TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

Tên đề tài:

XÂY DỰNG CÔNG CỤ QUẢN LÝ VÀ TRA CỨU TÀI LIỆU THÔNG MINH

SVTH : Nguyễn Trương Anh Minh. 21TCLC_NHAT1
GVHD: ThS. Trần Hồ Thuỷ Tiên

Đà Nẵng, 2025

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

1. Thông tin chung:

1. Họ và tên sinh viên: Nguyễn Trương Anh Minh
2. Lớp: 21TCLC_NHAT1 Số thẻ SV: 102210040
3. Tên đề tài: Xây dựng công cụ quản lý và tra cứu tài liệu thông minh
4. Người hướng dẫn: ThS. Trần Hồ Thuỷ Tiên Học hàm/ học vị: Thạc sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, khả năng ứng dụng của đề tài: (điểm tối đa là 2đ)

……..

……..

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án: (điểm tối đa là 4đ)

……..

……..

3. Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (điểm tối đa là 2đ)

……..

……..

4. Đề tài có giá trị khoa học/ có bài báo/ giải quyết vấn đề đặt ra của doanh nghiệp hoặc nhà trường: (điểm tối đa là 1đ)

……..

……..

5. Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:

……..

……..

III. Tinh thần, thái độ làm việc của sinh viên: (điểm tối đa 1đ)

……..

IV. Đánh giá:

1. Điểm đánh giá: ……../10 (lấy đến 1 số lẻ thập phân)
2. Đề nghị: ☐ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày tháng năm 2025

Người hướng dẫn

BẢN TÓM TẮT BẰNG TIẾNG NHẬT

知的な文書管理・検索システムの開発

Development of an Intelligent Document Management and Retrieval System

名前：NGUYEN TRUONG ANH MINH（グエン・チュオン・アイン・ミン）

学生番号：102210040

指導教官：TRAN HO THUY TIEN

所属：ダナン大学・工科大学・情報学部

Abstract: Our project introduces an intelligent document management and retrieval system designed to streamline the process of storing and accessing documents from various sources. The system addresses the common issue of document fragmentation by centralizing information into a unified platform, enabling users to retrieve relevant data quickly and efficiently. Integrated with a natural language AI assistant, the system allows users to interact using simple queries without needing to know document structure or location. This assistant leverages AI to interpret user intent and provide accurate, context-aware answers from the stored documents. By reducing search time and simplifying access to internal knowledge, the system improves productivity and supports informed decision-making. This project demonstrates how AI technologies can enhance information search experiences and boost productivity in modern work environments.

1. はじめに

近年、社内の情報や文書が増える中で、必要な情報をすぐに見つけることが難しくなっています。そこで本プロジェクトでは、AI を使った文書管理・検索システムを開発します。

このシステムは、さまざまな文書をまとめて管理し、ユーザーが自然な言葉で簡単に情報を探せるようにします。

本システムを通じて、多くの人が仕事の効率を上げ、より早く正しい情報にたどり着けることを目指します。

2. システム分析、設計

2.1. 機能分析

ユーザー認証機能グループ:

- サインアップ:
  - メールアドレス、ユーザー名、パスワードを入力してアカウントを作成する。
  - 二要素認証（MFA）の有効化・無効化が可能。
- ログイン:
  - Google アカウント（Gmail）を使ったログインに対応。
  - メールアドレスとパスワードによるログインも可能。

個人情報管理機能グループ:

- 個人情報の閲覧。
- 個人情報を更新。
- 二要素認証（MFA）のオン・オフを切り替える。

スペース管理機能グループ:

- 公開スペース関連:
  - 公開スペースの一覧を表示。

- 公開スペースには承認なしですぐに参加可能。

- スペース作成・管理:
  - 新しいスペースを作成（タイトル、説明）。
  - スペース情報の更新。
  - 制限の設定。
  - API キーの作成。
  - スペースの削除。

スペースメンバー管理機能グループ:

- ユーザーをスペースに招待する。
- スペースからメンバーを削除する。
- スペース参加の招待を確認し、「参加」または「拒否」を選択可能。

ドキュメント管理機能グループ:

- スペースにドキュメントをアップロードする。
- スペース内のドキュメント一覧を表示する。
- ドキュメントの詳細を表示する。
- ドキュメントをスペースから削除する。

インタラクション・サポート機能グループ:

- 参加中のスペースの詳細を表示。
- チャットボットを利用して、スペース内のドキュメントに関連する情報を検索。
- クエリ履歴を表示・再利用可能。

システム機能グループ:

- システムからログアウトする。

2.2. ユースケース図

図 1　概要のユースケース図

2.3. データベース図

図 2　データベース図

3. 実験と結果

3.1. 実験環境

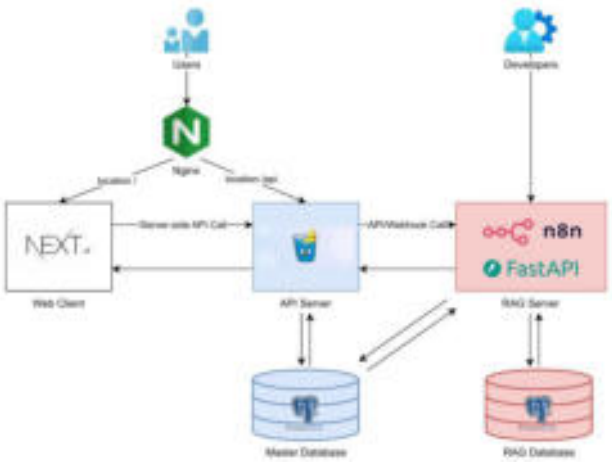


図 3　システム概要

- データベース: PostgreSQL
- 開発ツール: Visual Studio Code, Postman
- バックエンド開発言語: Go, Python
- バックエンドフレームワーク:
  Gin(Go で開発された API サーバー)
  FastAPI(Python で開発され LightRAG API サーバー)
- フロントエンド開発言語: TypeScript
- フロントエンドフレームワーク: NextJs
- AI 開発言語:
  Python(LightRAG), JavaScript (n8n)
- ソースコード管理: Git, Github。

3.2. 結果

ダッシュボード画面

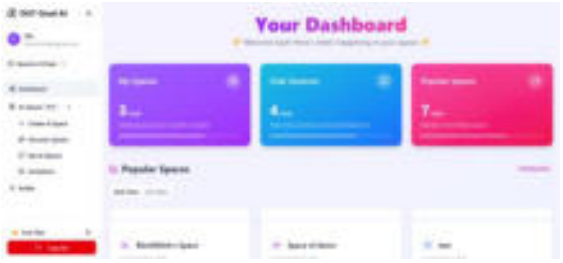


図 4 ダッシュボード画面

プロフィール画面

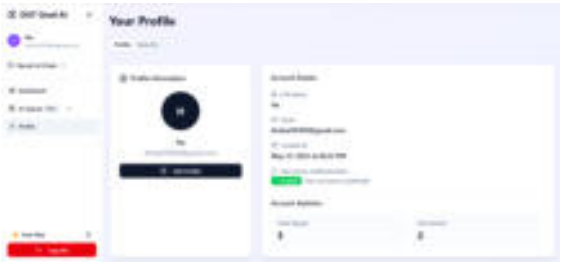


図 5 プロフィール画面

公開スペース一覧画面

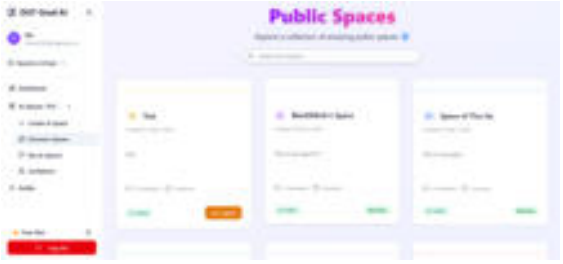


図 6 公開スペース一覧画面

ユーザーのスペース一覧画面

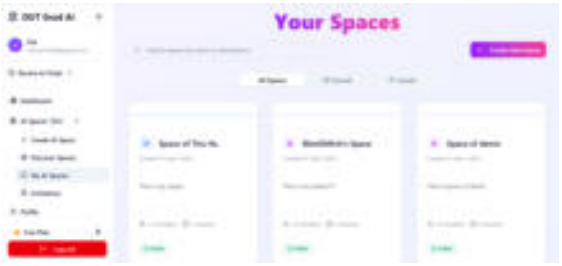


図 7 ユーザーのスペース一覧画面

スペース詳細画面

図 8 スペース詳細画面

スペース設定画面

図 9 スペース設定画面

チャットボット画面

図 10　チャットボット画面

4. 結論と開発の方向性

4.1. 達成結果

研究、理論の学習、プロジェクトの実行過程で、プロジェクトは当初の目標を達成することに成功しました。

理論について:

- 要件の決定、データ構造の設計、システムのデータベースの展開など、データベースの分析、設計、構築に関するスキルを養います。
- Go、Python、TypeScript のプログラミング言語や、Gin、FastAPI、Next.js などのフレームワークを理解し、システムを構築するために適用します。
- AI Agent と RAG（Retrieval-Augmented Generation）に関する知識を理解し、チャットボットによるドキュメント検索機能を実装しました。

申請に関して:

- ウェブサイトは当初提案された機能を実現します。
- システムは、ユーザーがドキュメントを直感的かつ使いやすく管理でき、AI 統合チャットボットを通じて必要な情報を迅速に取得できるようにし、Web ブラウザさえあればいつでもどこでもアクセス可能です。
- ユーザーインターフェースはシンプルで使いやすく、スペースごとにドキュメントを管理できます。

4.2. 欠点

- 時間の制約により、管理者向けの機能はまだ実装されていません。
- ユーザーやドキュメントの数が増加すると、チャットボットの検索や応答のパフォーマンスが低下する可能性があります。
- 同じスペース内でのメンバー同士のやり取りやグループでの共同作業をサポートする機能がまだ不十分です。

4.3. 開発の方向性

- AI を活用し、RAG モデルの最適化によってチャットボットの応答精度と速度を向上させ、より快適なドキュメント検索体験を提供します。
- スペース内でのリアルタイムチャットなど、ユーザー同士のやり取りを可能にするコラボレーション機能を強化します。
- Google Drive や OneDrive などの外部サービス、または企業内の管理システムと連携できる API 機能を追加します。

TÓM TẮT

Tên đề tài: Xây dựng công cụ quản lý và tra cứu tài liệu thông minh

Sinh viên thực hiện:

Nguyễn Trương Anh Minh Số thẻ SV: 102210040 Lớp: 21TCLC_NHAT1

Tóm tắt đề tài:

Trong thời đại số hiện nay, việc quản lý và khai thác hiệu quả tài liệu là một yêu cầu thiết yếu trong các tổ chức và doanh nghiệp. Tuy nhiên, tài liệu thường bị phân tán trên nhiều nền tảng khác nhau như Google Drive, Notion, Jira, Confluence,.. gây khó khăn trong việc tìm kiếm, truy xuất và sử dụng thông tin. Phương pháp tìm kiếm truyền thống dựa trên từ khóa thường không đảm bảo độ chính xác, dẫn đến kết quả không liên quan hoặc bỏ sót thông tin quan trọng, làm giảm hiệu quả công việc và gây mất thời gian.

Đề tài này hướng đến việc nghiên cứu và phát triển một hệ thống quản lý tài liệu tập trung với khả năng tra cứu thông minh, giúp người dùng tìm kiếm và khai thác thông tin nhanh chóng, chính xác và hiệu quả. Hệ thống cho phép lưu trữ, tổ chức tài liệu từ nhiều nguồn trên một nền tảng duy nhất, đồng thời tích hợp trợ lý ảo (AI Agent) ứng dụng mô hình RAG (Retrieval-Augmented Generation) để hỗ trợ người dùng truy vấn và tóm tắt thông tin trong tài liệu bằng ngôn ngữ tự nhiên.

Hệ thống được thiết kế theo mô hình phân lớp, hỗ trợ nhiều không gian làm việc (Space) riêng biệt cho từng nhóm người dùng, đảm bảo tính bảo mật, tính sử dụng và khả năng mở rộng. Giải pháp này không chỉ tiết kiệm thời gian tìm kiếm mà còn nâng cao hiệu suất làm việc, mang lại sự tiện lợi và hiệu quả trong quản lý tài liệu cho các cá nhân và tổ chức.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Nguyễn Trương Anh Minh Số thẻ sinh viên: 102210040
Lớp: 21TCLC_NHAT1 Khoa: Công nghệ thông tin Ngành: CNTT Việt-Nhật

1. *Tên đề tài đồ án:*
   Xây dựng công cụ quản lý và tra cứu tài liệu thông minh.
2. *Đề tài thuộc diện:* ☐ *Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*
3. *Các số liệu và dữ liệu ban đầu:*
   Không có
4. *Nội dung các phần thuyết minh và tính toán:*
   Nội dung các phần thuyết minh bao gồm:
   - Tổng quan đề tài: Giới thiệu tổng quan, mục đích, ý nghĩa của đề tài, đối tượng, phạm vi nghiên cứu và những kết quả dự kiến đạt được.
   - Cơ sở lý thuyết: Trình bày lý thuyết về thuật toán, mô hình học máy hoặc AI được áp dụng.
   - Công cụ phát triển: Trình bày những công cụ và nền tảng phát triển được sử dụng.
   - Phân tích thiết kế hệ thống: Phân tích các yêu cầu về chức năng, phi chức năng và triển khai thiết kế hệ thống thông qua các biểu đồ thiết kế hệ thống.
   - Triển khai thực tế: Trình bày môi trường triển khai và kết quả thực tế đạt được.
   - Kết luận và hướng phát triển: Đánh giá kết quả đạt được, chưa đạt được, khó khăn, cách khắc phục và đưa ra những định hướng phát triển trong tương lai.
   - Tài liệu tham khảo: Liệt kê các tài liệu tham khảo sử dụng trong đề tài.
5. *Các bản vẽ, đồ thị (ghi rõ các loại và kích thước bản vẽ):*
   Không có
6. *Họ tên người hướng dẫn:* ThS. Trần Hồ Thuỷ Tiên
7. *Ngày giao nhiệm vụ đồ án:*
8. *Ngày hoàn thành đồ án:*

Đà Nẵng, ngày tháng năm 2025.

Trưởng Bộ môn …….. **Người hướng dẫn**

GVHD: ThS. Trần Hồ Thuỷ Tiên

LỜI NÓI ĐẦU

Đề tài "Xây dựng công cụ quản lý và tra cứu tài liệu thông minh" là kết quả của quá trình học tập, nghiên cứu và ứng dụng công nghệ hiện đại, phù hợp với xu thế phát triển của xã hội trong thời đại số hoá hiện nay. Đây cũng là thành quả dựa trên kiến thức nền tảng vững chắc mà em đã tích luỹ trong quãng thời gian học tập tại Khoa Công nghệ Thông tin - Trường Đại học Bách khoa - Đại học Đà Nẵng, đồng thời đánh dấu cột mốc quan trọng trong hành trình hoàn thiện chương trình đại học và phát triển bản thân trong lĩnh vực công nghệ thông tin.

Em xin chân thành cảm ơn Nhà trường và quý thầy cô trong Khoa Công nghệ Thông tin, những người đã tận tâm giảng dạy, truyền đạt cho em những kiến thức và kỹ năng quý báu làm nền tảng để em thực hiện đề tài này. Chính sự chỉ dẫn tận tình và kiến thức chuyên sâu từ quý thầy cô là nguồn động lực to lớn giúp em vững tin trong quá trình học tập và nghiên cứu.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc đến cô Trần Hồ Thuỷ Tiên, người đã trực tiếp hướng dẫn, hỗ trợ và động viên em trong suốt quá trình thực hiện đề tài. Sự tận tụy và chỉ bảo quý báu của cô đã giúp em vượt qua nhiều khó khăn, hoàn thiện đề tài một cách hiệu quả nhất.

Trong quá trình thực hiện, do thời gian và năng lực còn hạn chế, chắc chắn đề tài không tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý, nhận xét của quý thầy cô và các bạn để đề tài có thể được hoàn thiện tốt hơn trong tương lai.

Cuối cùng, em xin kính chúc quý thầy cô, quý anh chị và các bạn dồi dào sức khỏe, thành công trong công việc và cuộc sống. Em hy vọng Khoa Công nghệ Thông tin sẽ ngày càng phát triển mạnh mẽ và đạt nhiều thành tựu nổi bật trong lĩnh vực giáo dục và nghiên cứu.

Trân trọng!

Sinh viên thực hiện

Nguyễn Trương Anh Minh

LỜI CAM ĐOAN

Em xin cam đoan:

1. Nội dung trong đồ án này em thực hiện dưới sự hướng dẫn trực tiếp của ThS. Trần Hồ Thuỷ Tiên. Các kết quả trong đồ án đều được thực hiện trong quá trình em đang học tập tại Khoa Công Nghệ Thông Tin, trường Đại học Bách Khoa Đà Nẵng.
2. Các tham khảo dùng trong đồ án đều được trích dẫn rõ ràng tên tác giả, tên công trình, thời gian, địa điểm công bố, được kèm với đường dẫn hợp lệ.
3. Nếu có những sao chép không hợp lệ, vi phạm quy chế đào tạo, em xin chịu hoàn toàn trách nhiệm.

