoạn văn truyền thống, phù hợp cho văn bản có cấu trúc rõ ràng.
- + Sentence windows: Tạo các cửa sổ câu trượt để bảo toàn ngữ cảnh cục bộ; nhằm tạo ra các chunks vừa đảm bảo ngữ nghĩa vừa có kích thước phù hợp cho việc embedding và truy xuất, tối ưu hóa cho bộ tài liệu đa dạng.
- Đánh dấu Metadata: Mỗi chunk được gắn kèm metadata quan trọng như tên file nguồn, phương pháp chunking, số trang để hỗ trợ quá trình truy xuất và reranking sau này.

### 2.5.2 Lập Chỉ mục (Indexing Strategies)

Để tìm kiếm nhanh chóng và hiệu quả, các chunk văn bản được chuyển đổi và lưu trữ trong các cấu trúc chỉ mục:

- Vector Embedding: Sử dụng mô hình SentenceTransformer ('all-mpnet-base-v2') để chuyển đổi mỗi chunk văn bản thành một vector số học (embedding) đại diện cho ý nghĩa ngữ nghĩa của nó. Mô hình all-mpnet-base-v2 được chọn sau khi cân nhắc giữa hiệu năng biểu diễn ngữ nghĩa (đặc biệt là khả năng xử lý tốt nhiều ngôn ngữ, bao gồm Tiếng Việt) và kích thước/tốc độ xử lý phù hợp cho môi trường thử nghiệm. Các mô hình lớn hơn có thể cho chất lượng embedding tốt hơn nhưng đòi hỏi tài nguyên cao hơn.
- Vector Database (FAISS): Các vector embedding được lưu trữ và lập chỉ mục bằng thư viện FAISS (Facebook AI Similarity Search). FAISS cho phép tìm kiếm tương đồng ngữ nghĩa cực kỳ nhanh chóng dựa trên khoảng cách vector (L2 - Euclidean distance). Hệ thống sử dụng IndexFlatL2. Việc sử dụng FAISS chạy cục bộ (local) là lựa chọn phù hợp trong giai đoạn thực tập và phát triển ban đầu do tính đơn giản trong cài đặt, không tốn chi phí và

đảm bảo tính riêng tư dữ liệu. So với các giải pháp đám mây (Pinecone, Weaviate), FAISS local thiếu đi khả năng scale tự động và các tính năng quản lý nâng cao, nhưng lại kiểm soát hoàn toàn môi trường. Các vector database mã nguồn mở khác như ChromaDB hay Milvus cũng là lựa chọn tiềm năng, tuy nhiên FAISS nổi bật về hiệu năng tìm kiếm tương đồng vector.

- Chỉ mục Từ khóa (TF-IDF): Song song với chỉ mục vector, hệ thống cũng xây dựng một chỉ mục dựa trên từ khóa bằng kỹ thuật TF-IDF thông qua `sklearn.feature_extraction.text.TfidfVectorizer`. Chỉ mục này giúp tìm kiếm các chunk chứa từ khóa chính xác có trong câu hỏi.

### 2.5.3 Xử lý Truy vấn và Truy xuất (Query Processing and Retrieval)

Khi người dùng đặt câu hỏi:

- Phân tách và Phân loại Truy vấn: Câu hỏi được phân tích để tách các câu hỏi con (nếu có) và xác định loại câu hỏi (tổng quan, chi tiết, về luật,...).
- Mở rộng Truy vấn (Query Expansion): Đối với các câu hỏi ngắn, đặc biệt là về điều luật, hệ thống có thể tự động mở rộng thành các câu hỏi đầy đủ hơn.
- Mã hóa Truy vấn: Câu hỏi (hoặc các biến thể) được mã hóa thành vector embedding tương tự như cách xử lý chunk.
- Truy xuất Kết hợp (Hybrid Retrieval): Hệ thống thực hiện đồng thời:
  - + Tìm kiếm Ngữ nghĩa: Dùng vector câu hỏi để truy vấn index FAISS, lấy ra các chunk gần nhất về mặt ngữ nghĩa.
  - + Tìm kiếm Từ khóa: Dùng vector TF-IDF của câu hỏi để truy vấn ma trận TF-IDF, lấy ra các chunk có độ tương đồng từ khóa cao nhất.
- Kết hợp Kết quả: Điểm số từ hai phương pháp tìm kiếm được kết hợp (với trọng số) để tạo danh sách ứng viên ban đầu.

