**Update RAG Data**

# Đối tượng sử dụng

**Quản lý hệ thống**: Những người chịu trách nhiệm quản lý, cập nhật và phân quyền dữ liệu trong hệ thống RAG, bao gồm quản trị viên hệ thống và nhân sự phụ trách tri thức của tổ chức.

# Tác dụng của hệ thống

Hệ thống cập nhật dữ liệu RAG được thiết kế nhằm đáp ứng các mục tiêu sau:

* **Cập nhật tri thức cho RAG**: Đảm bảo dữ liệu mới được tích hợp chính xác và kịp thời vào hệ thống RAG, cung cấp thông tin cập nhật cho các Agent phù hợp.
* **Phân quyền truy cập tài liệu**: Quản lý quyền truy cập vào các tài liệu dựa trên vai trò, bộ phận hoặc cá nhân, đảm bảo tính bảo mật và phù hợp của thông tin.

# Yêu cầu triển khai chi tiết

## Quản lý các loại dữ liệu riêng

Hệ thống phân loại dữ liệu dựa trên thư mục lưu trữ riêng biệt, thay vì gắn loại tài liệu trong metadata:

* **Thông báo sinh viên**: Tài liệu được lưu trong thư mục dành riêng cho sinh viên (ví dụ: /student\_notifications).
* **Thông báo giáo viên**: Tài liệu được lưu trong thư mục dành riêng cho giáo viên (ví dụ: /teacher\_notifications).
* **Thông báo chung**: Tài liệu được lưu trong thư mục chung (ví dụ: /general\_notifications).

Khi tài liệu được tải lên, hệ thống tự động xác định loại dữ liệu dựa trên thư mục lưu trữ và chuyển đến Agent tương ứng (Agent sinh viên, Agent giáo viên, Agent chung).

**Giải pháp**:

Mỗi tài liệu khi được tải lên sẽ được lưu vào thư mục tương ứng (sinh viên, giáo viên, chung).

Metadata của tài liệu sẽ bao gồm:

* **doc\_id**: Mã định danh duy nhất của tài liệu để liên kết với cơ sở dữ liệu.
* **Chủ đề** (subject, ví dụ: Công nghệ thông tin, Toán học).
* **ID người dùng** (user\_id, để xác định người có quyền truy cập).

Dữ liệu sẽ được embedding và lưu trữ trong cơ sở dữ liệu vector, với các Agent truy cập vào các phân vùng dữ liệu tương ứng dựa trên thư mục và doc\_id.

## Quản lý phân quyền trên từng file

Hệ thống đảm bảo phân quyền chi tiết cho từng tài liệu:

* **Phân quyền theo nhóm:** Ví dụ, tài liệu trong thư mục giáo viên bộ môn Công nghệ thông tin chỉ có thể truy cập bởi giáo viên thuộc bộ môn này.
* **Phân quyền cá nhân:** Một số tài liệu có thể được gửi riêng cho từng giáo viên hoặc sinh viên dựa trên user\_id.

**Giải pháp:**

* Lưu trữ thông tin phân quyền trong cơ sở dữ liệu với bảng phân quyền liên kết doc\_id (ID tài liệu).
* Khi một Agent truy xuất tài liệu, hệ thống kiểm tra quyền truy cập dựa trên metadata (doc\_id, user\_id, subject) và thư mục lưu trữ trước khi trả về kết quả.

## Quản lý metadata

Metadata đóng vai trò quan trọng trong việc tổ chức và truy xuất dữ liệu, với doc\_id là trường chính để liên kết giữa file và cơ sở dữ liệu.

Metadata cơ sở dữ liệu:

…

**Giải pháp:**

Sử dụng cơ sở dữ liệu NoSQL để lưu trữ metadata tài liệu và phân quyền.

# Quy trình triển khai

Quy trình cập nhật dữ liệu RAG được thiết kế để đảm bảo tài liệu được tải lên, xử lý, lưu trữ và phân quyền một cách chính xác, hiệu quả. Quy trình này gồm bốn giai đoạn chính: tải lên tài liệu, gắn metadata, embedding dữ liệu và quản lý phân quyền. Mỗi giai đoạn được thực hiện tuần tự, sử dụng các công nghệ phù hợp để đảm bảo dữ liệu sẵn sàng phục vụ các Agent một cách bảo mật và tối ưu.