Đà Nẵng, ngày……tháng……năm 2025

Sinh viên thực hiện

Nguyễn Trương Anh Minh

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Từ	Viết tắt của	Diễn giải
API	Application Programming Interface	Giao diện lập trình ứng dụng
CSS	Cascading Style Sheets	Ngôn ngữ định dạng phần tử trang
JS	Javascript	Ngôn ngữ lập trình kịch bản
TS	TypeScript	Ngôn ngữ lập trình mở rộng từ Javascript với hệ thống kiểu tĩnh
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
JSON	JavaScript Object Notation	Định dạng dữ liệu dựa trên đối tượng JavaScript
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
DB	Database	Cơ sở dữ liệu
EC2	Amazon Elastic Compute Cloud	Dịch vụ máy chủ ảo của Amazon
AWS	Amazon Web Services	Nền tảng dịch vụ điện toán đám mây của Amazon
S3	Simple Storage Service	Dịch vụ lưu trữ đối tượng đơn giản của Amazon
LLM	Large language model	Mô hình ngôn ngữ lớn
SLM	Small Language model	Mô hình ngôn ngữ nhỏ
RAG	Retrieval-Augmented Generation	Phương pháp tạo sinh tăng cường truy xuất

MỞ ĐẦU

1. Mục đích thực hiện đề tài

Trong bối cảnh chuyển đổi số diễn ra mạnh mẽ và khối lượng thông tin ngày càng gia tăng, việc quản lý và tra cứu tài liệu hiệu quả trở thành một yêu cầu thiết yếu đối với các cá nhân và tổ chức. Tuy nhiên, thực tế cho thấy việc tìm kiếm thông tin trong các kho tài liệu lớn, rời rạc và thiếu tổ chức khoa học đang gây ra nhiều khó khăn, làm giảm hiệu suất làm việc và tiềm ẩn nguy cơ bỏ sót những nội dung quan trọng.

Đề tài "Xây dựng công cụ quản lý và tra cứu tài liệu thông minh" được thực hiện với mục tiêu nghiên cứu và phát triển một hệ thống hỗ trợ người dùng trong việc lưu trữ, tìm kiếm và trích xuất thông tin từ tài liệu một cách nhanh chóng, chính xác và hiệu quả. Cụ thể, đề tài hướng tới việc ứng dụng mô hình ngôn ngữ lớn (Large Language Model - LLM) kết hợp với kỹ thuật Retrieval-Augmented Generation (RAG) để tạo ra một công cụ thông minh có khả năng truy xuất, tổng hợp và sinh câu trả lời dựa trên dữ liệu tài liệu từ nhiều nguồn phân tán và đa dạng.

Mục đích cuối cùng của đề tài là xây dựng một giải pháp quản lý tài liệu hiện đại, giúp tiết kiệm thời gian tra cứu, nâng cao hiệu quả làm việc và đảm bảo người dùng luôn có thể tiếp cận được thông tin quan trọng một cách kịp thời trong môi trường làm việc số hóa hiện nay.

2. Mục tiêu đề tài

Để hiện thực hoá mục đích đã đề ra, đề tài tập trung vào các mục tiêu cụ thể sau:

- Phát triển một hệ thống lưu trữ tài liệu tập trung, hỗ trợ người dùng tổ chức và quản lý tài liệu từ nhiều nguồn khác nhau trên một nền tảng thống nhất.
- Tích hợp chức năng truy vấn thông minh cho phép người dùng tra cứu và khai thác thông tin một cách nhanh chóng, chính xác đồng thời khắc phục những hạn chế của phương pháp tìm kiếm truyền thống.
- Thiết kế và triển khai một trợ lý ảo (ChatBot) thân thiện, hỗ trợ tương tác bằng ngôn ngữ tự nhiên, giúp người dùng dễ dàng truy vấn và nhận được câu trả lời phù hợp mà không cần phải nắm rõ vị trí hay cấu trúc phức tạp của tài liệu.Xây dựng hệ thống theo kiến trúc phân lớp, cho phép người dùng hoặc nhóm người dùng nhập liệu, quản lý tài liệu trong các không gian làm việc (Space) riêng biệt,

đồng thời đảm bảo các yêu cầu về bảo mật, tính linh hoạt và khả năng mở rộng trong tương lai.

3. Đối tượng và phạm vi nghiên cứu

- **Đối tượng nghiên cứu**

Người dùng hoặc nhóm người dùng có nhu cầu tra cứu thông tin cần thiết từ tài liệu một cách nhanh chóng và hiệu quả.

- **Phạm vi nghiên cứu**

Đề tài tập trung nghiên cứu và xây dựng một hệ thống quản lý và tra cứu tài liệu thông minh, bao gồm:

- Hệ thống cơ sở dữ liệu lưu trữ tài liệu, hỗ trợ truy vấn ngữ cảnh thông qua cơ chế vector hoá (vector database).
- Giao diện web cho phép người dùng quản lý tài liệu, tìm kiếm và tương tác với chatbot hỗ trợ hỏi-đáp dựa trên nội dung tài liệu.
- API tích hợp chatbot, giúp người dùng có thể nhúng chatbot vào hệ thống riêng của họ.

- **Phương pháp nghiên cứu**

Để thực hiện đề tài, các phương pháp nghiên cứu chính được áp dụng bao gồm:

- Nghiên cứu lý thuyết: Tìm hiểu các tài liệu liên quan đến Trí tuệ nhân tạo (AI), Xử lý ngôn ngữ tự nhiên (NLP), kiến trúc và nguyên lý hoạt động của Mô hình ngôn ngữ lớn (LLM), kỹ thuật Retrieval-Augmented Generation (RAG).
- Phân tích và xử lý dữ liệu: Thu thập và phân loại tài liệu do người dùng cung cấp; trích xuất các nội dung quan trọng, chuyển đổi dữ liệu sang văn bản và tóm tắt nhằm tăng tính cô đọng và dễ tiếp cận.
- Lưu trữ và truy xuất: Sử dụng Vector Database để lưu trữ các vector biểu diễn ngữ cảnh của từng phần nội dung tài liệu.
- Thực nghiệm với RAG và LightRAG: Triển khai mô hình tích hợp kỹ thuật RAG nhằm truy xuất dữ liệu phù hợp từ Vector Database, kết hợp với LLM để sinh câu trả lời chính xác và phù hợp ngữ cảnh với truy vấn của người dùng.

4. Những kết quả đạt được dự kiến

- **Về mặt lý thuyết**
  - Nắm quy trình phát triển và xây dựng phần mềm theo hướng hiện đại, đặc biệt là các hệ thống tích hợp trí tuệ nhân tạo.

- Vận dụng hiệu quả các kiến thức đã học tại trường và trong quá trình thực tập tốt nghiệp để phân tích, thiết kế và phát triển một hệ thống quản lý và truy vấn tài liệu thông minh.
- Hiểu và áp dụng các bước thiết kế cơ sở dữ liệu, xây dựng API, cũng như thiết kế giao diện người dùng cho nền tảng web tích hợp chatbot.

- **Về mặt ứng dụng**
  - Xây dựng thành công một công cụ quản lý và tra cứu tài liệu thông minh: Hệ thống cho phép người dùng tải lên tài liệu, tự động xử lý và phân tích nội dung và tích hợp chatbot có khả năng trả lời các câu hỏi liên quan đến tài liệu đó.
  - Tăng tính tiện ích và hiệu quả trong việc truy vấn thông tin: Người dùng không cần đọc toàn bộ tài liệu, mà có thể đặt câu hỏi trực tiếp bằng ngôn ngữ tự nhiên để chatbot phản hồi theo ngữ cảnh phù hợp, giúp tiết kiệm thời gian và nâng cao năng suất làm việc.
  - Hỗ trợ khả năng tích hợp vào hệ thống của người dùng: Cung cấp API giúp người dùng tích hợp chatbot tra cứu tài liệu vào nền tảng riêng (như website nội bộ hoặc các hệ thống quản lý tài liệu hiện có).

5. Cấu trúc đồ án

Đồ án bao gồm các nội dung sau:

- **Mở đầu** - Giới thiệu tổng quan, mục đích, mục tiêu, đối tượng, phạm vi nghiên cứu, phương pháp nghiên cứu và những kết quả dự kiến đạt được.
- **Chương 1: Cơ sở lý thuyết** – Trình bày cơ sở lý thuyết của các thuật toán, mô hình học máy hoặc AI được áp dụng trong đề tài.
- **Chương 2**: **Công cụ phát triển** – Trình bày cơ sở lý thuyết của các công cụ được áp dụng trong đề tài.
- **Chương 3: Phân tích thiết kế hệ thống** – Phân tích các yêu cầu về chức năng và phi chức năng của hệ thống, triển khai thiết kế hệ thống thông qua các biểu đồ thiết kế hệ thống.
- **Chương 4: Triển khai thực tế** - Trình bày môi trường triển khai và kết quả thực tế đạt được.
- **Kết luận và hướng phát triển** – Đánh giá kết quả đạt được, chưa đạt được, khó khăn, cách giải quyết và đưa ra những định hướng phát triển thêm trong tương lai.
- **Tài liệu tham khảo** – Liệt kê các tài liệu tham khảo sử dụng trong đề tài.

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về RAG

RAG [1] là một kiến trúc AI mạnh mẽ kết hợp giữa khả năng sinh ngôn ngữ của các Mô hình Ngôn ngữ Lớn (LLM) và một hệ thống truy xuất thông tin bên ngoài. Mục tiêu chính của RAG là nâng cao độ chính xác, tính cập nhật và mức độ đáng tin cậy của các phản hồi do LLM tạo ra, bằng cách cung cấp cho mô hình thông tin từ một nguồn kiến thức ngoài đáng tin cậy.

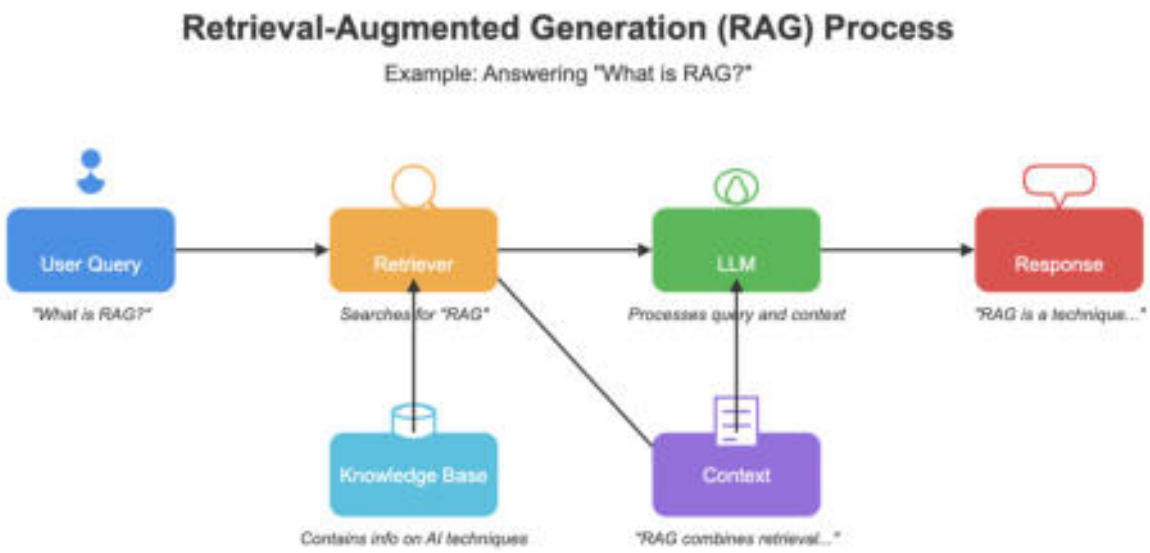
Ưu điểm của RAG (So với LLM truyền thống):

- Hiện tượng "ảo giác" (hallucinations): LLM có thể tạo ra thông tin không chính xác hoặc thậm chí bịa đặt nếu thiếu dữ liệu đáng tin cậy, hoặc bị huấn luyện trên tập dữ liệu không đầy đủ, lệch lạc.
- Kiến thức bị giới hạn và lỗi thời: Các mô hình LLM không thể cập nhật thông tin mới sau thời điểm "cut-off" trong quá trình huấn luyện, dẫn đến các phản hồi lỗi thời.
- Thiếu kiến thức chuyên sâu hoặc nội bộ: LLM không có quyền truy cập vào các tài liệu nội bộ doanh nghiệp, dữ liệu cá nhân, hoặc thông tin chuyên ngành nếu những thông tin đó không được đưa vào tập huấn luyện.

Cơ chế hoạt động của RAG:

Hệ thống RAG hoạt động theo hai giai đoạn chính:

- Giai đoạn Truy xuất (Retrieval Phase): Khi người dùng gửi một truy vấn, truy vấn đó sẽ được chuyển đổi thành một vector nhúng (embedding) bằng một mô hình nhúng (embedding model). Vector truy vấn này sau đó được sử dụng để tìm kiếm các tài liệu (hoặc các đoạn nhỏ - "chunks") liên quan nhất trong một Cơ sở dữ liệu Vector (Vector Database) như Qdrant, FAISS,... Cơ sở dữ liệu này chứa các vector embedding của các tài liệu bên ngoài (ví dụ: tài liệu nội bộ, sách, bài viết,..). Kết quả của giai đoạn này là một tập hợp các đoạn văn bản (context) được đánh giá là phù hợp nhất với truy vấn.
- Giai đoạn Sinh văn bản (Generation Phase): Truy vấn gốc của người dùng và các đoạn văn bản (context) đã được truy xuất ở giai đoạn trước sẽ được ghép nối và đưa vào làm đầu vào cho Mô hình Ngôn ngữ Lớn (LLM). LLM sử dụng thông tin ngữ cảnh mới này để sinh ra phản hồi. Việc có thêm thông tin liên quan giúp LLM trả lời chính xác, đầy đủ và đáng tin cậy hơn, tránh bịa đặt hoặc đưa ra thông tin lỗi thời.



Hình 1.1.1 Retrieval Augmented Generation

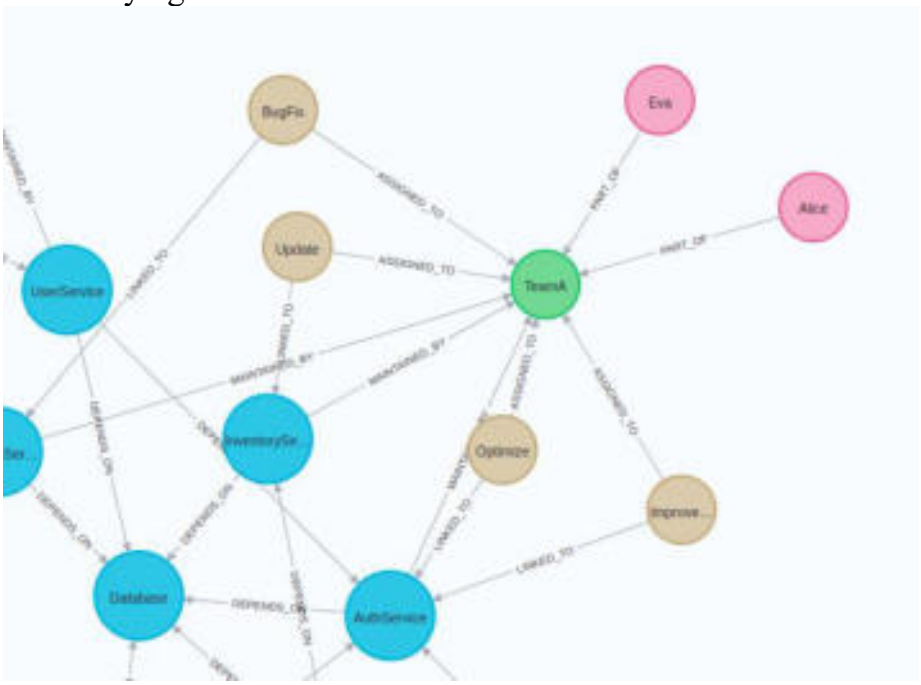
Nhược điểm của RAG:

- RAG gặp khó khăn khi nối kết các điểm thông tin. Điều này xảy ra khi trả lời một câu hỏi đòi hỏi phải đi qua các mảnh thông tin khác nhau, đôi khi thông tin sẽ khá rắc rối, sau đó được tổng hợp lại. Và đôi khi do sự phức tạp hoặc dư thừa từ dữ liệu đưa vào mà LLM trả lời sai.
- RAG hoạt động rất kém khi được yêu cầu hiểu một cách toàn diện một thông tin nào đó trên các bộ sưu tập dữ liệu lớn hoặc thậm chí trên các tài liệu lớn. Ví dụ khi đưa một tập dữ liệu lớn các điều lệ công ty và yêu cầu tóm tắt tất cả điều lệ công ty, thì mô hình RAG thông thường khó có thể trả lời được.

1.2. Tổng quan về GraphRAG

GraphRAG [2] là sự kết hợp giữa đồ thị tri thức (knowledge graph) và hệ thống Retrieval-Augmented Generation (RAG). Khi người dùng đặt những câu hỏi dạng Tóm tắt theo truy vấn (Query-Focused Summarization - QFS), trong đó bối cảnh của truy vấn rất quan trọng, thì RAG truyền thống thường gặp khó khăn trong việc đưa ra câu trả lời đầy đủ. GraphRAG khắc phục điều này bằng cách kết nối các thông tin liên quan từ

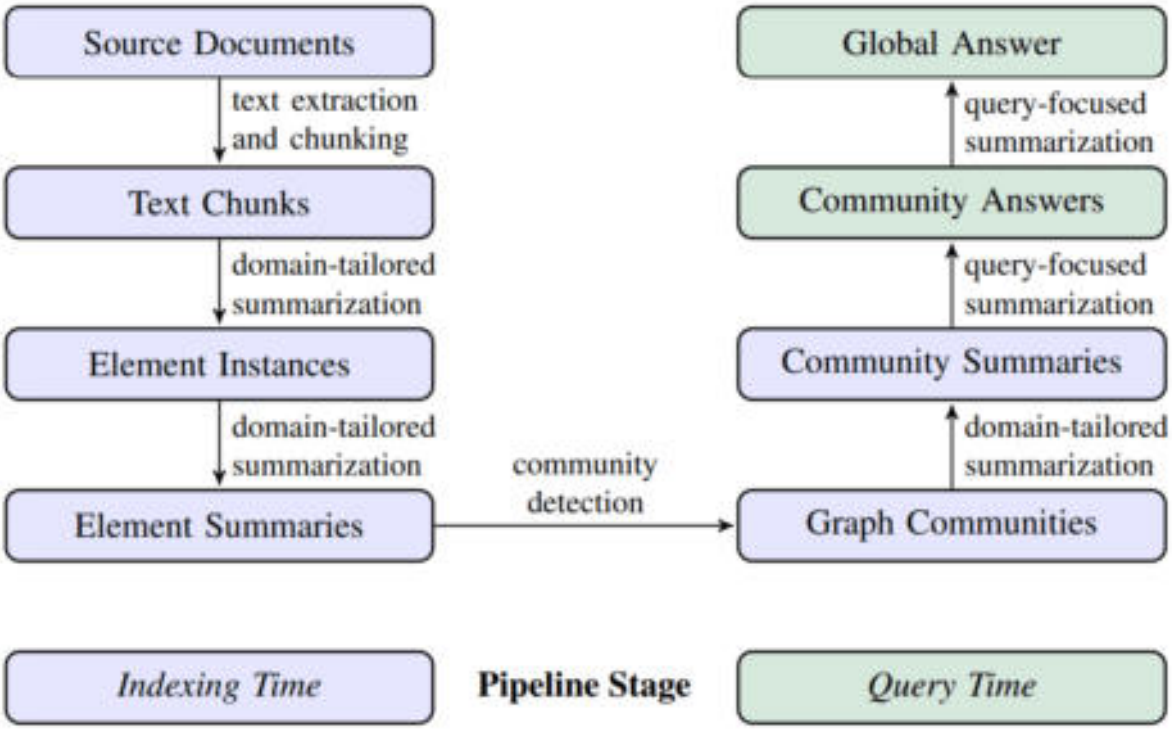
nhiều đoạn và tài liệu khác nhau, từ đó xây dựng một cái nhìn tổng thể hơn để rút ra những hiểu biết có ý nghĩa.