### 2.5.4 Sắp xếp lại và Tối ưu hóa Ngữ cảnh:

- Bước Reranking là cực kỳ quan trọng vì bước Retrieval ban đầu (kể cả Hybrid Search) có thể trả về một số lượng lớn các chunks ứng viên (ví dụ top 20 chunks), trong đó không phải tất cả đều liên quan chặt chẽ đến câu hỏi. Reranking giúp lọc và sắp xếp lại danh sách này dựa trên các tiêu chí tinh vi hơn, đảm bảo các chunks chất lượng nhất được đưa vào ngữ cảnh cho LLM.

- Reranking Nâng cao: Các chunk (đoạn) ứng viên được đưa qua hàm rerank `_results_for_vietnamese`. Hàm này tính toán lại điểm liên quan dựa trên nhiều yếu tố phức tạp hơn, bao gồm:
  - + BM25: Thuật toán xếp hạng dựa trên tần suất từ.
  - + Khớp Từ khóa: Mức độ trùng khớp từ khóa.
  - + Vị trí, Độ dài Chunk: Ưu tiên chunk xuất hiện sớm, độ dài phù hợp.
  - + Mức độ chi tiết (Granularity): Phù hợp với loại câu hỏi.
  - + Độ quan trọng: Ưu tiên chunk được đánh dấu quan trọng hoặc là điều luật.
  - + Kết quả là một danh sách các chunk được sắp xếp lại theo độ liên quan thực tế cao hơn.

Ví dụ, thuật toán BM25 giúp ưu tiên các chunks chứa thuật ngữ quan trọng trong câu hỏi với tần suất phù hợp. Việc xem xét vị trí giúp ưu tiên thông tin ở đầu tài liệu hoặc đầu mục. Độ chi tiết (Granularity) đảm bảo chọn chunk phù hợp với loại câu hỏi (tổng quan hay chi tiết).
- Xây dựng Ngữ cảnh Tối ưu (`build_optimized_context`):
  - + Chọn lọc các chunk có điểm reranking cao nhất.
  - + Sắp xếp lại các chunk dựa trên loại câu hỏi (ví dụ: đưa chunk tổng quan lên đầu cho câu hỏi tổng quan, đưa chunk điều luật lên đầu cho câu hỏi về luật).
  - + Thêm thông tin trích dẫn (tên file, số trang) vào đầu mỗi chunk.
  - + Kiểm soát độ dài ngữ cảnh để không vượt quá giới hạn token của Mô hình Ngôn ngữ Lớn (LLM), cắt bớt nếu cần.

### 2.5.5 Sinh Câu trả lời và Hậu xử lý:

- Prompt Engineering: Ngữ cảnh tối ưu và câu hỏi được kết hợp thành một prompt chi tiết. Prompt này bao gồm "System Prompt" (hướng dẫn LLM cách trả lời, các ràng buộc về định dạng, độ chính xác, không dùng kiến thức ngoài, giữ văn phong gốc, xử lý tiếng Việt) và "User Prompt" (chứa câu hỏi và ngữ cảnh).
- Gọi LLM (Gemini): Prompt được gửi đến API của Google Gemini (`gemini-2.0-flash`) thông qua hàm `generate_with_retry`. Hàm này có cơ chế xử lý lỗi, tự động thử lại và chuyển đổi API key khi cần. Cấu hình `temperature` và `max_output_tokens` được điều chỉnh linh hoạt theo loại câu hỏi.