## Tải lên tài liệu

* Quản lý hệ thống tải tài liệu lên thông qua giao diện người dùng thân thiện hoặc API RESTful. Giao diện cho phép chọn thư mục lưu trữ tương ứng, chẳng hạn thông báo sinh viên, thông báo giáo viên, hoặc thông báo chung.
* Sau khi tải lên, hệ thống kiểm tra định dạng file (hỗ trợ PDF, DOCX, TXT, v.v.) bằng các thư viện như PyPDF2 hoặc python-docx để trích xuất nội dung. Nếu file không đúng định dạng hoặc không thể trích xuất, hệ thống thông báo lỗi và yêu cầu người dùng tải lại. Nội dung trích xuất được lưu tạm thời để xử lý ở các bước tiếp theo.

## Gắn metadata

* Sau khi trích xuất nội dung, hệ thống tiến hành gắn metadata để tổ chức và hỗ trợ truy xuất tài liệu. Quản lý hệ thống nhập thông tin metadata quan trọng, bao gồm user\_id (định danh người hoặc nhóm được phép truy cập) và subject (chủ đề tài liệu, ví dụ: Công nghệ thông tin, Toán học) thông qua giao diện. Metadata này sẽ được đính kèm với tài liệu để embedding sau này.
* Đồng thời metadata được lưu trữ trong cơ sở dữ liệu NoSQL như MongoDB hoặc DynamoDB với chi tiết hơn, với cấu trúc linh hoạt để hỗ trợ mở rộng. Ví dụ, một tài liệu có metadata như: {doc\_id: "uuid123", user\_id: ["user1", "user2"], subject: "Công nghệ thông tin", created\_at: "2025-07-29T16:08:00", folder\_path: "/teacher\_notifications"}. Giai đoạn này đảm bảo tài liệu được gắn thông tin đầy đủ để phân quyền và truy xuất sau này.

## Embedding dữ liệu

Giai đoạn embedding chuyển đổi nội dung tài liệu thành vector để hỗ trợ tìm kiếm ngữ nghĩa. Quy trình này gồm ba bước: chia đoạn nội dung, tạo embedding và lưu trữ trong FAISS.

* Đầu tiên, nội dung tài liệu được chia thành các đoạn nhỏ (chunk) với kích thước từ 512 đến 2000 ký tự, sử dụng thư viện LangChain hoặc SentenceTransformers. Việc chia đoạn đảm bảo mỗi đoạn giữ được ngữ nghĩa đầy đủ, ví dụ: một thông báo dài 10.000 ký tự có thể được chia thành 5-10 đoạn, mỗi đoạn đại diện cho một phần nội dung độc lập.
* Tiếp theo, mỗi đoạn được chuyển thành vector embedding bằng mô hình Gemini 2.5 Flash, được chọn vì độ chính xác cao và hiệu quả với văn bản. Các embedding được gắn với doc\_id để duy trì liên kết với tài liệu gốc.
* Cuối cùng, các vector embedding được lưu trữ trong cơ sở dữ liệu vector FAISS. Tùy vào quy mô dữ liệu, hệ thống sử dụng IndexFlatL2 cho tìm kiếm chính xác hoặc IndexIVFFlat cho tìm kiếm gần đúng nhanh hơn. Mỗi vector được lưu cùng doc\_id và folder\_path để đảm bảo các Agent chỉ truy xuất dữ liệu từ phân vùng phù hợp.