Hình 1.2.1 Knowledge Graph

Thông qua việc sử dụng đồ thị tri thức, các mối quan hệ giữa các thực thể và đối tượng trong dữ liệu được giữ nguyên, giúp mô hình có biểu diễn thông tin phong phú hơn.

Tương tự như RAG, quy trình của GraphRAG cũng bao gồm hai giai đoạn chính: lập chỉ mục (indexing) và truy vấn (querying). Tuy nhiên, trong giai đoạn lập chỉ mục, GraphRAG trích xuất các thực thể và mối quan hệ từ tài liệu thông qua các prompt được thiết kế kỹ và các bước kiểm tra thông tin. Bằng cách sử dụng một LLM (mô hình ngôn ngữ lớn), GraphRAG xác định các thực thể (tên, danh mục, mô tả) và các mối quan hệ (kết nối giữa các thực thể, kèm theo điểm số mức độ liên kết – ví dụ từ 1 đến 10). Trong bước tiếp theo, thông qua clustering ngữ nghĩa (semantic clustering), các nút có liên quan gần được nhóm lại thành các cụm phân cấp, gọi là communities (cộng đồng). Ở giai đoạn truy vấn, khi người dùng đặt câu hỏi, các thực thể và mối quan hệ trong truy vấn được xác định và so sánh với chỉ mục đồ thị, sau đó các cộng đồng liên quan nhất được chọn. Tóm tắt của các cộng đồng này sẽ được đưa vào LLM để tạo phản hồi trung gian ở các mức độ địa phương (local) và tổng thể (global), cùng với điểm số hữu ích (helpfulness score) từ 0 đến 100. Cuối cùng, phản hồi trung gian này được đưa qua một quá trình multi-stage map-reduce để tạo ra câu trả lời cuối cùng toàn diện nhất.



Hình 1.2.2 Quy trình hoạt động của GraphRAG

Ưu điểm của GraphRAG

- Nhờ khả năng nắm bắt mối quan hệ giữa các thực thể dữ liệu, GraphRAG đặc biệt phù hợp để trả lời các truy vấn phức tạp. Đây được gọi là hiểu ngữ cảnh nâng cao (enhanced contextual understanding) – lý do chính khiến đồ thị tri thức được tích hợp thêm vào hệ thống RAG, vốn chỉ có khả năng truy xuất phẳng (flat data retrieval).
- Một lợi ích khác là nâng cao độ chính xác và chất lượng phản hồi, thậm chí có thể cao gấp 3 lần so với RAG truyền thống. Việc tóm tắt ở cả mức độ local và global giúp GraphRAG hiểu rõ hơn ý định người dùng, từ đó đưa ra phản hồi tốt hơn.

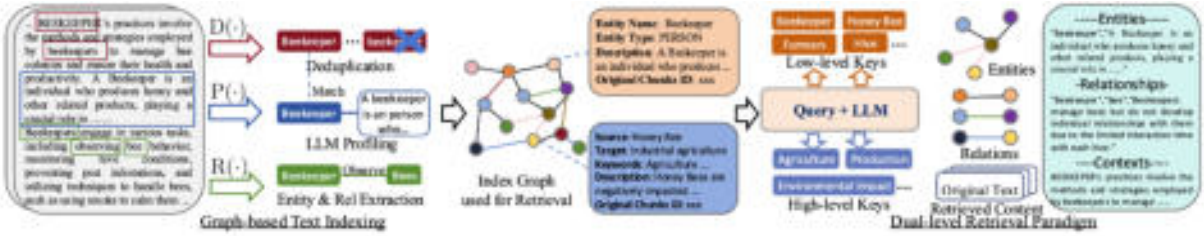
Hạn chế của GraphRAG

- Về mặt hạn chế, chi phí tính toán cao là nhược điểm chính của GraphRAG. Nhiều lần gọi API để xây dựng và truy vấn đồ thị tri thức khiến quá trình này trở nên chậm, dễ bị giới hạn tốc độ (rate limit) và tốn kém. (Ví dụ, xử lý một cuốn sách như A Christmas Carol (~32.000 từ) bằng GPT-4o có thể tốn 6–7 USD).
- Một nhược điểm khác là khi cần thêm dữ liệu mới vào đồ thị hiện có, hệ thống phải tái xây dựng lại toàn bộ đồ thị tri thức cho cả dữ liệu cũ và mới, gây ra tình trạng thiếu hiệu quả nghiêm trọng trong cập nhật.

1.3. Tổng quan về LightRAG: Giải pháp thay thế nhẹ hơn cho GraphRAG

LightRAG [3] là một dự án mã nguồn mở, được khởi xướng vào tháng 10 năm 2024 bởi Đại học Khoa học Dữ liệu Hồng Kông (HKUDS). Mục tiêu chính của LightRAG là khắc phục các nhược điểm của GraphRAG, đặc biệt là chi phí tính toán cao, thông qua các cơ chế:

- Loại bỏ trùng lặp (deduplication)
- Truy xuất hai tầng (dual-level retrieval)
- Cơ chế chia đoạn (chunking) tối ưu hơn



Hình 1.3.1 Tổng quan kiến trúc của LightRAG

LightRAG khác biệt với GraphRAG ở chỗ không sử dụng kỹ thuật gom cụm ngữ nghĩa (semantic clustering). Thay vào đó, LightRAG áp dụng chiến lược phân tích các đoạn thông tin (chunks) thành các thực thể (entities) và từ khóa ở hai tầng – low-level và high-level. Trong giai đoạn truy vấn, mô hình LLM sẽ phân tích câu hỏi của người dùng thành các từ khóa low-level (chi tiết, cụ thể) và high-level (khái quát, chủ đề rộng), sau đó sử dụng các từ khóa này để tìm kiếm thông tin trong đồ thị tri thức một cách chính xác hơn.

Nhờ kiến trúc này, khi cần thêm mới hoặc cập nhật thông tin, LightRAG chỉ đơn giản chèn hoặc điều chỉnh các nút và quan hệ trong đồ thị tri thức mà không cần thực hiện lại quá trình semantic clustering như ở GraphRAG. Điều này giúp giảm thiểu đáng kể chi phí xử lý, thời gian cập nhật và độ phức tạp của hệ thống.

Hơn nữa, với mô hình truy xuất hai tầng, LightRAG có thể đáp ứng hiệu quả cho cả hai loại truy vấn:

- Truy vấn chi tiết (low-level): trả lời dựa trên các thực thể liên quan trực tiếp đến truy vấn.
- Truy vấn toàn cục (high-level): tổng hợp các quan hệ rộng hơn để tạo ra phản hồi có chiều sâu và phù hợp ngữ cảnh.

Nhờ các đặc điểm này, LightRAG không chỉ giữ được độ chính xác và khả năng tổng hợp thông tin như GraphRAG, mà còn đảm bảo hiệu suất cao hơn và chi phí vận hành thấp hơn, đặc biệt phù hợp với các hệ thống sản phẩm cần cập nhật liên tục hoặc triển khai ở quy mô lớn.

Bảng 1.3.1 Bảng so sánh giữa RAG truyền thống, GraphRAG và LightRAG

(nguồn: Learnopencv/lightrag)

	Agriculture		**CS**		**Legal**		**Mix**	
	NaiveRAG	**LightRAG**	NaiveRAG	**LightRAG**	NaiveRAG	**LightRAG**	NaiveRAG	**LightRAG**
Comprehensiveness	32.4%	67.6%	38.4%	61.6%	16.4%	83.6%	38.8%	61.2%
Diversity	23.6%	76.4%	38.0%	62.0%	13.6%	86.4%	32.4%	67.6%
Empowerment	32.4%	67.6%	38.8%	61.2%	16.4%	83.6%	42.8%	57.2%
Overall	32.4%	67.6%	38.8%	61.2%	15.2%	84.8%	40.0%	60.0%
	RQ-RAG	**LightRAG**	RQ-RAG	**LightRAG**	RQ-RAG	**LightRAG**	RQ-RAG	**LightRAG**
Comprehensiveness	31.6%	68.4%	38.8%	61.2%	15.2%	84.8%	39.2%	60.8%
Diversity	29.2%	70.8%	39.2%	60.8%	11.6%	88.4%	30.8%	69.2%
Empowerment	31.6%	68.4%	36.4%	63.6%	15.2%	84.8%	42.4%	57.6%
Overall	32.4%	67.6%	38.0%	62.0%	14.4%	85.6%	40.0%	60.0%
	HyDE	**LightRAG**	HyDE	**LightRAG**	HyDE	**LightRAG**	HyDE	**LightRAG**
Comprehensiveness	26.0%	74.0%	41.6%	58.4%	26.8%	73.2%	40.4%	59.6%
Diversity	24.0%	76.0%	38.8%	61.2%	20.0%	80.0%	32.4%	67.6%
Empowerment	25.2%	74.8%	40.8%	59.2%	26.0%	74.0%	46.0%	54.0%
Overall	24.8%	75.2%	41.6%	58.4%	26.4%	73.6%	42.4%	57.6%
	GraphRAG	**LightRAG**	GraphRAG	**LightRAG**	GraphRAG	**LightRAG**	GraphRAG	**LightRAG**
Comprehensiveness	45.6%	54.4%	48.4%	51.6%	48.4%	51.6%	50.4%	49.6%
Diversity	22.8%	77.2%	40.8%	59.2%	26.4%	73.6%	36.0%	64.0%
Empowerment	41.2%	58.8%	45.2%	54.8%	43.6%	56.4%	50.8%	49.2%
Overall	45.2%	54.8%	48.0%	52.0%	47.2%	52.8%	50.4%	49.6%

Mặc dù đạt được nhiều cải tiến về hiệu suất và chi phí so với GraphRAG, LightRAG vẫn tồn tại một số hạn chế nhất định.

Không giống RAG truyền thống chỉ cần embedding để truy xuất, LightRAG cần dùng LLM để phân tích và sinh ra các keyword ngữ nghĩa (cả high-level và low-level) từ cả truy vấn lẫn tài liệu. Điều này có thể gây tốn kém về token và độ trễ, đặc biệt khi xử lý nhiều tài liệu hoặc truy vấn phức tạp.

1.4. Áp dụng các mô hình RAG vào phân tích tài liệu

Giải quyết bài toán truy vấn tài liệu hiệu quả là một yêu cầu cốt lõi trong các hệ thống lưu trữ và truy xuất thông tin hiện đại. Trong bối cảnh dữ liệu ngày càng đa dạng và phức tạp, việc áp dụng các mô hình RAG (Retrieval-Augmented Generation) phù hợp với từng loại tài liệu đóng vai trò quan trọng nhằm tối ưu hóa độ chính xác, hiệu suất xử lý và chi phí vận hành của hệ thống.

Tài liệu số trong thực tế thường được phân thành ba loại chính: văn bản thuần túy, bảng biểu, và hình ảnh. Mỗi loại dữ liệu mang đặc điểm riêng về cấu trúc và mức độ biểu đạt thông tin, do đó cần các chiến lược xử lý khác nhau.

- **Văn bản thuần túy**

Đối với tài liệu văn bản như điều lệ công ty, hợp đồng, báo cáo chuyên môn..., đây là loại dữ liệu có chiều sâu ngữ nghĩa cao, nhiều tầng thông tin đan xen và mối quan hệ logic phức tạp giữa các đoạn văn bản. Trong trường hợp này, mô hình LightRAG được xem là lựa chọn tối ưu nhất.

Khác với GraphRAG, LightRAG không sử dụng kỹ thuật semantic clustering để nhóm các đoạn thông tin liên quan, mà phân tích trực tiếp các "chunk" văn bản thành

các thực thể (entity) và từ khóa ở hai cấp độ: low-level keyword (chi tiết cụ thể) và high-level keyword (chủ đề tổng quát). Khi truy vấn, LLM sẽ phân tích câu hỏi của người dùng thành các từ khóa tương ứng ở hai tầng, sau đó tìm kiếm trong đồ thị tri thức để thu thập thông tin phù hợp.

Do LightRAG sử dụng cấu trúc entity–keyword đơn giản thay vì cụm ngữ nghĩa và cho phép cập nhật gia tăng (incremental update), nên hệ thống không cần xây dựng lại toàn bộ đồ thị tri thức khi có dữ liệu mới, giúp giảm chi phí đáng kể trong vận hành.

- **Bảng biểu và Hình ảnh**

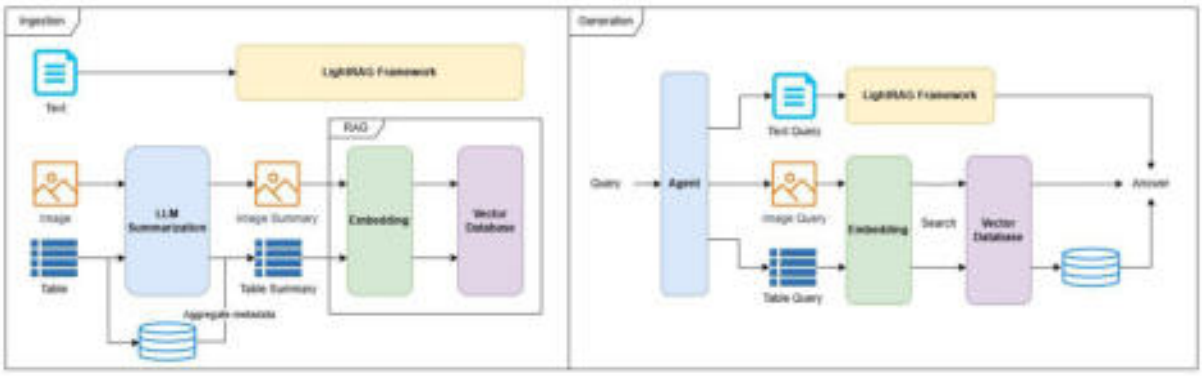
Đối với bảng và hình ảnh, đặc điểm chung là dữ liệu thường ngắn, mang tính tóm tắt, và ít các quan hệ ngữ nghĩa đa chiều như văn bản thuần túy. Ví dụ điển hình gồm: bảng KPI, báo cáo tài chính, ảnh chụp báo cáo kỹ thuật, biểu đồ minh hoạ, …

Thay vì áp dụng các mô hình như GraphRAG hay LightRAG (vốn phát huy hiệu quả khi có nhiều tầng ngữ nghĩa và mối quan hệ phức tạp), hệ thống sẽ sử dụng một mô hình tóm tắt đơn giản là dùng LLM với prompt được thiết kế riêng để tóm tắt nội dung.

Sau khi được tóm tắt, thông tin sẽ được lưu ở dạng đoạn văn bản ngắn, kèm theo các metadata cần thiết (ví dụ: tiêu đề bảng, thời gian, loại hình ảnh, nhãn dữ liệu...). Do các đoạn này ngắn gọn, rõ ràng, không cần truy xuất ngữ nghĩa sâu nên có thể xử lý hiệu quả bằng RAG truyền thống với chi phí tối thiểu.

Ngoài ra, một lưu ý quan trọng là đối với bảng và hình ảnh, nội dung gốc (dưới dạng đoạn tóm tắt đầu tiên) cần được giữ lại và index thẳng, vì quá trình trích xuất entity hay semantic graph như trong GraphRAG/LightRAG sẽ loại bỏ các thông tin này, không đảm bảo khả năng đối chiếu chính xác về sau.

Đối với dữ liệu hình ảnh, quá trình xử lý chỉ cần dừng lại ở việc tạo tóm tắt nội dung hình ảnh và lưu trữ kèm theo đường dẫn tới hình ảnh gốc, nhằm phục vụ việc tham chiếu trực quan khi cần. Tuy nhiên, với dữ liệu bảng biểu, việc chỉ lưu phần tóm tắt là không đủ. Toàn bộ dữ liệu của bảng cần được lưu trữ riêng biệt trong một hệ quản trị cơ sở dữ liệu (chẳng hạn như SQL Database), cho phép hệ thống truy xuất chi tiết từng phần nội dung bảng theo truy vấn cụ thể, đảm bảo tính đầy đủ và chính xác của thông tin được cung cấp.