- Hậu xử lý câu trả lời:
  - + Loại bỏ các cụm từ thừa ở đầu/cuối.
  - + Sửa lỗi Tiếng Việt: Áp dụng regex và gọi lại Gemini (fix\_vietnamese\_spacing) để sửa lỗi tách từ tiếng Việt.
  - + Đảm bảo giữ nguyên định dạng liệt kê gốc.
  - + Xác định Nguồn Chính xác (identify\_most\_relevant\_pages): Phân tích câu trả lời và ngữ cảnh để xác định chính xác file và trang nào chứa thông tin được LLM sử dụng.
  - + Thêm thông tin nguồn đã được xác minh vào cuối câu trả lời.
- Câu hỏi Gợi ý (generate\_similar\_questions): Nếu không tìm thấy thông tin, hệ thống sử dụng LLM để tạo ra các câu hỏi tương tự, gợi ý cho người dùng.

#### **2.5.6 Tối ưu hóa cho Tiếng Việt:**

Hệ thống chú trọng việc xử lý và tối ưu hóa cho tiếng Việt:

- Thư viện NLP Tiếng Việt: Sử dụng các thư viện như underthesea (word tokenize, text normalize) và pyvi (ViTokenizer) cho các tác vụ chuẩn hóa và tách từ tiếng Việt (nếu có sẵn).
- BM25: Thuật toán BM25 được áp dụng trên văn bản đã được tokenize tiếng Việt để cải thiện xếp hạng từ khóa.
- Reranking: Các tiêu chí reranking có xem xét đặc thù của câu hỏi và tài liệu tiếng Việt.
- Hậu xử lý: Bước sửa lỗi tách từ bằng Gemini là một cơ chế quan trọng để đảm bảo chất lượng câu trả lời tiếng Việt.
- Prompt Engineering: Các chỉ dẫn trong system prompt yêu cầu LLM xử lý đúng ngữ pháp và văn phong tiếng Việt, giữ nguyên định dạng.

#### **2.5.7 Kiến trúc và Quản lý Hệ thống:**

- Framework Web: Sử dụng Flask để xây dựng giao diện web và API.
- Xác thực và Quản lý Người dùng (Supabase): Tích hợp với Supabase để quản lý đăng ký, đăng nhập, phiên làm việc, đặt lại mật khẩu (supabase\_modules.auth).
- Quản lý Dữ liệu Người dùng (Supabase & Local):

- + Lịch sử chat (`supabase_modules.chat_history`) và thông tin file (`supabase_modules.file_manager`) được lưu trữ trên Supabase, gắn với từng người dùng.
- + Dữ liệu trạng thái RAG (metadata, embeddings, indices FAISS/TF-IDF) được lưu cục bộ trong thư mục riêng của mỗi người dùng (`save_state`, `load_state`), đảm bảo tính riêng tư và cá nhân hóa.
- + Có cơ chế di chuyển dữ liệu từ `localStorage` sang Supabase khi người dùng đăng nhập lần đầu (`initialize_user_data`).
- Quản lý Trạng thái: Hệ thống lưu và tải trạng thái (metadata, indices) để duy trì dữ liệu giữa các phiên làm việc và khởi động lại.
- Xử lý Lỗi API Key: Tự động chuyển đổi giữa các API key Gemini khi gặp lỗi hết hạn mức hoặc lỗi key.
- Theo dõi Hiệu suất: Có các hàm cơ bản để theo dõi thời gian truy xuất, thời gian trả lời (`track_performance`, `analyze_performance`), mặc dù việc lưu trữ và phân tích chi tiết có thể cần hoàn thiện thêm.

