GIỚI THIỆU VỀ MÔ HÌNH RAG VÀ TRIỂN KHAI

Nội dung

[1. RAG là gì ? 3](#_Toc205450423)

[2. Mô hình triển khai RAG Mongo 3](#_Toc205450424)

[A. Ingestion (Tiền xử lý dữ liệu) 5](#_Toc205450425)

[B. Retrieval (Truy xuất dữ liệu liên quan) 5](#_Toc205450426)

[C. MongoDB Atlas Vector Search là gì? 6](#_Toc205450427)

[3. Triển khai dự án 6](#_Toc205450428)

[A.Mô tả dự án 6](#_Toc205450429)

[B.Triển khai 6](#_Toc205450430)

[1. Back-end 6](#_Toc205450431)

[2. Front-end 6](#_Toc205450432)

# RAG là gì ?

* **RAG** là một cách triển khai **Retrieval-Augmented Generation (RAG)** sử dụng **MongoDB** (cụ thể là **MongoDB Atlas Vector Search**) làm **vector database** để lưu và truy vấn dữ liệu **embedding** phục vụ cho việc tìm kiếm thông tin liên quan.
* **Tóm tắt :**

RAG (Retrieval-Augmented Generation) là kiến trúc mà mô hình ngôn ngữ lớn (LLM) được kết hợp với một công cụ truy vấn dữ liệu bên ngoài để:

1. "Retrieve" – truy vấn các tài liệu liên quan từ một kho kiến thức (vector store).
2. "Augment" – đưa dữ liệu truy vấn vào prompt.
3. "Generate" – mô hình LLM sinh câu trả lời từ nội dung được tăng cường.

Ở trong ứng dụng sắp triển khai, ta chỉ sử dụng R ~ Retrieve để tìm kiếm tài liệu tương đồng nghĩa.

* **RAG MONGO là gì ?**

Là kiến trúc **RAG** trong đó dung **Mongo Atlas** để :

* + Lưu các **document** và **embedding vector** tương ứng (thường là từ SentenceTransformer).
  + Truy vấn bằng **$vectorSearch** để tìm các tài liệu liên quan (similarity search).
  + Dùng các tài liệu này làm **context đầu vào** cho LLM (ví dụ GPT, llama, Claude…) để tạo câu trả lời.

# Mô hình triển khai RAG Mongo

A diagram of a diagram

AI-generated content may be incorrect.

*(Nguồn :* [*https://www.mongodb.com/docs/atlas/atlas-vector-search/rag/*](https://www.mongodb.com/docs/atlas/atlas-vector-search/rag/) *)*

*A diagram of a diagram

AI-generated content may be incorrect.*

*(Nguồn : https://medium.com/@yashpaliwal42/why-mongodb-leads-the-way-in-retrieval-augmented-generation-rag-a8d5653973cd)*

* Mô hình này gồm 3 giai đoạn chính: **Ingestion**, **Retrieval**, và **Generation** , tuy nhiên ta sẽ chỉ cần quan tâm về Ingestion và Retrieval cho mục đích tìm kiếm dữ liệu liên quan.

## A. Ingestion (Tiền xử lý dữ liệu)

- Lấy dữ liệu đầu vào (ví dụ: tài liệu, báo cáo, bài viết, v.v.) và chuyển nó thành vector embedding để lưu trữ.

**Quy trình:**

* **Data Source**: Là nơi chứa dữ liệu gốc.
* **Data Processing and Chunking**: Chia nhỏ tài liệu thành từng đoạn dễ xử lý (ví dụ: từng đoạn văn/bản ghi).
* **Embedding Model (Voyage AI)**: Dùng mô hình embedding để biến các đoạn text thành vector (số hóa ngữ nghĩa).
* **MongoDB Atlas**: Lưu các vector embeddings vào cơ sở dữ liệu có hỗ trợ tìm kiếm vector – gọi là **Atlas Vector Search**.

## B. Retrieval (Truy xuất dữ liệu liên quan)

- Khi có truy vấn từ người dùng, hệ thống sẽ tìm các đoạn tài liệu liên quan nhất.

**Quy trình:**

* **Query**: Người dùng nhập câu hỏi.
* **Embedding Model**: Chuyển câu hỏi thành vector.
* **Atlas Vector Search**: So khớp vector truy vấn với vector trong MongoDB để tìm ra các **tài liệu gần đúng nhất** (semantically similar).

## C. MongoDB Atlas Vector Search là gì?

* Là tính năng cho phép **lưu trữ và tìm kiếm vector (embedding)** trong MongoDB.
* Hỗ trợ **tìm kiếm theo độ tương đồng ngữ nghĩa**, thay vì tìm chuỗi chính xác như tìm kiếm truyền thống.

# Triển khai dự án

## A.Mô tả dự án

* Cho phép người dùng upload lên file với các định dạng ví dụ như pdf , word, audio , image , …
* Cho phép người dùng có thể tìm kiếm các thông tin liên quan nội dung tương ứng trong các file đã tải lên .

## B.Triển khai

### 1. Back-end

* Công nghệ sử dụng : Spring boot , python , flask , AI model .
* Triển khai dự án gồm 3 service bao gồm:
  + RAG-Service( service chính )
  + OCR-Service
  + Whisper-Service

#### Chi tiết các service

1. RAG-Service
   * + - Xử lý và lưu trữ file đa định dạng (PDF, ảnh, audio, văn bản) bằng cách trích xuất nội dung (OCR, Whisper) và lưu vào MongoDB kèm bản không dấu.
       - Tìm kiếm nội dung văn bản trong các file đã lưu bằng $regex
       - Tích hợp các service khác qua WebClient là Flask OCR và Whisper để xử lý ảnh và âm thanh.
     + Luồng xử lý :

[Client]

|

| 1. POST /upload

v

[UploadController]

|

v

[FileService.processFile(file)]

|

|---► Nếu file là ảnh → extractTextFromImage()

| |

| |--- Gửi ảnh tới: http://localhost:5000/read-text (OCR Flask service)

| |--- Nhận: { ai\_corrected\_text: "..." }

|

|---► Nếu file là audio → transcribeAudio()

| |

| |--- Gửi audio tới: http://localhost:5001/transcribe (Whisper Flask service)

| |--- Nhận: { text: "..." }

|

|---► Nếu file là PDF → extractTextFromPdf()

| |

| |--- Lặp từng trang:

| | └─ render thành ảnh → gửi OCR Flask service

|

|---► Nếu là text → dùng Apache Tika để parse trực tiếp

|

└─> Tiền xử lý: removeVietnameseDiacritics()

|

└─> Lưu vào MongoDB (collection: "chunks")

1. OCR-Service
2. Whisper-service

### 2. Front-end