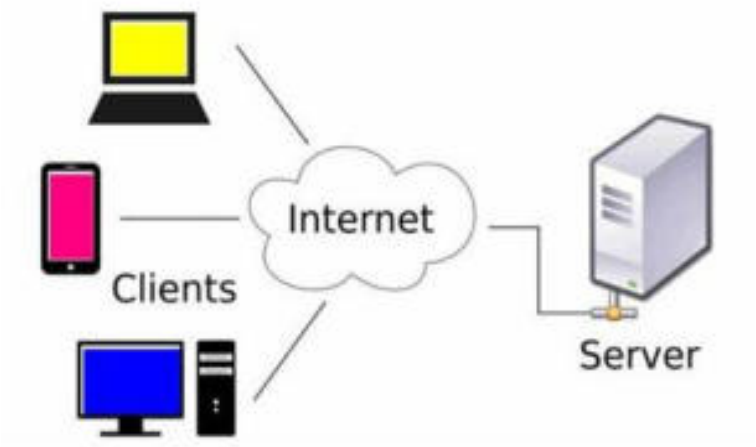


Hình 1.4.1 Tổng quan kiến trúc hệ thống lưu trữ và truy vấn tài liệu

CHƯƠNG 2. CÔNG CỤ PHÁT TRIỂN

2.1. Mô hình Khách–Chủ (Client-Server)

Dự án lựa chọn mô hình kiến trúc client-server để triển khai hệ thống.



Hình 2.1.1 Mô hình Client-Server

Mô hình client-server [4] là một kiến trúc phần mềm phổ biến, trong đó một hoặc nhiều máy khách (client) gửi yêu cầu dịch vụ đến máy chủ (server). Máy chủ là thành phần chịu trách nhiệm xử lý các yêu cầu và cung cấp tài nguyên như cơ sở dữ liệu, tập tin hoặc dịch vụ web. Máy khách là các ứng dụng hoặc thiết bị đầu cuối có nhiệm vụ gửi yêu cầu và sử dụng các dịch vụ được cung cấp bởi máy chủ. Quá trình giao tiếp giữa máy khách và máy chủ thường được thực hiện thông qua mạng nội bộ (LAN) hoặc mạng diện rộng (Internet).

Ưu điểm của mô hình client-server:

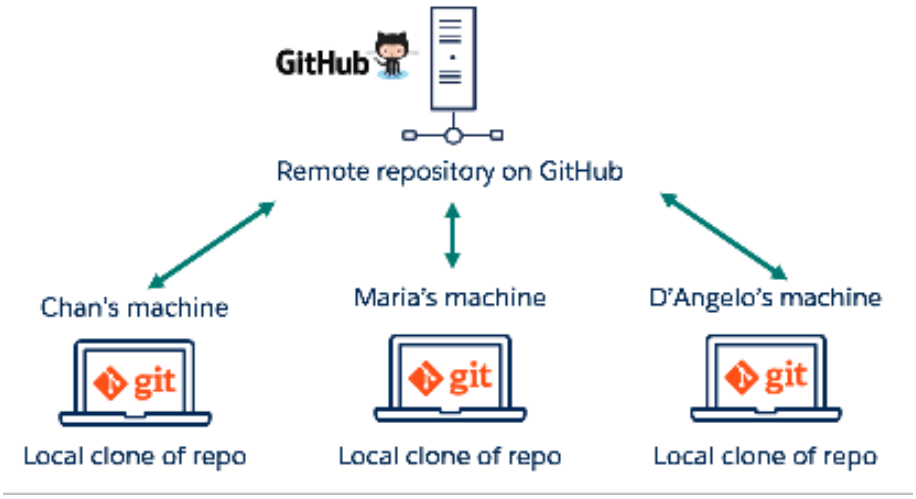
- Quản lý tập trung: Tài nguyên và dữ liệu được quản lý tập trung tại máy chủ, giúp việc kiểm soát, bảo trì và cập nhật hệ thống trở nên thuận tiện và hiệu quả hơn.
- Tăng cường bảo mật: Tài nguyên và dữ liệu được lưu trữ và quản lý tập trung, thuận tiện trong việc thiết lập các biện pháp bảo mật.
- Hiệu suất cao: Máy chủ có thể được cấu hình mạnh mẽ để xử lý nhiều yêu cầu từ máy khách cùng lúc, nâng cao hiệu suất của hệ thống.
- Khả năng mở rộng: Dễ dàng mở rộng hệ thống bằng cách thêm máy khách hoặc nâng cấp máy chủ mà không ảnh hưởng nhiều đến toàn bộ hệ thống.

Nhược điểm của mô hình client-server:

- Phụ thuộc vào máy chủ: Nếu máy chủ gặp sự cố , bị quá tải hoặc bị tấn công, toàn bộ hệ thống sẽ bị ảnh hưởng, gây gián đoạn dịch vụ.
- Chi phí cao: Việc thiết lập, vận hành và duy trì máy chủ cùng các biện pháp bảo mật, sao lưu và quản lý có thể tiêu tốn nhiều chi phí.
- Nguy cơ tắc nghẽn: Khi có quá nhiều máy khách cùng truy cập và gửi yêu cầu trong cùng một thời điểm, có thể dẫn đến tắc nghẽn mạng hoặc giảm hiệu suất xử lý.
- Khó khăn trong việc mở rộng: Trong một số trường hợp, việc mở rộng máy chủ để đáp ứng nhu cầu ngày càng tăng của máy khách có thể gặp khó khăn và đòi hỏi chi phí lớn.

2.2. Quản lý phiên bản và CI/CD

Đề tài sử dụng Git, GitHub và GitHub Actions để quản lý mã nguồn và tự động hóa quy trình phát triển phần mềm. Sau đây là một vài đặc điểm chính của các công nghệ này:



Hình 2.2.1 GitHub

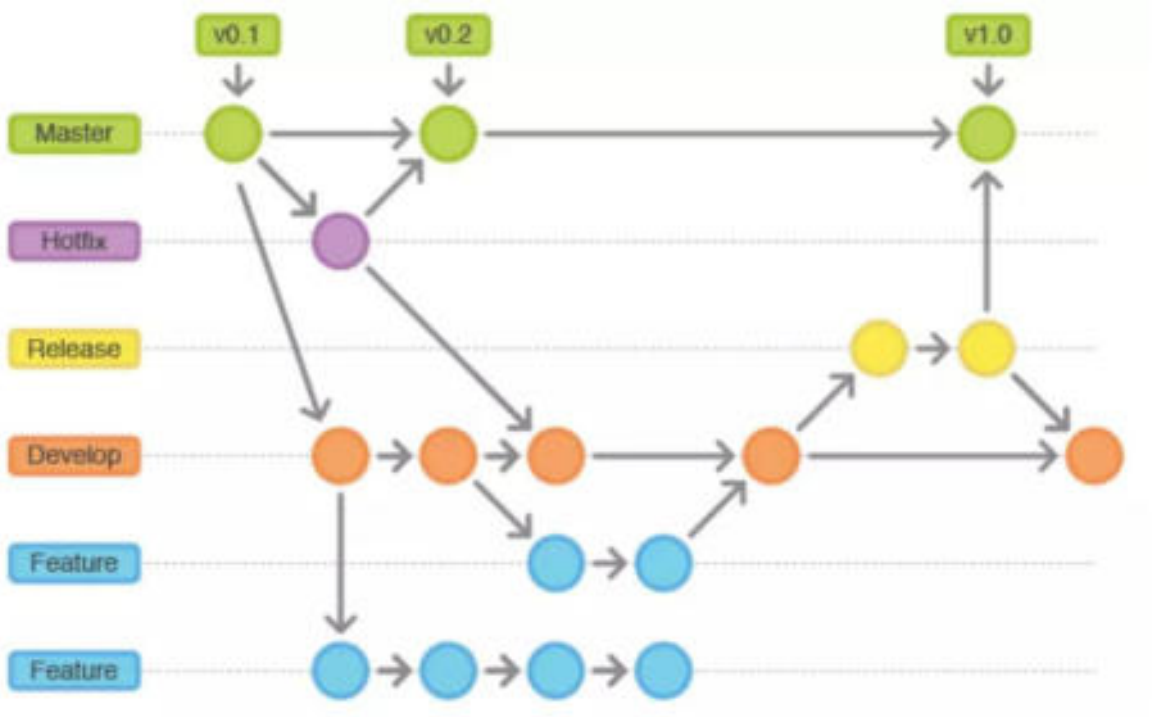
Git [5] là một hệ thống quản lý phiên bản phân tán (DVCS – Distributed Version Control System) được phát triển bởi Linus Torvalds vào năm 2005. Git cho phép các nhà phát triển lưu trữ, theo dõi, và quản lý các phiên bản của mã nguồn một cách linh hoạt, hiệu quả và an toàn. Nhờ khả năng phân tán, mỗi lập trình viên có thể làm việc độc lập trên máy tính của mình, sau đó đồng bộ thay đổi với kho mã chung.

GitHub [6] là một nền tảng lưu trữ mã nguồn dựa trên web, ra mắt vào năm 2008. GitHub cung cấp môi trường quản lý mã nguồn tập trung, hỗ trợ chia sẻ, phối hợp và quản lý dự án phần mềm hiệu quả. Nền tảng này tích hợp nhiều tính năng như issue

tracking, pull request, và wiki, giúp các nhóm phát triển làm việc cùng nhau dễ dàng hơn.

GitHub Actions [7] là công cụ tự động hoá quy trình CI/CD (Continuous Integration / Continuous Deployment) do GitHub phát triển, giúp tự động xây dựng, kiểm thử và triển khai phần mềm khi có thay đổi mã nguồn. Với GitHub Actions, nhóm phát triển có thể thiết lập các workflow để giảm thiểu thao tác thủ công, đảm bảo chất lượng và tốc độ ra mắt sản phẩm.

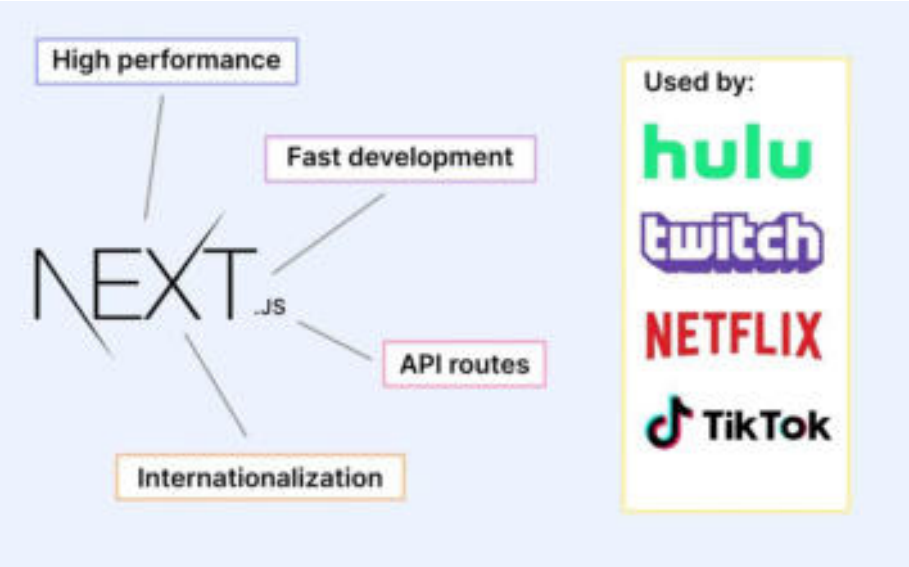
Quy trình làm việc với Git (Git flow):



Hình 2.2.2 Git Flow

Git flow là một mô hình quy trình làm việc phổ biến khi sử dụng Git, được giới thiệu bởi Vincent Driessen. Git Flow giúp quản lý quá trình phát triển phần mềm một cách hệ thống và có tổ chức, đặc biệt hữu ích cho các dự án có chu kỳ phát hành phức tạp.

2.3. Tổng quan về NextJS



Hình 2.3.1 NextJS

NextJS [8] là một framework có mã nguồn mở được xây dựng trên nền tảng của React, được phát triển bởi Vercel và lần đầu tiên ra mắt vào năm 2016. Framework này được thiết kế nhằm giúp lập trình viên dễ dàng xây dựng các ứng dụng web nhanh, tối ưu hóa SEO và có hiệu suất cao.

Framework này nổi bật nhờ khả năng render phía server (SSR - Server-side Rendering), tạo trang tĩnh (SSG - Static Site Generation) và routing tự động, giúp tăng hiệu suất, cải thiện trải nghiệm người dùng và hỗ trợ phát triển nhanh chóng.

Ưu điểm của NextJS:

- Sử dụng SSR và SSG: Giúp cải thiện tốc độ tải trang và khả năng SEO.
- Có nhiều tính năng giúp tối ưu hoá hiệu suất như Code Splitting, Lazy Loading, Image Optimization,...
- Fast Refresh: Tính năng giúp tự động làm mới giao diện mà không cần load lại toàn bộ trang.
- Tự động tạo file CSS dành riêng cho mỗi trang, giúp tránh xung đột trong việc sử dụng và quản lý các file CSS.
- Hỗ trợ TypeScript: NextJS cũng hỗ trợ sử dụng Typescript giúp cải thiện tính rõ ràng cho code và thuận tiện cho việc debug về sau.
- Cộng đồng lớn: NextJS có một cộng đồng sử dụng đông đảo, điều này được chứng minh ở trên chính trang Github của NextJS khi nó đang đạt khoảng hơn 100k sao. Điều này giúp cho NextJS có thêm nhiều nguồn tài liệu phong phú và các plugin hữu ích.

2.4. Tổng quan về TypeScript

Hình 2.4.1 TypeScript

TypeScript [9] là một ngôn ngữ lập trình được phát triển bởi Microsoft, là superset (phần mở rộng) của JavaScript. Điều này có nghĩa là mọi mã JavaScript hợp lệ đều là mã hợp lệ trong TypeScript, nhưng TypeScript bổ sung thêm các tính năng như kiểu dữ liệu tĩnh (static typing) và các công cụ kiểm tra lỗi trong thời gian biên dịch (compile-time error checking).

Đặc điểm nổi bật:

- Hệ thống kiểu tĩnh: Giúp phát hiện lỗi sớm trước khi chạy chương trình, từ đó giảm thiểu lỗi runtime.
- Tự động suy luận kiểu (Type Inference): TypeScript có khả năng suy đoán kiểu của biến nếu không được khai báo rõ ràng.
- Hỗ trợ lập trình hướng đối tượng (OOP): TypeScript cung cấp các tính năng như class, interface, inheritance, access modifier… giúp lập trình viên xây dựng các ứng dụng lớn một cách rõ ràng và có cấu trúc.
- Khả năng tương thích cao với JavaScript: Có thể sử dụng song song với mã JavaScript hiện có.
- Hỗ trợ tốt cho IDE: TypeScript cung cấp khả năng autocomplete, kiểm tra lỗi, refactor thông minh trong các trình soạn thảo như VSCode.

2.5. Tổng quan về Tailwind CSS

Hình 2.5.1 Tailwind CSS

Tailwind CSS [10] là một framework CSS hiện đại theo hướng utility-first, nghĩa là thay vì viết các class CSS riêng cho từng thành phần, lập trình viên sử dụng các class có sẵn để xây dựng giao diện một cách nhanh chóng và hiệu quả.

Đặc điểm nổi bật:

- Utility-first: Cung cấp hàng trăm class như p-4, text-center, bg-blue-500,… giúp bạn tùy biến layout và giao diện trực tiếp trong HTML.
- Tùy biến cao: Dễ dàng cấu hình qua file tailwind.config.js để thay đổi màu sắc, font, breakpoint, v.v.
- Hiệu suất cao: Tailwind sử dụng PurgeCSS để loại bỏ CSS không dùng đến, giúp tối ưu kích thước file khi deploy.
- Phù hợp với thiết kế hiện đại: Cung cấp sẵn các công cụ để xây dựng responsive design, dark mode, hover, focus, animation,...

Ứng dụng:

Tailwind CSS được sử dụng rộng rãi trong nhiều lĩnh vực phát triển giao diện người dùng nhờ sự linh hoạt và hiệu quả:

- Xây dựng giao diện web hiện đại: Dễ dàng tạo ra giao diện đẹp mắt, chuẩn responsive mà không cần viết nhiều CSS tùy chỉnh.
- Phát triển ứng dụng với React/Next.js: Tailwind giúp tăng tốc quá trình thiết kế UI trong các ứng dụng frontend hiện đại bằng cách kết hợp trực tiếp với JSX/TSX.

- Tối ưu hiệu suất khi deploy: Tailwind đi kèm với công cụ build giúp giảm kích thước file CSS chỉ còn vài KB, phù hợp với các dự án cần hiệu suất cao.

2.6. Tổng quan về Go

Hình 2.6.1 Golang

Go [11] là một ngôn ngữ lập trình mã nguồn mở được phát triển bởi Google vào năm 2009. Go được thiết kế để xây dựng các ứng dụng hiệu năng cao, đơn giản và dễ bảo trì. Với cú pháp gần gũi và khả năng xử lý song song mạnh mẽ, Go rất phù hợp cho các dịch vụ backend và hệ thống phân tán.

Các đặc trưng của Go:

- Đơn giản và dễ học: Cú pháp của Go rõ ràng, tránh các phức tạp không cần thiết, giúp lập trình viên nhanh chóng nắm bắt và phát triển ứng dụng.
- Đa luồng (concurrency): Go hỗ trợ song song bằng goroutines rất nhẹ, giúp xử lý nhiều tác vụ đồng thời hiệu quả mà không tốn nhiều tài nguyên.
- Hiệu suất cao: Go được biên dịch thành mã máy chạy nhanh, thích hợp cho các ứng dụng cần tốc độ xử lý lớn.
- Quản lý bộ nhớ tự động: Go có garbage collector giúp quản lý bộ nhớ hiệu quả mà không làm chậm ứng dụng nhiều.
- Hệ thống mô-đun: Go có hệ thống quản lý package và module giúp tổ chức và tái sử dụng mã nguồn dễ dàng.

Ưu điểm của Go:

- Hiệu suất nhanh và ổn định, phù hợp cho các dịch vụ backend có quy mô lớn.
- Dễ dàng viết code song song, xử lý đồng thời các yêu cầu phức tạp.
- Cộng đồng phát triển mạnh, nhiều thư viện và công cụ hỗ trợ.
- Công cụ build, test, và deploy tích hợp sẵn giúp tăng tốc quá trình phát triển.