## 2.6 Kết quả thực tế của hệ thống:

- Thời gian xử lý: Thời gian xử lý phụ thuộc vào cấu hình máy chủ/máy trạm thực thi. Trên môi trường thử nghiệm với cấu hình (Laptop có CPU Core i5, 16GB RAM) thì thời gian xử lý trung bình (trích xuất, chunking, embedding, indexing) cho một file tài liệu là khá lâu. Ví dụ: đối với 1 file tài liệu PDF có độ dài khoảng 11 trang thì máy em xử lý mất khoảng 40 giây. Nhưng đối với các máy chủ có cấu hình mạnh hơn thì thời gian xử lý sẽ nhanh hơn.
- Thời gian phản hồi: Thời gian trung bình từ khi người dùng gửi câu hỏi đến khi nhận được câu trả lời từ chatbot đối với các truy vấn thông thường thường dưới 2 đến 5 giây cho những câu hỏi có câu trả lời ngắn (ví dụ như: Các câu hỏi về khái niệm,...) Và mất khoảng trên 10 giây đối với những câu hỏi cần câu trả lời dài (ví dụ như: Trình bày chi tiết về,...)
- Độ chính xác/liên quan: Qua quá trình thử nghiệm với bộ tài liệu mẫu về các tài liệu của pháp luật (Ví dụ: LUẬT TRẬT TỰ, AN TOÀN GIAO THÔNG ĐƯỜNG BỘ.pdf) thì hệ thống RAG cho thấy khả năng truy xuất các đoạn văn bản (chunks) liên quan cao. Câu trả lời được sinh ra bởi Gemini API

thường bám sát vào ngữ cảnh được cung cấp, giảm thiểu hiện tượng 'ảo giác' (hallucination) và có kèm theo trích dẫn nguồn (tên file, trang ước tính) giúp người dùng kiểm chứng. Dưới đây là ví dụ:



Hình 19: Kiểm tra độ chính xác của hệ thống RAG

Với tài liệu thật dưới đây thì đúng là điều 89 bắt đầu từ trang 59 trong tài liệu này:

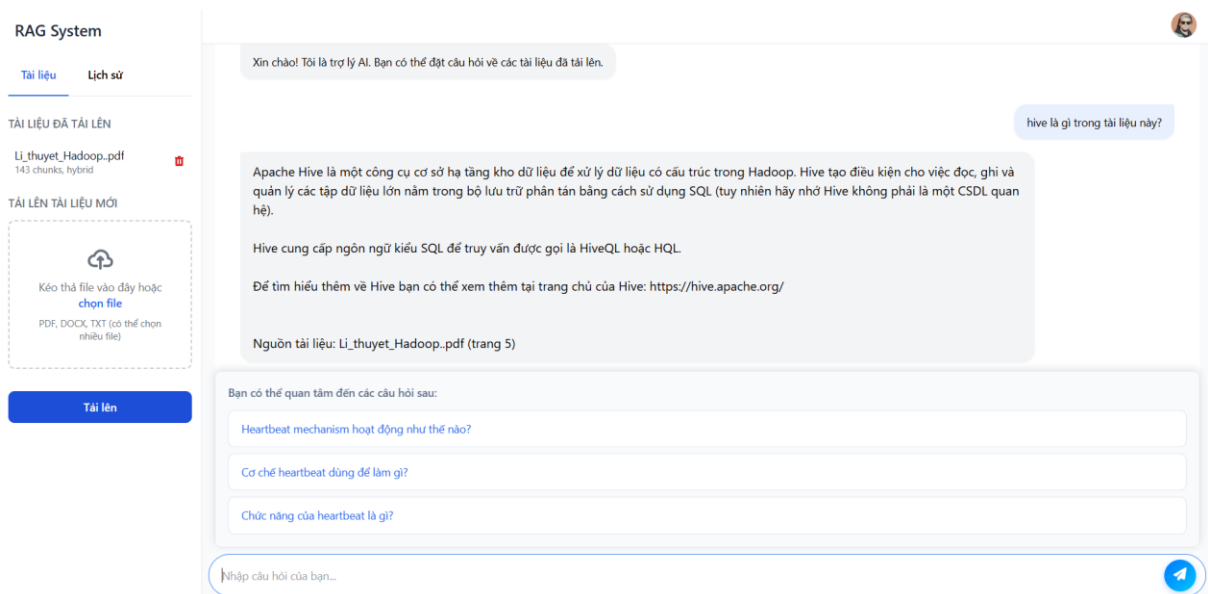
#### Điều 89. Quy định chuyển tiếp

1. Giấy phép lái xe được cấp trước ngày Luật này có hiệu lực thi hành được tiếp tục sử dụng theo thời hạn ghi trên giấy phép lái xe.
2. Giấy phép lái xe được cấp trước ngày Luật này có hiệu lực thi hành nếu chưa thực hiện đổi, cấp lại theo quy định của Luật này có hiệu lực sử dụng như sau:
  - a) Giấy phép lái xe hạng A1 được tiếp tục điều khiển xe mô tô hai bánh có dung tích xi-lanh từ 50 cm<sup>3</sup> đến dưới 175 cm<sup>3</sup> hoặc có công suất động cơ điện từ 04 kW đến dưới 14 kW;
  - b) Giấy phép lái xe hạng A2 được tiếp tục điều khiển xe mô tô hai bánh có dung tích xi-lanh từ 175 cm<sup>3</sup> trở lên hoặc có công suất động cơ điện từ 14 kW trở lên và các loại xe quy định cho giấy phép lái xe hạng A1 quy định tại điểm a khoản này;

- c) Giấy phép lái xe hạng A3 được tiếp tục điều khiển xe mô tô ba bánh, các loại xe quy định cho giấy phép lái xe hạng A1 quy định tại điểm a khoản này và các xe tương tự;

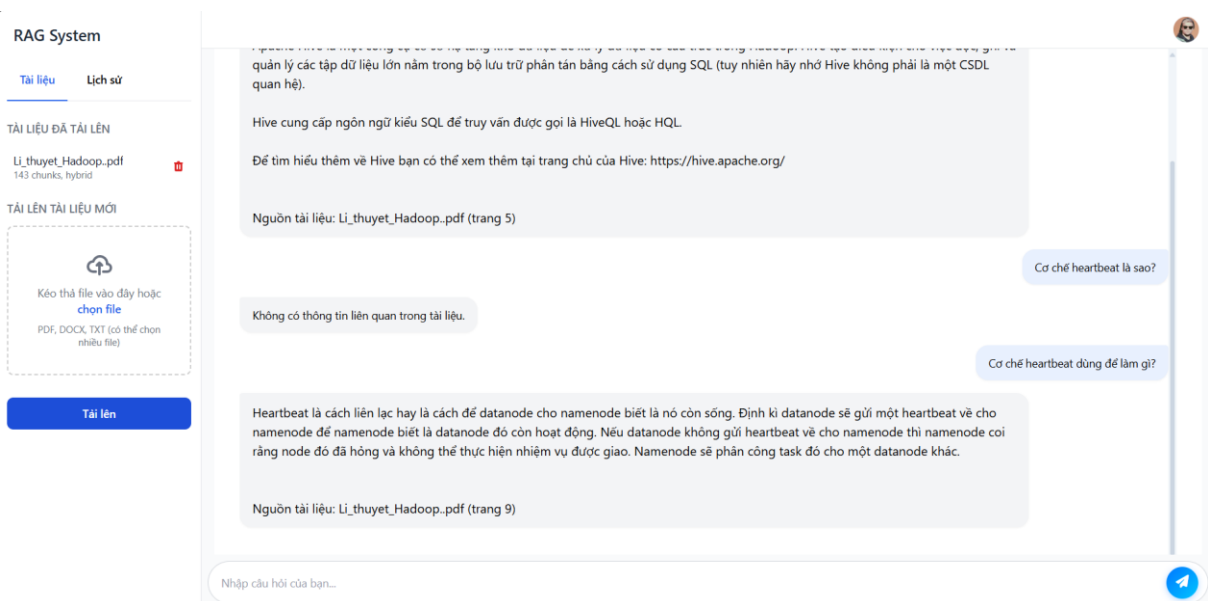
Hình 20: Đối chiếu với tài liệu thật

- Khả năng xử lý lỗi: Đôi khi hệ thống có thể không trả lời được câu hỏi của người dùng thì sẽ đưa ra một số câu hỏi mẫu có ý nghĩa tương tự để người dùng chọn.