## Quản lý phân quyền

* Giai đoạn này đảm bảo chỉ người dùng hoặc nhóm được phép mới truy cập được tài liệu. Thông tin phân quyền được lưu trong cơ sở dữ liệu NoSQL, sử dụng bảng riêng liên kết doc\_id với danh sách user\_id hoặc group\_id. Ví dụ, một tài liệu trong agent giáo viên có thể chỉ định chỉ giáo viên bộ môn Công nghệ thông tin được truy cập, dựa trên group\_id hoặc subject.
* Khi Agent gửi truy vấn, hệ thống thực hiện kiểm tra quyền truy cập:
* Kiểm tra user\_id hoặc group\_id của Agent trong bảng phân quyền để xác minh quyền truy cập với doc\_id.
* Xác minh subject của tài liệu phù hợp với quyền của Agent (ví dụ, Agent giáo viên Công nghệ thông tin chỉ truy xuất tài liệu có subject là "Công nghệ thông tin").
* Nếu quyền hợp lệ, hệ thống truy xuất embedding từ FAISS và trả kết quả; nếu không, trả về thông báo lỗi.

# Công nghệ sử dụng

* **FAISS:** Lưu trữ và truy xuất vector embedding, hỗ trợ tìm kiếm gần đúng với hiệu suất cao.
* **Gemini 2.5 Flash:** Mô hình tạo embedding với độ chính xác cao, phù hợp cho các tài liệu văn bản.
* **NoSQL (MongoDB/DynamoDB):** Lưu trữ metadata và thông tin phân quyền.
* **API RESTful:** Giao tiếp giữa giao diện người dùng, Agent và hệ thống backend.
* **LangChain/SentenceTransformers:** Xử lý chia đoạn nội dung và tạo embedding (nếu cần bổ trợ cho Gemini).

# Giao diện

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**CHI TIẾT TRIỂN KHAI**

Tổng thời gian: 46 giờ

1. Thiết lập môi trường và cơ sở dữ liệu: 4 giờ

- Cài đặt Python, Node.js, MongoDB, và FAISS.

- Tạo collection trong MongoDB cho:

- documents: Lưu doc\_id, user\_id, subject, folder\_path.

- permissions: Lưu doc\_id, user\_id, group\_id.

- Cấu hình FAISS với IndexFlatL2 cho lưu trữ vector embedding.

- Kết nối và kiểm tra Gemini 2.5 Flash API.

2. Phát triển backend và API: 12 giờ

- Xây dựng API RESTful bằng FastAPI:

- POST /upload: Hỗ trợ tải file (PDF, DOCX, TXT), trích xuất nội dung bằng PyPDF2/python-docx, lưu vào thư mục tương ứng (/student\_notifications, /teacher\_notifications, /general\_notifications).

- POST /metadata: Gắn metadata (doc\_id, user\_id, subject, folder\_path) và lưu vào MongoDB.

- POST /embed: Chia nội dung thành đoạn (512-2000 ký tự) bằng LangChain, tạo embedding bằng Gemini 2.5 Flash, lưu vector vào FAISS.

- GET /query: Kiểm tra phân quyền (user\_id, group\_id, subject), truy xuất vector từ FAISS, trả kết quả.

- Xử lý lỗi: Kiểm tra định dạng file, thông báo nếu file không hợp lệ.

3. Phát triển giao diện người dùng (React): 24 giờ

- Thiết kế giao diện bằng React và Tailwind CSS:

- Trang tải lên: Form chọn file và thư mục (sinh viên, giáo viên, chung), hiển thị thông báo lỗi nếu file không hợp lệ.

- Trang nhập metadata: Form nhập user\_id, subject, tự động điền folder\_path dựa trên thư mục.

- Trang quản lý phân quyền: Giao diện chọn user\_id/group\_id để gán quyền.

- Trang truy vấn: Form nhập truy vấn, hiển thị danh sách tài liệu được phép truy cập.

- Tích hợp API với Axios: Gửi dữ liệu tới các endpoint /upload, /metadata, /embed, /query.

- Đảm bảo giao diện responsive, hỗ trợ Chrome, Firefox, và thiết bị di động.

4. Kiểm thử và triển khai: 6 giờ

- Kiểm tra toàn diện: Tải file, trích xuất nội dung, gắn metadata, embedding, phân quyền, truy vấn.

- Kiểm tra giao diện trên các trình duyệt và thiết bị.

- Sửa lỗi và triển khai trên server/localhost.