Với những tính năng ưu việt này, Go đã trở thành một trong những ngôn ngữ lập trình được ưa chuộng trong phát triển các hệ thống backend, microservice và các ứng dụng hiệu năng cao, và ngày càng được sử dụng rộng rãi trong ngành công nghiệp phần mềm.

2.7. Tổng quan về FastAPI

FastAPI [12] là một framework web hiện đại, hiệu suất cao dành cho Python, được thiết kế đặc biệt để xây dựng các API RESTful một cách nhanh chóng, dễ dàng, và hiệu quả. FastAPI được tạo bởi Sebastián Ramírez và lần đầu tiên ra mắt vào năm 2018.

Framework này được xây dựng dựa trên Starlette (cho phần web) và Pydantic (cho phần kiểm tra và phân tích dữ liệu), kết hợp tính năng gõ kiểu (type hinting) của Python 3.6+ để cung cấp mã nguồn rõ ràng, dễ kiểm soát và hạn chế lỗi.

Hình 2.7.1 FastAPI

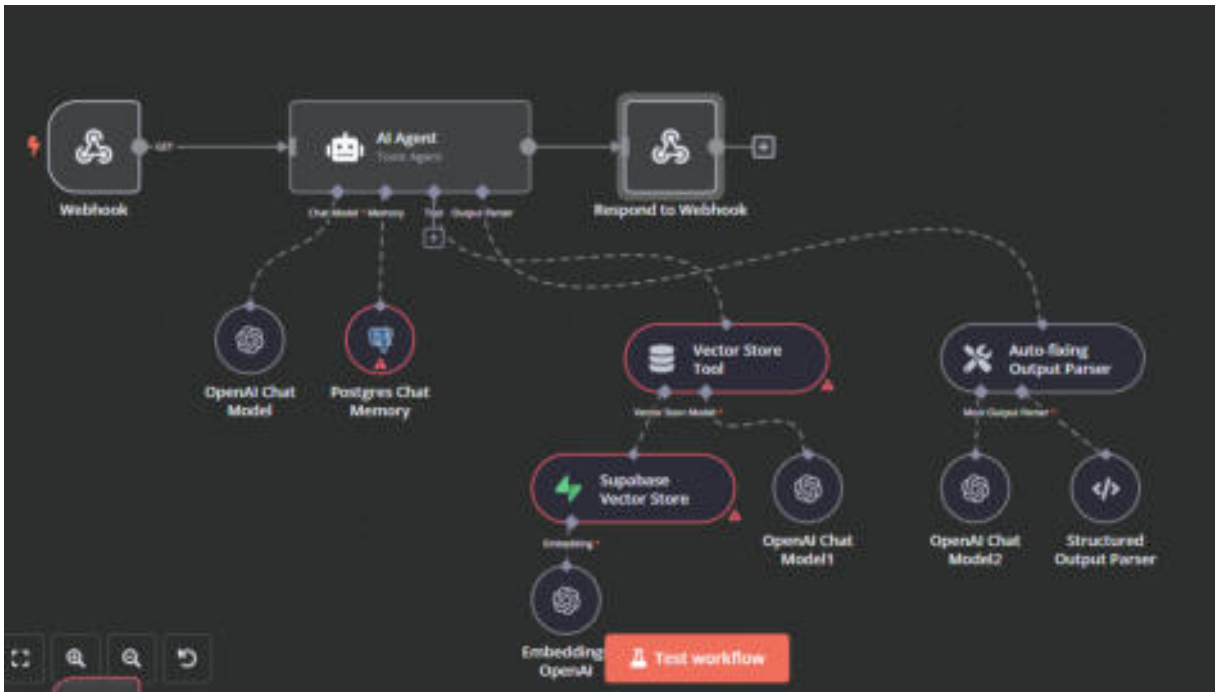
Đặc điểm nổi bật:

- Hiệu năng cao: So với các framework Python khác, FastAPI có tốc độ gần tương đương với Go nhờ vào nền tảng Starlette (async/asgi).
- Tự động sinh tài liệu API: FastAPI tự động tạo tài liệu Swagger UI và ReDoc dựa trên định nghĩa của lập trình viên.
- Hỗ trợ xác thực & phân quyền: Hỗ trợ OAuth2, JWT,... đơn giản hóa việc bảo mật API.
- Tích hợp kiểm tra dữ liệu thông minh: Sử dụng Pydantic giúp kiểm tra, xác thực và mô tả dữ liệu rõ ràng.
- Hỗ trợ async/await: Cho phép viết các endpoint bất đồng bộ dễ dàng.

Ưu điểm:

- Dễ học và dễ mở rộng, phù hợp cả cho người mới học và dự án lớn.
- Tự động sinh tài liệu API giúp giảm thời gian viết tài liệu.
- Cộng đồng phát triển nhanh, có nhiều thư viện hỗ trợ.

2.8. Tổng quan về n8n



Hình 2.8.1 AI Agent trong n8n

n8n [13] là một công cụ tự động hóa quy trình làm việc (workflow automation) mã nguồn mở, được thiết kế để kết nối các dịch vụ và hệ thống khác nhau lại với nhau. n8n cho phép người dùng dễ dàng tạo các workflow phức tạp mà không cần phải viết nhiều mã, tương tự như các công cụ như Zapier hoặc Integromat, nhưng với khả năng self-host (tự triển khai), mở rộng linh hoạt và tùy biến cao hơn.

Đặc điểm nổi bật:

- Kết nối đa dịch vụ: Hỗ trợ hơn 300 dịch vụ như Slack, Gmail, Google Sheets, GitHub, MySQL, PostgreSQL,... chỉ qua thao tác kéo thả.
- Tạo workflow dễ dàng: Cung cấp giao diện trực quan dạng low-code/no-code để kéo thả các node và định nghĩa luồng xử lý.
- Tùy biến linh hoạt: Có thể viết mã JavaScript trong mỗi node để xử lý logic nghiệp vụ nâng cao.
- Tích hợp dữ liệu: Kết nối và truyền dữ liệu qua lại giữa các hệ thống nội bộ hoặc dịch vụ đám mây.
- Self-hosted: Có thể triển khai trên server riêng, đảm bảo quyền riêng tư và bảo mật dữ liệu.
- Open-source: Mã nguồn mở hoàn toàn, dễ mở rộng và tùy chỉnh theo nhu cầu.

Ưu điểm:

- Tự động hóa nhanh chóng các quy trình lặp đi lặp lại như gửi email, cập nhật dữ liệu, thông báo sự kiện,...
- Giao diện kéo-thả dễ dùng, phù hợp cả với người không chuyên lập trình.
- Có thể tích hợp vào hệ thống hiện tại như một phần backend hoặc hỗ trợ các dịch vụ AI, API nội bộ.
- Tiết kiệm chi phí so với các công cụ thương mại như Zapier khi triển khai tại chỗ.

Nhược điểm:

- Một số node tích hợp cần cấu hình phức tạp nếu kết nối đến hệ thống riêng biệt hoặc cần bảo mật cao.
- Với các workflow phức tạp và số lượng lớn node, hiệu suất có thể giảm nếu không tối ưu.

2.9. Tổng quan về PostgreSQL

Hình 2.9.1 PostgreSQL

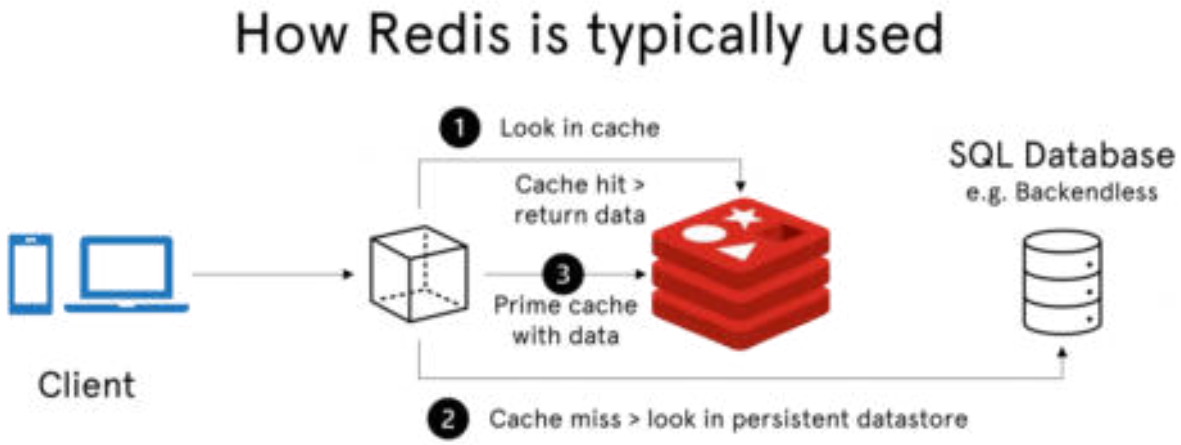
PostgreSQL[14] là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở mạnh mẽ, được phát triển từ năm 1986 tại Đại học California, Berkeley. PostgreSQL nổi bật với tính toàn vẹn dữ liệu cao, hỗ trợ đầy đủ chuẩn SQL và khả năng mở rộng mạnh mẽ, thường được dùng trong các hệ thống quy mô lớn và yêu cầu cao về độ tin cậy.

Đặc điểm nổi bật:

- Mã nguồn mở và miễn phí: PostgreSQL là phần mềm mã nguồn mở, có cộng đồng phát triển lớn và hỗ trợ mạnh mẽ.
- Khả năng mở rộng và hiệu suất cao: Hỗ trợ xử lý một lượng lớn dữ liệu và có thể mở rộng dễ dàng.
- Tính năng phong phú: Hỗ trợ nhiều kiểu dữ liệu, bao gồm cả JSON, XML, và các kiểu dữ liệu địa lý.
- Bảo mật cao: Cung cấp các cơ chế bảo mật tiên tiến như mã hóa SSL, xác thực mạnh mẽ và quản lý quyền truy cập chi tiết.
- Tính toàn vẹn dữ liệu: Hỗ trợ các tính năng như khóa ngoại, trigger, view, và stored procedure, đảm bảo tính toàn vẹn của dữ liệu.

Việc sử dụng PostgreSQL trong hệ thống mang lại nhiều lợi ích về hiệu suất, tính ổn định và bảo mật. Đây là một lựa chọn phù hợp cho các ứng dụng yêu cầu quản lý dữ liệu phức tạp và lớn. PostgreSQL không chỉ đáp ứng được các yêu cầu hiện tại mà còn có khả năng mở rộng và phát triển trong tương lai.

2.10. Tổng quan về Redis



Hình 2.10.1 Redis

Redis [15] là một cơ sở dữ liệu NoSQL dạng key-value in-memory mã nguồn mở, được sử dụng phổ biến như một bộ nhớ đệm (cache), message broker, và hệ thống lưu trữ dữ liệu tạm thời hiệu năng cao. Redis được phát triển lần đầu bởi Salvatore Sanfilippo vào năm 2009.

Một số ưu điểm của redis:

- Hiệu suất cao: Redis lưu trữ dữ liệu trong bộ nhớ, giúp truy xuất dữ liệu cực kỳ nhanh chóng.
- Hỗ trợ đa dạng cấu trúc dữ liệu: Cho phép quản lý dữ liệu phức tạp một cách dễ dàng.
- Đơn giản và dễ sử dụng: Redis có cú pháp đơn giản, dễ triển khai và quản lý.
- Khả năng mở rộng: Hỗ trợ clustering và replication, dễ dàng mở rộng khi hệ thống phát triển.
- Khả năng sử dụng làm cache: Giảm tải cho cơ sở dữ liệu chính, tăng tốc độ truy cập dữ liệu.

Việc sử dụng Redis trong hệ đã mang lại nhiều lợi ích về hiệu suất và tính linh hoạt. Redis không chỉ giúp quản lý phiên làm việc của người dùng một cách hiệu quả mà còn tối ưu hóa việc truy xuất dữ liệu thông qua cơ chế cache. Redis là một giải pháp

mạnh mẽ và linh hoạt, phù hợp cho các hệ thống yêu cầu tốc độ cao và khả năng mở rộng.

2.11. Tổng quan về Qdrant

Hình 2.11.1 Qdrant Vector Database

Qdrant [16] là một cơ sở dữ liệu vector hiệu năng cao (high-performance vector database), được thiết kế đặc biệt để xử lý các bài toán tìm kiếm tương đồng (similarity search) trong các hệ thống sử dụng biểu diễn dữ liệu dưới dạng vector. Đây là một thành phần quan trọng trong các mô hình RAG (Retrieval-Augmented Generation) hiện đại.

Đặc điểm của Qdrant:

- Cơ sở dữ liệu vector: Qdrant lưu trữ các vector đại diện cho dữ liệu phức tạp như văn bản, hình ảnh, âm thanh,... Những vector này được tạo ra từ các mô hình học máy (machine learning) hoặc mô hình ngôn ngữ (language models).
- Tìm kiếm tương tự (Similarity Search): Khi người dùng đưa ra truy vấn, Qdrant sẽ so sánh vector truy vấn với các vector đã lưu trữ để tìm ra các kết quả có độ tương đồng cao nhất, thay vì tìm kiếm chính xác từng từ khóa như trong các cơ sở dữ liệu truyền thống.
- Hiệu suất cao: Qdrant tối ưu cho việc xử lý hàng triệu vector cùng lúc, với tốc độ truy xuất nhanh nhờ các thuật toán indexing tiên tiến và cơ chế lưu trữ in-memory kết hợp với lưu trữ ổ cứng hiệu quả.
- Hỗ trợ clustering và filtering: Qdrant cho phép phân nhóm (clustering) và lọc kết quả theo các điều kiện bổ sung, giúp cải thiện độ chính xác và tính hữu dụng trong các ứng dụng thực tế.

Ứng dụng trong hệ thống RAG:

Trong các hệ thống Retrieval-Augmented Generation (RAG), Qdrant đóng vai trò như một thành phần lưu trữ và truy xuất dữ liệu dựa trên vector embedding của các tài liệu hoặc kiến thức:

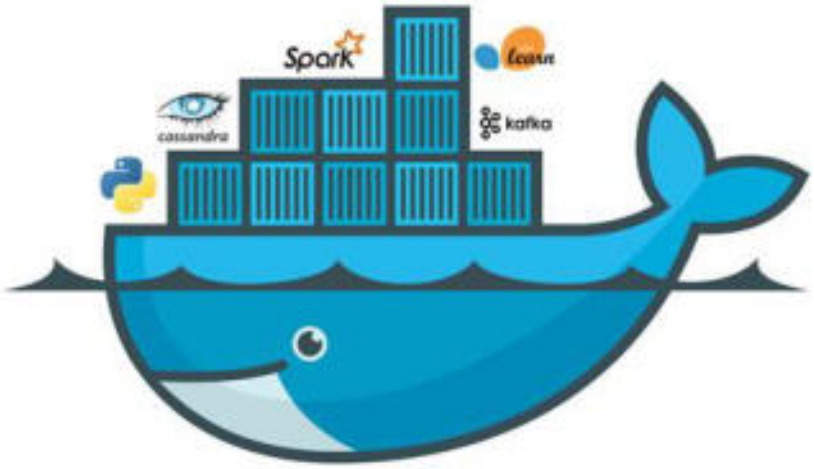
- Tăng cường khả năng truy vấn: Qdrant hỗ trợ tìm kiếm nhanh các tài liệu có nội dung gần giống với câu hỏi hoặc ngữ cảnh do người dùng cung cấp.

- Tích hợp với mô hình sinh ngôn ngữ: Kết quả truy xuất từ Qdrant được sử dụng làm đầu vào để mô hình sinh ngôn ngữ tạo ra câu trả lời hoặc nội dung phù hợp.
- Cải thiện hiệu quả và độ chính xác: Bằng việc kết hợp giữa tìm kiếm tương đống theo vector và mô hình sinh ngôn ngữ, hệ thống có thể cung cấp thông tin chính xác và phong phú hơn so với các phương pháp truy xuất truyền thống.

Ưu điểm của Qdrant:

- Tối ưu hoá cho việc xử lý dữ liệu vector có kích thước và số lượng lớn.
- Dễ dàng tích hợp với các hệ thống AI và Machine Learning hiện đại.
- Hỗ trợ đa nền tảng và cung cấp API RESTful thuận tiện cho việc phát triển ứng dụng.
- Khả năng mở rộng linh hoạt, phù hợp với các hệ thống quy mô lớn.

2.12. Tổng quan về Docker



Hình 2.12.1 Docker

Docker [17] là một nền tảng mã nguồn mở dùng để phát triển, vận chuyển và chạy các ứng dụng. Docker cho phép bạn tách ứng dụng ra khỏi cơ sở hạ tầng, giúp việc phân phối phần mềm trở nên nhanh chóng và linh hoạt hơn. Với Docker, bạn có thể quản lý cơ sở hạ tầng tương tự như cách bạn quản lý mã nguồn của ứng dụng. Bằng cách tận dụng các phương pháp vận chuyển, kiểm thử và triển khai mã mà Docker cung cấp, bạn có thể giảm đáng kể độ trễ giữa giai đoạn phát triển và đưa mã vào môi trường sản xuất.