Hình 21: Khả năng xử lý lỗi khi không trả lời được câu hỏi

Và hệ thống có thể trả lời ngay với câu hỏi tương tự đó:



Hình 22: Kết quả của khả năng xử lý lỗi khi không trả lời được câu hỏi

## PHẦN 3 KẾT LUẬN

### 3.1 Tự nhận xét, đánh giá

Qua 06 tuần thực tập tốt nghiệp tại Trung tâm Ngoại ngữ - Tin học Victory, bản thân em đã có cơ hội quý báu để áp dụng kiến thức đã học vào thực tế, đồng thời tiếp thu nhiều kinh nghiệm và kỹ năng mới trong lĩnh vực phát triển phần mềm, đặc biệt là trong việc xây dựng ứng dụng tích hợp Trí tuệ Nhân tạo (AI).

- Về kiến thức và kỹ năng: Em đã củng cố vững chắc kiến thức về phát triển web với Flask, quản lý cơ sở dữ liệu và xác thực người dùng thông qua việc tích hợp Supabase. Quan trọng hơn, em đã đi sâu tìm hiểu và triển khai thành công kiến trúc RAG (Retrieval-Augmented Generation), một kỹ thuật tiên tiến và hiệu quả cho các bài toán hỏi đáp dựa trên tài liệu. Em đã nắm bắt và áp dụng được các kỹ thuật cốt lõi của RAG như:

- + Trích xuất và xử lý văn bản từ nhiều định dạng file.
- + Thực hiện các chiến lược phân đoạn văn bản (chunking) đa dạng (sentence, paragraph, semantic, token, adaptive, hierarchical, multi-granularity, hybrid) để tối ưu hóa việc truy xuất.
- + Xây dựng và sử dụng các chỉ mục tìm kiếm kết hợp (Hybrid Search) bao gồm Vector Embedding (SentenceTransformer + FAISS) và chỉ mục từ khóa (TF-IDF).
- + Triển khai kỹ thuật Reranking nâng cao (sử dụng BM25, từ khóa, vị trí, độ dài, granularity) để cải thiện độ chính xác của kết quả truy xuất, đặc biệt tối ưu cho tiếng Việt.
- + Thiết kế và tối ưu hóa prompt (Prompt Engineering) cho mô hình ngôn ngữ lớn (Gemini) để đảm bảo câu trả lời chính xác, đầy đủ và giữ nguyên văn phong tài liệu gốc.
- + Xử lý các vấn đề đặc thù của tiếng Việt trong NLP như tách từ và sửa lỗi dấu cách.
- + Quản lý API key và xử lý lỗi hiệu quả.

- Về kinh nghiệm thực tiễn: Em đã học được cách làm việc trong một quy trình phát triển phần mềm, từ việc phân tích yêu cầu, lựa chọn công nghệ, thiết kế kiến trúc, lập trình, kiểm thử đến triển khai cơ bản. Việc đối mặt và giải quyết các thách thức kỹ thuật như tối ưu hóa hiệu suất truy xuất, xử lý giới hạn token của LLM, và

đảm bảo tính chính xác của nguồn thông tin đã giúp em rèn luyện khả năng giải quyết vấn đề và tư duy logic.

- Điểm mạnh: Em nhận thấy mình có khả năng học hỏi khá tốt với các công nghệ mới, chủ động tìm tòi giải pháp cho các vấn đề kỹ thuật phức tạp. Việc triển khai thành công nhiều phương pháp chunking và kỹ thuật reranking phức tạp cho thấy khả năng nắm bắt và áp dụng các thuật toán AI. Khả năng tích hợp các dịch vụ bên ngoài như Supabase và Google Gemini cũng là một điểm mạnh.