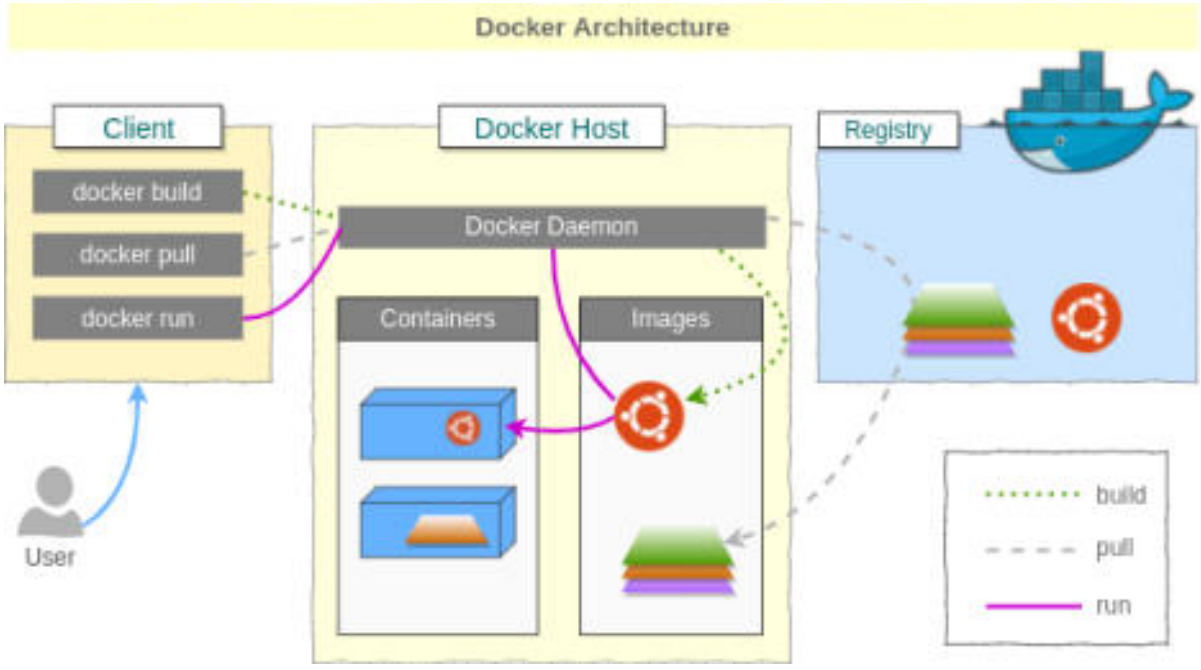
Container (hay vùng chứa) cung cấp môi trường nhẹ, nhất quán bao gồm ứng dụng và toàn bộ các phần phụ thuộc cần thiết để chạy ứng dụng đó

Đặc trưng cơ bản của Docker:

- Consistency - Tính nhất quán: Loại bỏ các vấn đề kiểu "nó chạy được trên máy của tôi" bằng cách đảm bảo môi trường phát triển đồng nhất. Tất cả lập trình viên đều sử dụng cùng một môi trường, giúp dự án chạy ổn định và dễ tái tạo.
- Isolation - Tính cô lập, độc lập: Mỗi container hoạt động độc lập với các ứng dụng khác, giúp ngăn ngừa xung đột phần mềm và đảm bảo môi trường riêng biệt cho từng ứng dụng.
- Portability - Tính di động, linh động: Các container có thể chạy trên bất kỳ hệ thống nào hỗ trợ Docker, bất kể nền tảng cơ sở hạ tầng bên dưới là gì (Windows, macOS, Linux, máy thật hay máy ảo).
- Efficiency - Tính hiệu quả: Các container chia sẻ nhân hệ điều hành máy chủ, giảm chi phí tài nguyên.

So sánh Docker và Virtual Machines:

- Docker container nhẹ hơn so với VM truyền thống.
- VM chạy trên các hệ điều hành guest riêng biệt, trong khi các container chia sẻ với hệ điều hành máy chủ, thông qua docker engine để ảo hóa một OS. Điều này dẫn đến thời gian khởi động nhanh hơn và sử dụng tài nguyên hiệu quả hơn với các container



Hình 2.12.2 Kiến trúc Docker

Việc sử dụng Docker trong hệ thống mang lại nhiều lợi ích về tính nhất quán, hiệu quả tài nguyên và khả năng quản lý. Docker không chỉ đảm bảo ứng dụng chạy mượt mà trong mọi môi trường mà còn giúp dễ dàng triển khai và mở rộng hệ thống. Docker

là một công cụ mạnh mẽ và linh hoạt, phù hợp cho các ứng dụng yêu cầu tính nhất quán và khả năng mở rộng cao.

2.13. Tổng quan về điện toán đám mây AWS

Hình 2.13.1 Amazon Web Services

Amazon Web Services – AWS [18] là một nền tảng dịch vụ điện toán đám mây của Amazon, ra mắt vào năm 2006. AWS cung cấp một loạt các dịch vụ điện toán đám mây bao gồm máy chủ, lưu trữ, cơ sở dữ liệu, mạng, và các dịch vụ ứng dụng khác. AWS được biết đến với quy mô lớn, độ tin cậy cao và sự đa dạng của các dịch vụ.

Các dịch vụ chính:

- Compute: Cung cấp các dịch vụ như EC2 (Elastic Compute Cloud), Lambda (Serverless Computing), và Elastic Beanstalk.
- Storage: Cung cấp các dịch vụ lưu trữ như S3 (Simple Storage Service), EBS (Elastic Block Store), và Glacier.
- Database: Hỗ trợ nhiều loại cơ sở dữ liệu như RDS (Relational Database Service), DynamoDB, và Aurora.
- Networking: Cung cấp các dịch vụ mạng như VPC (Virtual Private Cloud), CloudFront, và Route 53.
- AI và Machine Learning: Cung cấp các dịch vụ AI và Machine Learning như SageMaker, Rekognition, và Comprehend.

Lý do lựa chọn AWS:

- Quy mô và phạm vi: AWS có mặt trên toàn cầu với nhiều trung tâm dữ liệu, cung cấp độ tin cậy và khả năng mở rộng cao.
- Đa dạng dịch vụ: AWS cung cấp một danh mục dịch vụ phong phú và đa dạng, phù hợp với nhiều nhu cầu khác nhau của doanh nghiệp.
- Công nghệ tiên phong: AWS là người tiên phong trong lĩnh vực điện toán đám mây và liên tục cải tiến các dịch vụ của mình để đáp ứng nhu cầu thị trường.

2.13.1. Dịch vụ EC2

Hình 2.13.2 Amazon EC2

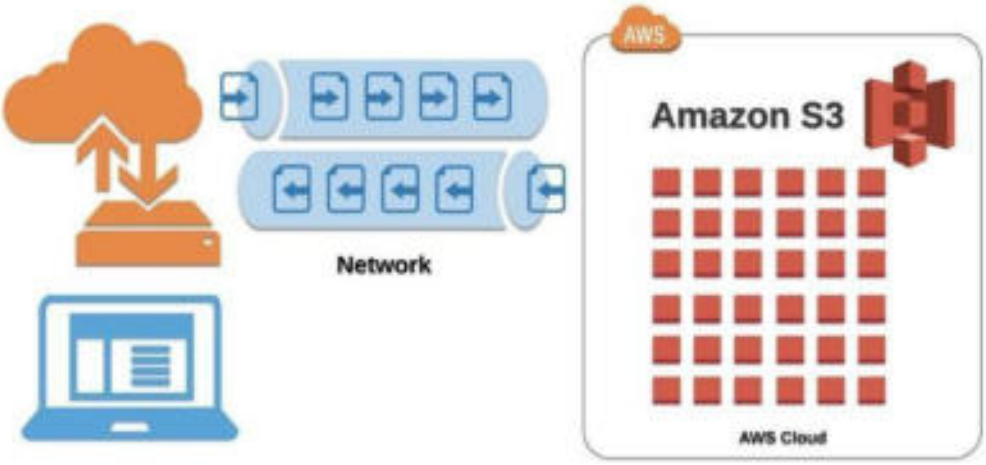
Elastic Compute Cloud – Amazon EC2 [19] là một dịch vụ cung cấp máy chủ ảo trên nền tảng điện toán đám mây do Amazon Web Services phát triển. EC2 cho phép người dùng dễ dàng tạo, cấu hình và vận hành các máy chủ ảo (instances) với nhiều hệ điều hành, cấu hình phần cứng và phần mềm khác nhau tùy theo nhu cầu sử dụng.

Các tính năng chính của EC2:

- Tùy chỉnh linh hoạt: Người dùng có thể lựa chọn loại máy chủ phù hợp với nhu cầu về CPU, bộ nhớ, dung lượng lưu trữ, và mạng.
- Khả năng mở rộng: EC2 hỗ trợ auto-scaling, giúp tự động tăng hoặc giảm số lượng máy chủ khi lưu lượng thay đổi.
- Bảo mật: EC2 hỗ trợ cấu hình firewall thông qua Security Groups và các tính năng bảo mật mạng khác như Virtual Private Cloud (VPC).
- Tích hợp mạnh mẽ: EC2 có thể tích hợp với các dịch vụ khác của AWS như S3, RDS, CloudWatch, IAM,... để xây dựng hệ thống hoàn chỉnh.
- Thanh toán linh hoạt: Người dùng chỉ phải trả phí cho thời gian thực tế sử dụng tài nguyên.

Việc sử dụng EC2 giúp đảm bảo rằng hệ thống có hiệu suất ổn định, dễ mở rộng và có thể tùy chỉnh linh hoạt để phục vụ số lượng người dùng lớn mà vẫn đảm bảo tính bảo mật và hiệu quả vận hành.

2.13.2. AWS S3



Hình 2.13.3 Amazon S3

Simple Storage Service – AWS S3 [20] là một dịch vụ lưu trữ đối tượng (object storage) được cung cấp bởi Amazon Web Services, cho phép người dùng lưu trữ và truy xuất dữ liệu dưới dạng các đối tượng (objects) bên trong các vùng lưu trữ được gọi là buckets.

Các tính năng chính của AWS S3:

- Lưu trữ dữ liệu dưới dạng đối tượng: Dữ liệu được lưu dưới dạng key-value với các đối tượng (objects) được tổ chức trong các bucket. Mỗi object bao gồm dữ liệu, metadata và một khóa định danh duy nhất.
- Độ bền và sẵn sàng cao: Amazon S3 được thiết kế để đảm bảo độ bền dữ liệu rất cao, với nhiều lớp bảo vệ và sao lưu dữ liệu, giúp giảm nguy cơ mất mát. Dịch vụ cũng cung cấp khả năng sẵn sàng gần như liên tục, đáp ứng nhu cầu truy cập dữ liệu ổn định của người dùng.
- Quản lý truy cập: Hỗ trợ phân quyền truy cập chi tiết qua IAM (Identity and Access Management), bucket policies và ACL (Access Control List).
- Hỗ trợ các tính năng nâng cao: Bao gồm versioning (phiên bản hóa dữ liệu), lifecycle policies (chính sách vòng đời), encryption (mã hóa dữ liệu), và sự kiện (event trigger).
- Dễ dàng tích hợp: S3 dễ dàng tích hợp với các dịch vụ khác như EC2, Lambda, CloudFront và các công cụ CI/CD để tạo hệ thống lưu trữ động và linh hoạt.

Nhờ vào khả năng mở rộng và tính ổn định cao, S3 là giải pháp lý tưởng để lưu trữ dữ liệu tĩnh trong các hệ thống hiện đại.

2.13.3. AWS Fargate

Hình 2.13.4 AWS Fargate

AWS Fargate [21] là một dịch vụ tính toán không máy chủ (serverless) của Amazon Web Services, được thiết kế để chạy container mà không cần quản lý máy chủ hoặc cụm máy chủ (cluster). Với Fargate, người dùng chỉ cần định nghĩa các yêu cầu về tài nguyên cần thiết cho container như CPU và bộ nhớ, còn toàn bộ quá trình triển khai, mở rộng và quản lý hạ tầng vật lý do AWS tự động xử lý.

Fargate giúp đơn giản hóa việc vận hành các ứng dụng container, cho phép các nhóm phát triển tập trung vào xây dựng ứng dụng thay vì phải bận tâm đến việc cấu hình và duy trì cơ sở hạ tầng.

Ưu điểm của AWS Fargate:

- Không cần quản lý máy chủ: Giảm thiểu công việc vận hành. Fargate tự động triển khai, mở rộng và phân phối tài nguyên cho container.
- Tự động mở rộng: Tài nguyên được cấp phát linh hoạt và tự động theo nhu cầu thực tế, giúp ứng dụng vận hành hiệu quả hơn.
- Bảo mật cao: Mỗi container chạy trong môi trường cô lập, dễ dàng kiểm soát quyền truy cập giúp tăng cường bảo vệ ứng dụng.
- Tích hợp tốt với các dịch vụ AWS: Hoạt động mượt mà cùng các dịch vụ khác như Amazon ECS, EKS, IAM, CloudWatch.

AWS Fargate rất phù hợp cho các ứng dụng hiện đại sử dụng kiến trúc microservices và cần triển khai nhanh, dễ dàng mở rộng mà không phải quản lý hạ tầng phức tạp.

2.14. Tổng quan về Google Cloud Platform (GCP) và GCP VM (Compute Engine)

Hình 2.14.1 Google Cloud Platform

Google Cloud Platform – GCP [22] là một nền tảng điện toán đám mây do Google phát triển và cung cấp. GCP hỗ trợ các nhà phát triển và doanh nghiệp trong việc xây dựng, vận hành và quản lý ứng dụng, hệ thống với khả năng mở rộng linh hoạt và độ tin cậy cao trên nền tảng đám mây.

GCP cung cấp đa dạng các dịch vụ như lưu trữ dữ liệu, máy chủ ảo, cơ sở dữ liệu, Trí tuệ nhân tạo và học máy (AI/ML), Xử lý dữ liệu lớn (Big Data), và nhiều công cụ hỗ trợ phát triển ứng dụng hiện đại.

Hình 2.14.2 Google Compute Engine

Google Compute Engine [23] là dịch vụ cung cấp máy ảo (Virtual Machine - VM) trên nền tảng đám mây của Google. Người dùng có thể dễ dàng tạo và quản lý các máy ảo với cấu hình tùy chỉnh, phục vụ cho nhiều mục đích như chạy ứng dụng, dịch vụ hoặc môi trường phát triển.

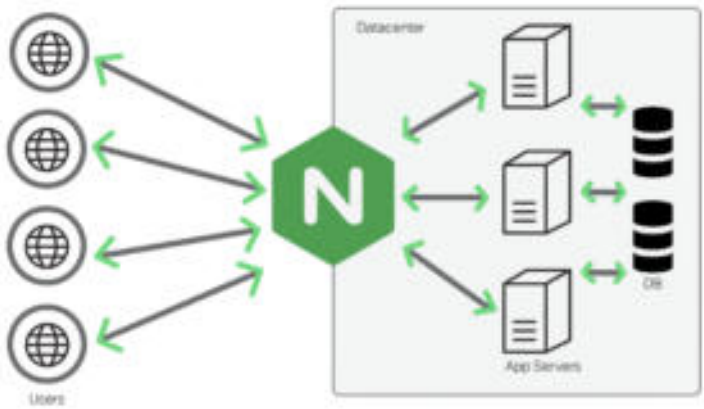
Các máy ảo trên Compute Engine có thể chạy nhiều hệ điều hành khác nhau như Linux, Windows, và được tích hợp nhiều tính năng như auto-scaling, load balancing, và bảo mật cao.

Ưu điểm của GCP Compute Engine:

- Khả năng tùy biến cấu hình máy ảo gồm CPU, RAM, ổ đĩa theo nhu cầu thực tế.
- Sử dụng mạng lưới toàn cầu của Google giúp giảm độ trễ và tăng tốc độ truy cập.
- Hỗ trợ tính năng tự động mở rộng (auto-scaling) và cân bằng tải (load balancing).
- Tích hợp các cơ chế bảo mật và quản lý truy cập dễ dàng.
- Thanh toán linh hoạt dựa theo thời gian sử dụng thực tế (pay-as-you-go).

2.15. Tổng quan về Nginx

Nginx [24] là một máy chủ web mã nguồn mở phổ biến, nổi bật với khả năng xử lý đồng thời số lượng lớn kết nối một cách hiệu quả. Ngoài chức năng chính là máy chủ HTTP, Nginx còn được sử dụng rộng rãi làm reverse proxy, load balancer, và API gateway trong các hệ thống phân tán và hiện đại như microservices.



Hình 2.15.1 Nginx

Đặc điểm chính của Nginx:

- Web Server (HTTP Server): Phục vụ các nội dung tĩnh như HTML, CSS, JavaScript, hình ảnh…
- Reverse Proxy: Làm trung gian chuyển tiếp request từ client đến server backend như Node.js, Java Spring, Python Flask, v.v.
- Load Balancer: Phân phối yêu cầu truy cập đến nhiều server backend để tăng hiệu năng và độ sẵn sàng.
- API Gateway nhẹ: Nginx có thể điều phối truy cập đến các dịch vụ khác nhau thông qua cấu hình routing.

Ưu điểm nổi bật

- Hiệu năng cao: Xử lý hàng nghìn kết nối đồng thời nhờ mô hình sự kiện bất đồng bộ (asynchronous event-driven).
- Tài nguyên nhẹ: Tiêu tốn rất ít CPU và RAM so với các máy chủ web truyền thống như Apache.

- Cấu hình linh hoạt: Dễ dàng cài đặt các rule cho routing, bảo mật, giới hạn băng thông, cache…
- Khả năng mở rộng: Phù hợp cho cả hệ thống nhỏ lẫn hạ tầng cloud quy mô lớn.

CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.1. Các tác nhân chính của hệ thống

Hệ thống gồm tác nhân chính là:

- Khách: Là người chưa đăng nhập vào hệ thống. Khách có thể thực hiện các thao tác như đăng ký, đăng nhập và đăng xuất.
- Người dùng: Là người đã đăng nhập. Người dùng có thể tạo Space mới, xem danh sách các Space công khai, tham gia các Space công khai và quản lý lời mời tham gia Space. Ngoài ra, người dùng có thể đăng xuất khỏi hệ thống.\
- Thành viên Space: Là người đã tham gia vào một Space cụ thể. Tuỳ vào vai trò (role), họ có thể quản lý tài liệu, sử dụng chatbot trong Space, quản lý thành viên (mời/xoá thành viên), và thực hiện các chức năng quản trị Space như cập nhật thông tin không gian, thiết lập giới hạn, tạo API key, hoặc xoá Space.

3.2. Các chức năng của hệ thống

Nhóm chức năng xác thực người dùng

- Đăng ký:
  - Đăng ký bằng email, username, password.
- Bật/tắt xác thực hai yếu tố (MFA).
- Đăng nhập:
  - Đăng nhập bằng Gmail (Google Sign-in).
  - Đăng nhập bằng email và mật khẩu.

Nhóm chức năng quản lý thông tin cá nhân:

- Xem chi tiết thông tin cá nhân.
- Cập nhật thông tin cá nhân.
- Bật/tắt xác thực hai yếu tố (MFA).

Nhóm chức năng quản lý space:

- Space công khai:
  - Xem danh sách các space công khai.
  - Tham gia space công khai (không cần phê duyệt).
- Tạo và quản lý space cá nhân
  - Tạo mới một space.
  - Cập nhật thông tin space.

- Thiết lập giới hạn
- Tạo Api key
- Xoá space

Nhóm chức năng quản lý thành viên Space:

- Mời người dùng vào space.
- Xoá thành viên khỏi space.
- Xác nhận và phản hồi lời mời tham gia space.

Nhóm chức năng quản lý tài liệu:

- Tải lên tài liệu vào space.
- Xem danh sách tài liệu trong space.
- Xem chi tiết từng tài liệu.
- Xoá tài liệu khỏi space.

Nhóm chức năng tương tác và hỗ trợ:

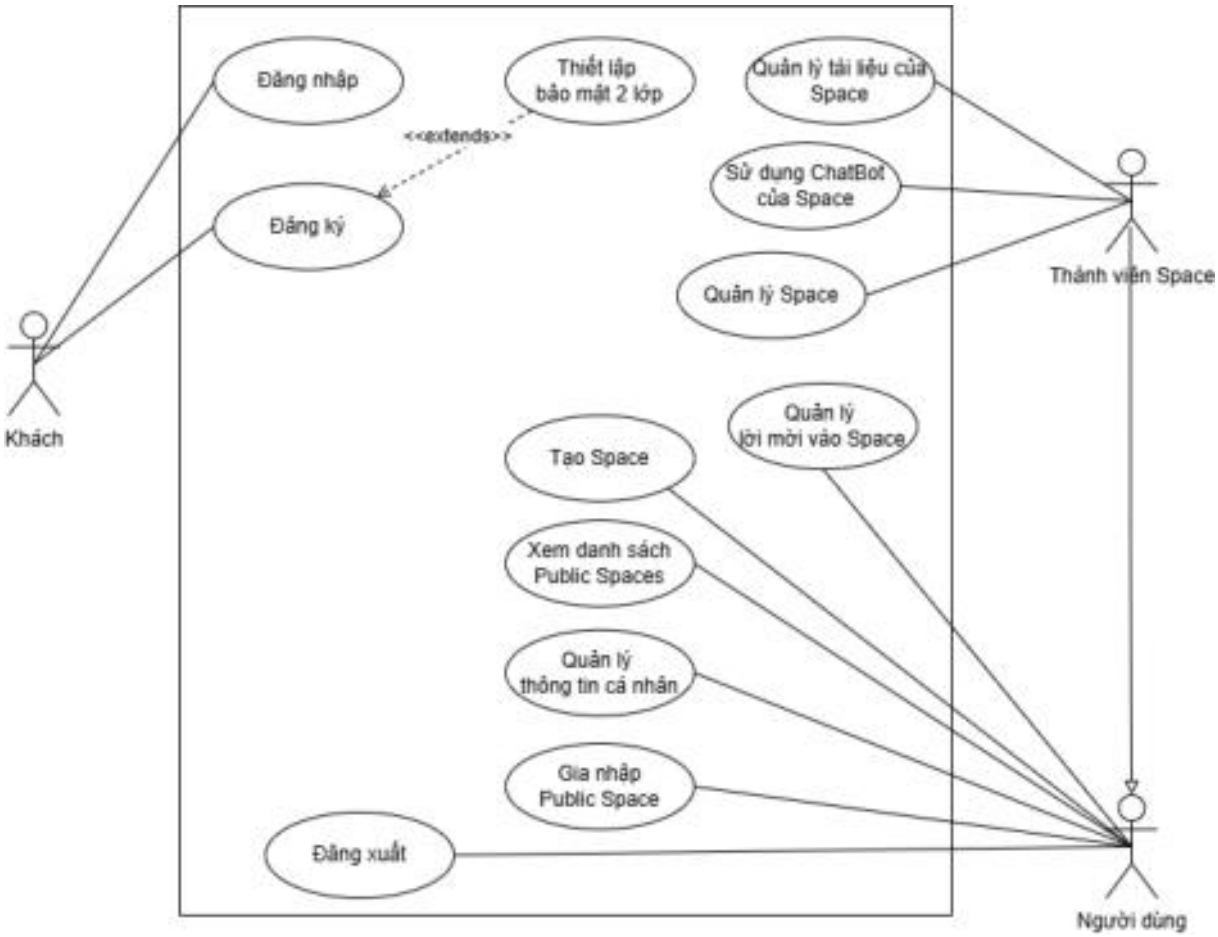
- Xem chi tiết space đang tham gia.
- Sử dụng chatbot để tìm kiếm thông tin liên quan đến tài liệu trong space.
- Xem và tra cứu lại lịch sử truy vấn.

Nhóm chức năng hệ thống:

- Đăng xuất khỏi hệ thống

3.3. Biểu đồ usecase

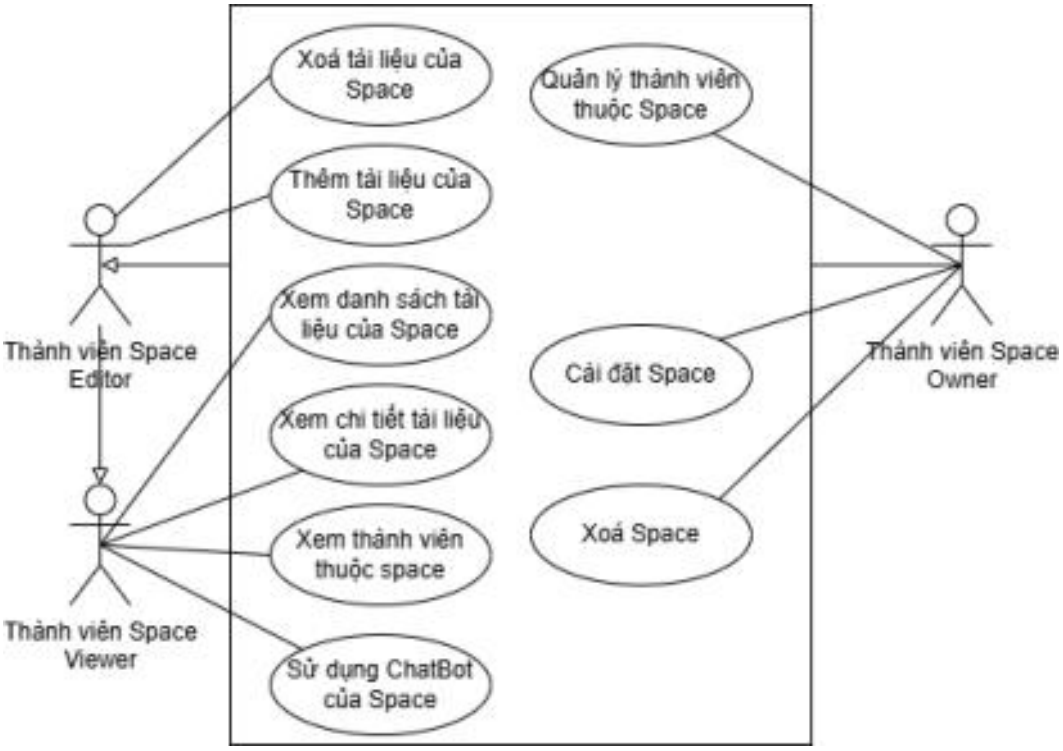
3.3.1. Biểu đồ usecase tổng quan



Hình 3.3.1 Biểu đồ usecase tổng quan.

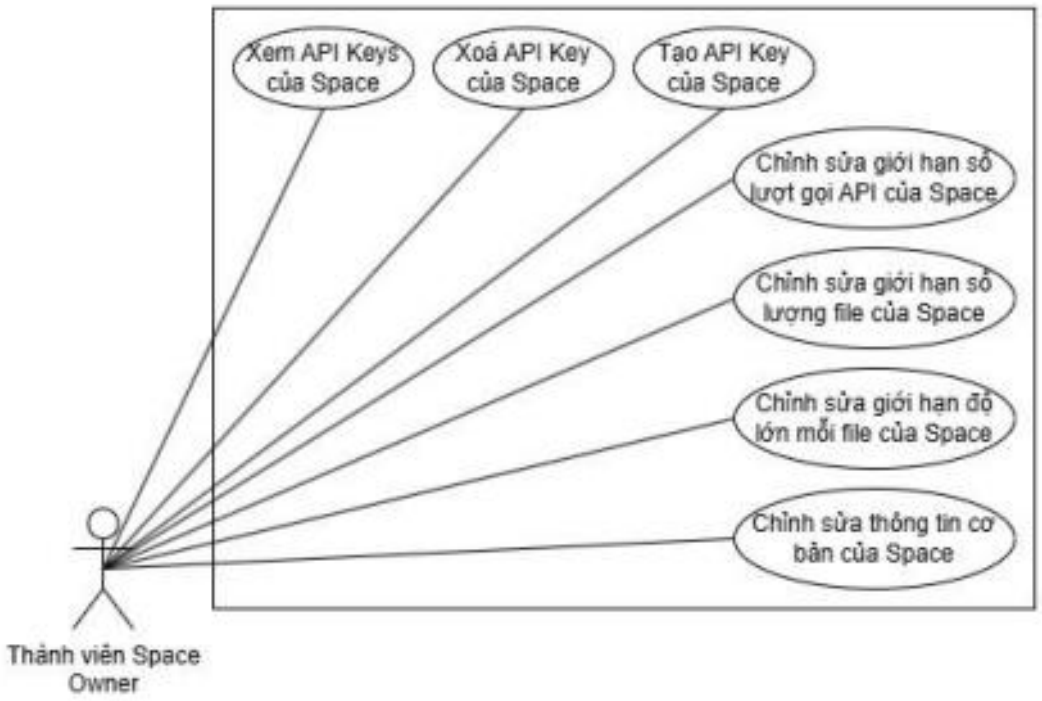
Usecase tổng quan mô tả các chức năng chính của hệ thống với ba vai trò chính: Khách (Guest), Người dùng (User) và Thành viên Space (Space Member). Khách là người chưa có tài khoản, có thể thực hiện các chức năng như đăng ký tài khoản và đăng nhập vào hệ thống. Trong quá trình đăng ký, người dùng có thể lựa chọn thiết lập bảo mật hai lớp như một tính năng mở rộng để tăng cường an toàn cho tài khoản. Sau khi đăng nhập, người dùng có thêm quyền truy cập vào các chức năng nâng cao hơn như tạo mới một Space, xem danh sách các Public Space, quản lý thông tin cá nhân, gia nhập vào các Public Space và quản lý các lời mời tham gia từ các Space khác. Đối với những người dùng đã là thành viên của một Space, họ sẽ có thêm các quyền quản trị như sử dụng ChatBot hỗ trợ trong Space, quản lý nội dung và tài liệu liên quan, cũng như điều hành hoạt động chung trong Space đó. Việc phân chia vai trò và chức năng này giúp hệ thống đảm bảo tính bảo mật, phân quyền hợp lý và nâng cao trải nghiệm người dùng.

3.3.2. Biểu đồ usecase phân rã



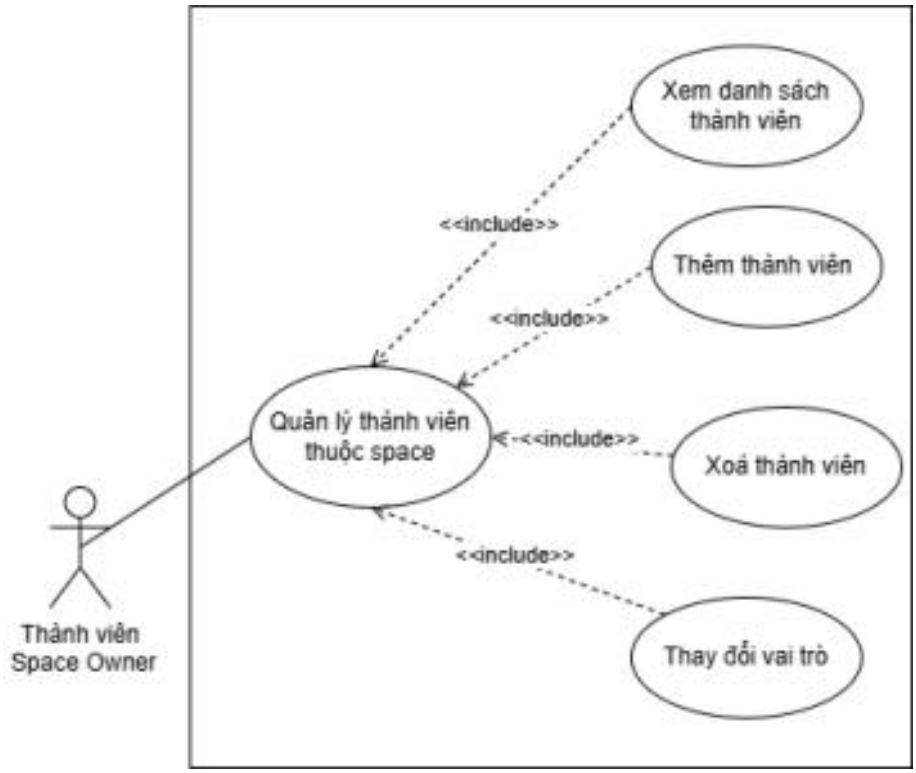
Hình 3.3.2 Biểu đồ Usecase phân rã chức năng theo vai trò trong một Space

Use case mô tả các chức năng tương ứng với từng vai trò thành viên trong một Space: Viewer, Editor và Owner. Thành viên Viewer là người có quyền hạn cơ bản nhất trong hệ thống, chỉ có thể xem danh sách và xem chi tiết tài liệu trong Space, xem thành viên thuộc space và sử dụng ChatBot hỗ trợ. Đây là vai trò dành cho người dùng chỉ truy cập nội dung, không có quyền chỉnh sửa hay quản lý. Thành viên Editor có mức quyền cao hơn, kế thừa toàn bộ chức năng của Viewer, đồng thời có thêm quyền thêm và xóa tài liệu trong Space. Chủ sở hữu Space (Owner) là người có toàn quyền trong không gian làm việc. Ngoài các chức năng của Editor, Owner còn được quản lý người dùng, quản lý thành viên, thiết lập cài đặt, xoá tài liệu và xóa space. Vai trò này thường do người tạo Space hoặc quản trị viên đảm nhiệm. Việc phân quyền rõ ràng giúp đảm bảo kiểm soát hiệu quả hoạt động và nội dung trong từng Space.



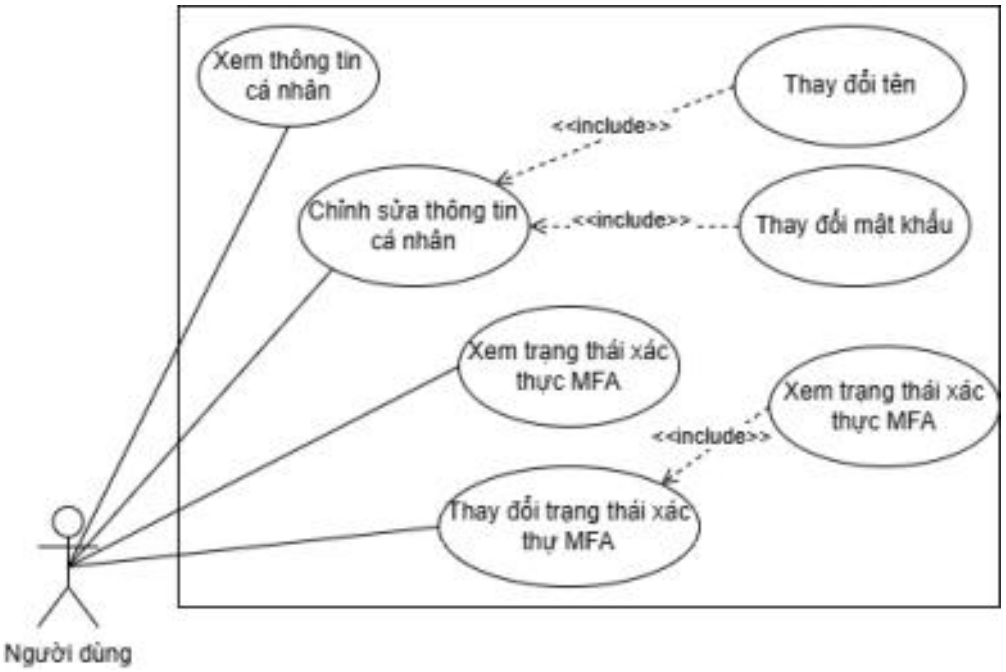
Hình 3.3.3 Biểu đồ Usecase phân rã chức năng cài đặt Space của chủ Space

Biểu đồ mô tả các chức năng nâng cao mà chỉ Thành viên Space Owner (chủ sở hữu Space) có quyền thực hiện. Các chức năng bao gồm: tạo, xem và xóa API Key của Space; chỉnh sửa các giới hạn liên quan đến lượt gọi API, số lượng file và dung lượng mỗi file; cùng với việc cập nhật thông tin cơ bản của Space. Đây là các thao tác quản trị cấp cao, đảm bảo quyền kiểm soát và cấu hình toàn diện cho người quản lý Space.



Hình 3.3.4 Biểu đồ Usecase phân rã chức năng quản lý thành viên

Quyền "Quản lý thành viên" cho phép Space Owner thực hiện các thao tác liên quan đến quản trị người dùng trong một không gian space. Cụ thể, chủ sở hữu không gian có thể xem danh sách thành viên hiện có, thêm thành viên mới vào space thông qua việc gửi lời mời, xóa thành viên không còn nhu cầu tham gia và thay đổi vai trò (Viewer, Editor hoặc Owner) để phân quyền phù hợp. Tất cả các chức năng này đều được bao gồm trong Use Case "Quản lý thành viên thuộc space", nhằm đảm bảo chủ sở hữu có thể kiểm soát, phân quyền và bảo mật không gian làm việc một cách hiệu quả.