- Hạn chế và hướng khắc phục: Mặc dù đã cố gắng, em nhận thấy vẫn cần cải thiện kỹ năng tối ưu hóa mã nguồn để hệ thống hoạt động hiệu quả hơn nữa, đặc biệt là tối ưu tốc độ truy xuất, reranking với lượng dữ liệu lớn và một vấn đề nữa là đôi khi hệ thống ước tính số trang chưa chính xác, dẫn đến việc trích dẫn nguồn thông tin chưa đúng trang. Giao diện người dùng (UI/UX) hiện tại còn khá cơ bản và cần được đầu tư cải thiện để thân thiện hơn. Trong tương lai, em sẽ tiếp tục nghiên cứu sâu hơn về các kỹ thuật RAG, các mô hình embedding và LLM mới, cũng như các phương pháp tối ưu hóa hiệu năng hệ thống.

- Hướng phát triển:

+ Tối ưu hóa Chunking

- Tự động chọn phương pháp chunking theo loại tài liệu hoặc câu hỏi.
- Đánh giá hiệu quả chunking để tối ưu hóa độ chính xác.
- Nghiên cứu chunking dựa trên đồ thị tri thức để tăng độ mạch lạc.

+ Nâng cao Embedding Models

- Tích hợp các mô hình embedding mạnh hơn, tối ưu cho tiếng Việt (PhoBERT, vEmbed).
- Fine-tune embedding để cải thiện độ chính xác trên dữ liệu chuyên ngành.

+ Cải tiến Truy xuất & Reranking

- Sử dụng FAISS nâng cao (IndexIVFFlat, HNSWFlat) để tăng tốc tìm kiếm.
- Áp dụng mô hình reranker mạnh như cross-encoder hoặc Learning-to-Rank.
- Mở rộng truy vấn để tăng tính bao phủ thông tin.

+ Tối ưu hóa Ngữ cảnh

- Nén ngữ cảnh thông minh để tối ưu hóa số lượng token.
- Cải thiện khả năng ghi nhớ hội thoại và tóm tắt trước khi sinh câu trả lời.

+ Tích hợp LLM & Kỹ thuật Prompt

- Thử nghiệm các mô hình LLM mới phù hợp với RAG và tiếng Việt.
- Xem xét tinh chỉnh LLM trên tập dữ liệu đặc thù.
- + Tăng cường Hỗ trợ Tiếng Việt
  - Tích hợp sâu hơn NLP tiếng Việt, xử lý sắc thái ngữ nghĩa chính xác hơn.
  - Ưu tiên sử dụng mô hình ngôn ngữ và embedding tối ưu cho tiếng Việt.
- + Cải thiện UI/UX
  - Trực quan hóa nguồn tài liệu, cải thiện trải nghiệm người dùng.
  - Tích hợp phản hồi để thu thập dữ liệu cải thiện hệ thống.
  - Mở rộng tìm kiếm lịch sử chat và cá nhân hóa tùy chọn hệ thống.
- + Nâng cao Khả năng Triển khai
  - Container hóa với Docker, triển khai trên cloud (AWS, GCP, Azure).
  - Xử lý tác vụ bất đồng bộ để tăng hiệu suất.

### **3.2 Kết luận, kiến nghị**

Quá trình thực tập đã giúp em không chỉ củng cố kiến thức lý thuyết mà còn trang bị những kỹ năng thực hành quý báu trong việc xây dựng một ứng dụng web hoàn chỉnh, tích hợp các công nghệ AI tiên tiến như kiến trúc RAG. Việc lựa chọn và triển khai thành công giải pháp kỹ thuật này là một bước tiến quan trọng. Dự án đã hoàn thành các mục tiêu đề ra, tạo ra một hệ thống hỏi đáp tài liệu thông minh, cá nhân hóa và có khả năng mở rộng. Những kinh nghiệm tích lũy được trong kỳ thực tập này chắc chắn sẽ là nền tảng vững chắc cho sự nghiệp phát triển phần mềm của em trong tương lai.