Hình 3.3.5 Biểu đồ Usecase phân rã chức năng quản lý thông tin cá nhân

Người dùng có thể thực hiện các thao tác liên quan đến quản lý tài khoản cá nhân của mình, bao gồm: xem thông tin cá nhân, chỉnh sửa thông tin cá nhân như thay đổi tên hiển thị, mật khẩu, cũng như xem và thay đổi trạng thái xác thực đa yếu tố (MFA). Việc cho phép người dùng điều chỉnh các thông tin này giúp tăng tính chủ động, bảo mật và cá nhân hóa trải nghiệm sử dụng hệ thống.

3.4. Đặc tả biểu đồ usecase

3.4.1. Đặc tả chức năng "theo vai trò trong một space"

Bảng 3.4.1 Đặc tả usecase "Xem danh sách tài liệu của space"

Mã Use case	UC01
Tên Use case	Xem danh sách tài liệu của Space.
Tác nhân	Thành viên Space Viewer, Editor, Owner.
Mô tả	Cho phép người dùng xem tất cả tài liệu hiện có trong Space.
Tiền điều kiện	Người dùng đã đăng nhập và là thành viên của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng truy cập trang chi tiết space. 2. Hệ thống hiển thị danh sách tài liệu hiện có trong Space.
Luồng ngoại lệ	Không có tài liệu nào → hệ thống hiển thị thông báo "Hiện chưa có tài liệu nào trong space".

(Thất bại)	
Hậu điều kiện	Danh sách tài liệu được hiển thị đầy đủ.

Bảng 3.4.2 Đặc tả usecase "Thêm tài liệu của space"

Mã Use case	UC02
Tên Use case	Thêm tài liệu của Space.
Tác nhân	Thành viên Space Editor, Owner
Mô tả	Cho phép người dùng tải lên và thêm tài liệu mới vào Space.
Tiền điều kiện	Người dùng đã đăng nhập và có quyền Editor hoặc Owner.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn chức năng "Thêm tài liệu". 2. Chọn file và điền thông tin mô tả 3. Hệ thống xác nhận, hiển thị trạng thái xử lý upload, lưu tài liệu vào Space.
Luồng ngoại lệ (Thất bại)	File không hợp lệ, file vượt quá giới hạn dung lượng/định dạng, xử lý file lỗi → thông báo lỗi.
Hậu điều kiện	Danh sách tài liệu được hiển thị đầy đủ.

Bảng 3.4.3 Đặc tả usecase "Xoá tài liệu của space"

Mã Use case	UC03
Tên Use case	Xoá tài liệu của Space.
Tác nhân	Thành viên Space Editor, Owner
Mô tả	Cho phép người dùng xóa các tài liệu không còn sử dụng.
Tiền điều kiện	Người dùng đã đăng nhập và có quyền Editor hoặc Owner.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tài liệu cần xóa. 2. Chọn "Xóa" và xác nhận. 3. Hệ thống tiến hành xóa và cập nhật danh sách.
Luồng ngoại lệ	Tài liệu không tồn tại hoặc đã bị xóa trước đó → thông báo lỗi.

(Thất bại)	
Hậu điều kiện	Tài liệu không còn xuất hiện trong danh sách.

Bảng 3.4.4 Đặc tả usecase "Xem chi tiết tài liệu của Space"

Mã Use case	UC04
Tên Use case	Xem chi tiết tài liệu của Space
Tác nhân	Thành viên Space Viewer, Editor, Owner.
Mô tả	Cho phép người dùng mở và xem nội dung chi tiết của một tài liệu.
Tiền điều kiện	Người dùng đã đăng nhập và là thành viên của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn một tài liệu từ danh sách 2. Hệ thống hiển thị nội dung chi tiết của tài liệu đó.
Luồng ngoại lệ (Thất bại)	Tài liệu lỗi hoặc không thể hiển thị → hiển thị thông báo lỗi.
Hậu điều kiện	Tài liệu được mở và hiển thị nội dung chi tiết.

Bảng 3.4.5 Đặc tả usecase "Xem thành viên thuộc Space"

Mã Use case	UC05
Tên Use case	Xem thành viên thuộc Space.
Tác nhân	Thành viên Space Viewer, Editor, Owner.
Mô tả	Cho phép người dùng xem danh sách thành viên đang tham gia Space.
Tiền điều kiện	Người dùng đã đăng nhập và là thành viên của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Thành viên" 2. Hệ thống hiển thị danh sách người dùng thuộc space.
Luồng ngoại lệ	Danh sách bị lỗi khi tải → thông báo lỗi.

(Thất bại)	
Hậu điều kiện	Danh sách thành viên được hiển thị.

Bảng 3.4.6 Đặc tả usecase "Sử dụng ChatBot của Space"

Mã Use case	UC06
Tên Use case	Sử dụng ChatBot của Space.
Tác nhân	Thành viên Space Viewer, Editor, Owner.
Mô tả	Cho phép người dùng trò chuyện với ChatBot để tương tác với nội dung tài liệu.
Tiền điều kiện	Người dùng đã đăng nhập và là thành viên của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng truy cập trang chatbox. 2. Người dùng nhập nội dung câu hỏi. 3. Hệ thống ChatBot xử lý và trả lời dựa trên tài liệu đã có.
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Người dùng nhận được câu trả lời từ ChatBot.

Bảng 3.4.7 Đặc tả usecase "Quản lý thành viên thuộc Space"

Mã Use case	UC07
Tên Use case	Quản lý thành viên thuộc Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cho phép chủ Space thêm, xóa hoặc chỉnh sửa quyền của thành viên thuộc Space.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Thành viên" 2. Thêm/xóa/chỉnh sửa quyền các thành viên. 3. Hệ thống cập nhật danh sách thành viên
Luồng ngoại lệ	Người dùng đã bị xóa trước đó → thông báo lỗi

(Thất bại)	
Hậu điều kiện	Danh sách thành viên với quyền được hiển thị đúng.

Bảng 3.4.8 Đặc tả usecase "Cài đặt Space"

Mã Use case	UC08
Tên Use case	Cài đặt Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cho phép chủ Space chỉnh sửa giới hạn API, dung lượng file, số lượng file và thông tin Space.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Cài đặt" 2. Chỉnh sửa các giới hạn và thông tin cơ bản 3. Lưu thay đổi
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Giới hạn và thông tin Space được cập nhật.

Bảng 3.4.9 Đặc tả usecase "Xóa Space"

Mã Use case	UC09
Tên Use case	Xoá Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cho phép chủ Space xóa toàn bộ Space, bao gồm dữ liệu và thành viên.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn "Xoá space". 2. Hệ thống yêu cầu xác nhận. 3. Tiến hành xóa toàn bộ dữ liệu Space
Luồng ngoại lệ	Hủy thao tác trước khi xác nhận → không thực hiện xóa

(Thất bại)	
Hậu điều kiện	Space bị xóa hoàn toàn khỏi hệ thống.

3.4.2. Đặc tả chức năng cài đặt Space của chủ space (Space Owner)

Bảng 3.4.10 Đặc tả usecase "Xem API Keys của Space"

Mã Use case	UC10
Tên Use case	Xem API Keys của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cho phép chủ Space xem danh sách các API Key đang tồn tại.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Chat Api". 2. Hệ thống hiển thị danh sách API Key hiện có.
Luồng ngoại lệ (Thất bại)	Dữ liệu không thể tải → hiển thị lỗi.
Hậu điều kiện	Danh sách API Keys được hiển thị.

Bảng 3.4.11 Đặc tả usecase "Tạo API Key của Space"

Mã Use case	UC11
Tên Use case	Tạo API Key của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cho phép chủ Space tạo một API Key cho Space.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	3. Người dùng chọn "Tạo API Key". 4. Nhập tên/mô tả cho Key. 5. Hệ thống sinh API Key và hiển thị.

Luồng ngoại lệ (Thất bại)	Hệ thống tạo thất bại → hiển thị lỗi
Hậu điều kiện	API Key mới được tạo và lưu.

Bảng 3.4.12 Đặc tả usecase "Xoá API Key của Space"

Mã Use case	UC12
Tên Use case	Xoá API Key của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Xóa bỏ API Key không còn sử dụng.
Tiền điều kiện	Có ít nhất 1 API Key tồn tại.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn API Key muốn xóa. 2. Xác nhận xóa. 3. Hệ thống xóa và cập nhật danh sách.
Luồng ngoại lệ (Thất bại)	API Key đã bị xóa trước đó → hiển thị lỗi.
Hậu điều kiện	API Key không còn xuất hiện.

Bảng 3.4.13 Đặc tả usecase "Chỉnh sửa giới hạn số lượt gọi API của Space"

Mã Use case	UC13
Tên Use case	Chỉnh sửa giới hạn số lượt gọi API của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Thiết lập hoặc điều chỉnh số lượt gọi API mỗi ngày.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Giới hạn" 2. Nhập giới hạn mới 3. Hệ thống lưu và áp dụng giới hạn

Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Giới hạn gọi API được cập nhật.

Bảng 3.4.14 Đặc tả usecase "Chỉnh sửa giới hạn số lượng file của Space"

Mã Use case	UC14
Tên Use case	Chỉnh sửa giới hạn số lượng file của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Giới hạn tổng số lượng file có thể tải lên Space
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab"Giới hạn" 2. Chọn số lượng file tối đa 3. Lưu thay đổi
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Hệ thống cập nhật và giới hạn số lượng file.

Bảng 3.4.15 Đặc tả usecase "Chỉnh sửa giới hạn độ lớn mỗi file của tài liệu"

Mã Use case	UC15
Tên Use case	Chỉnh sửa giới hạn độ lớn mỗi file của tài liệu.
Tác nhân	Thành viên Space Owner.
Mô tả	Giới hạn độ lớn file có thể tải lên Space
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Giới hạn" 2. Chọn kích thước file tối đa 3. Lưu thay đổi

Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Kích thước tối đa mỗi file được cập nhật.

Bảng 3.4.16 Đặc tả usecase "Chỉnh sửa thông tin cơ bản của Space"

Mã Use case	UC16
Tên Use case	Chỉnh sửa thông tin cơ bản của Space.
Tác nhân	Thành viên Space Owner.
Mô tả	Cập nhật thông tin chi tiết của space.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "General" 2. Nhập nội dung cần cập nhật 3. Lưu thay đổi
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Thông tin cơ bản của Space được cập nhật thành công.

3.4.3. Đặc tả chức năng quản lý thành viên

Bảng 3.4.17 Đặc tả usecase "Thêm thành viên"

Mã Use case	UC17
Tên Use case	Thêm thành viên.
Tác nhân	Thành viên Space Owner.
Mô tả	Chủ space cần thêm mới thành viên vào space với role tương ứng thông qua link hoặc lời mời.
Tiền điều kiện	Người dùng đã đăng nhập và có quyền Owner.

Luồng sự kiện chính (Thành công)	1. Người dùng chọn "Mời thành viên". 2. Tìm kiếm theo tên hoặc email. 3. Chọn vai trò (role). 4. Copy link hoặc bấm gửi lời mời.
Luồng ngoại lệ (Thất bại)	Thành viên đã tồn tại → Thông báo đã là thành viên thuộc space.
Hậu điều kiện	Thành viên mới được thêm vào danh sách.

Bảng 3.4.18 Đặc tả usecase " Thay đổi vai trò"

Mã Use case	UC18
Tên Use case	Thay đổi vai trò thành viên.
Tác nhân	Thành viên Space Owner.
Mô tả	Chủ space thay đổi vai trò của một thành viên hiện có (ví dụ từ Viewer sang Editor).
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn thành viên cần đổi vai trò. 2. Chọn vai trò mới trong danh sách. 3. Hệ thống cập nhật và hiển thị vai trò mới.
Luồng ngoại lệ (Thất bại)	Không được thay đổi vai trò chính mình.
Hậu điều kiện	Vai trò của thành viên được cập nhật thành công.

Bảng 3.4.19 Đặc tả usecase "Xoá thành viên"

Mã Use case	UC19
Tên Use case	Xoá thành viên.
Tác nhân	Thành viên Space Owner.
Mô tả	Chủ Space muốn xoá một thành viên khỏi Space.
Tiền điều kiện	Người dùng đã đăng nhập và là chủ của Space.

Luồng sự kiện chính (Thành công)	1. Người dùng chọn một thành viên cụ thể. 2. Chọn chức năng "Xoá thành viên". 3. Hệ thống xác nhận hành động và xoá khỏi danh sách.
Luồng ngoại lệ (Thất bại)	Không thể tự xoá chính mình.
Hậu điều kiện	Thành viên bị xoá khỏi Space và mất quyền truy cập.

3.4.4. Đặc tả chức năng quản lý thông tin cá nhân

Bảng 3.4.20 Đặc tả usecase "Xem thông tin cá nhân"

Mã Use case	UC20
Tên Use case	Xem thông tin cá nhân.
Tác nhân	Người dùng.
Mô tả	Cho phép người dùng xem thông tin cá nhân của mình như tên, email, ngày tạo tài khoản, trạng thái bảo mật MFA,…
Tiền điều kiện	Người dùng đã đăng nhập.
Luồng sự kiện chính (Thành công)	1. Người dùng truy cập "Thông tin cá nhân". 2. Hệ thống hiển thị thông tin người dùng.
Luồng ngoại lệ (Thất bại)	Không thể truy xuất dữ liệu → thông báo lỗi
Hậu điều kiện	Thông tin cá nhân được hiển thị.

Bảng 3.4.21 Đặc tả usecase "Chỉnh sửa thông tin cá nhân"

Mã Use case	UC21
Tên Use case	Chỉnh sửa thông tin cá nhân.
Tác nhân	Người dùng.
Mô tả	Cho phép người dùng cập nhật thông tin cá nhân của mình.
Tiền điều kiện	Người dùng đã đăng nhập.

Luồng sự kiện chính (Thành công)	1. Người dùng truy cập "Thông tin cá nhân". 2. Chọn "Chỉnh sửa". 3. Người dùng cập nhật và nhấn lưu. 4. Hệ thống xác nhận và cập nhật dữ liệu
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Thông tin cá nhân được cập nhật.

Bảng 3.4.22 Đặc tả usecase "Xem trạng thái xác thực MFA"

Mã Use case	UC22
Tên Use case	Thay đổi trạng thái xác thực MFA.
Tác nhân	Người dùng.
Mô tả	Hiển thị trạng thái hiện tại của xác thực MFA (đang bật/tắt)
Tiền điều kiện	Người dùng đã đăng nhập.
Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Bảo mật". 2. Hệ thống hiển thị trạng thái MFA.
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Trạng thái MFA được hiển thị.

Bảng 3.4.23 Đặc tả usecase "Thay đổi trạng thái xác thực MFA"

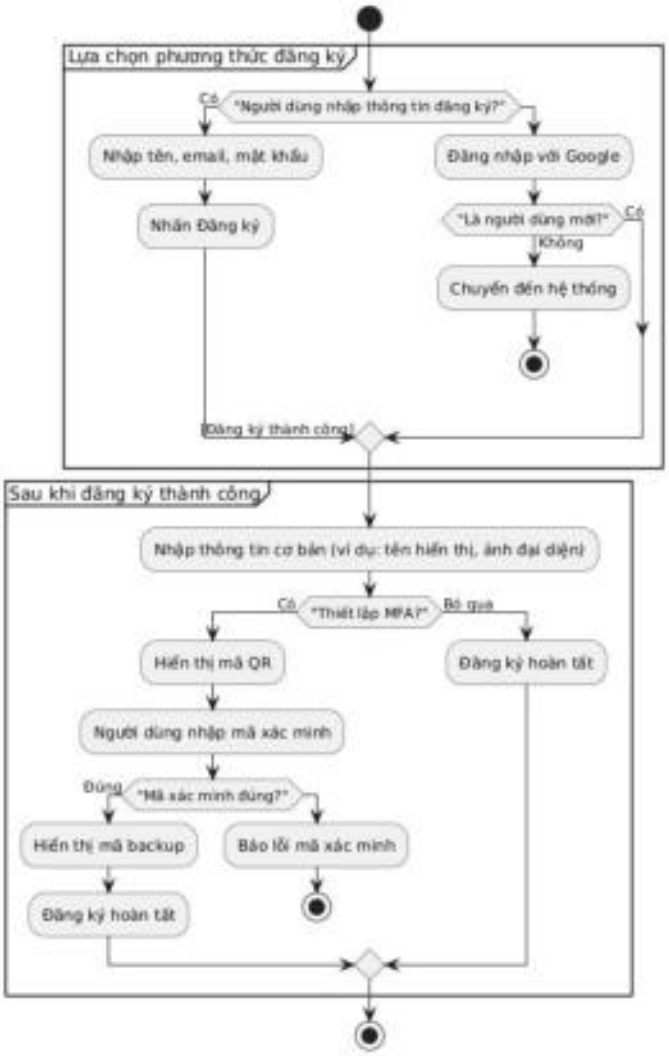
Mã Use case	UC23
Tên Use case	Thay đổi trạng thái xác thực MFA.
Tác nhân	Người dùng.
Mô tả	Cho phép người dùng bật hoặc tắt MFA cho tài khoản
Tiền điều kiện	Người dùng đã đăng nhập.

Luồng sự kiện chính (Thành công)	1. Người dùng chọn tab "Bảo mật". 2. Chọn "Thiết lập xác thực 2 yếu tố". 3. Quét mã QR sau đó nhập code. 4. Chọn xác minh và kích hoạt 5. Hệ thống cập nhật trạng thái.
Luồng ngoại lệ (Thất bại)	
Hậu điều kiện	Trạng thái MFA được cập nhật thành công.

3.5. Biểu đồ hoạt động

3.5.1. Ứng dụng Web (Web App)

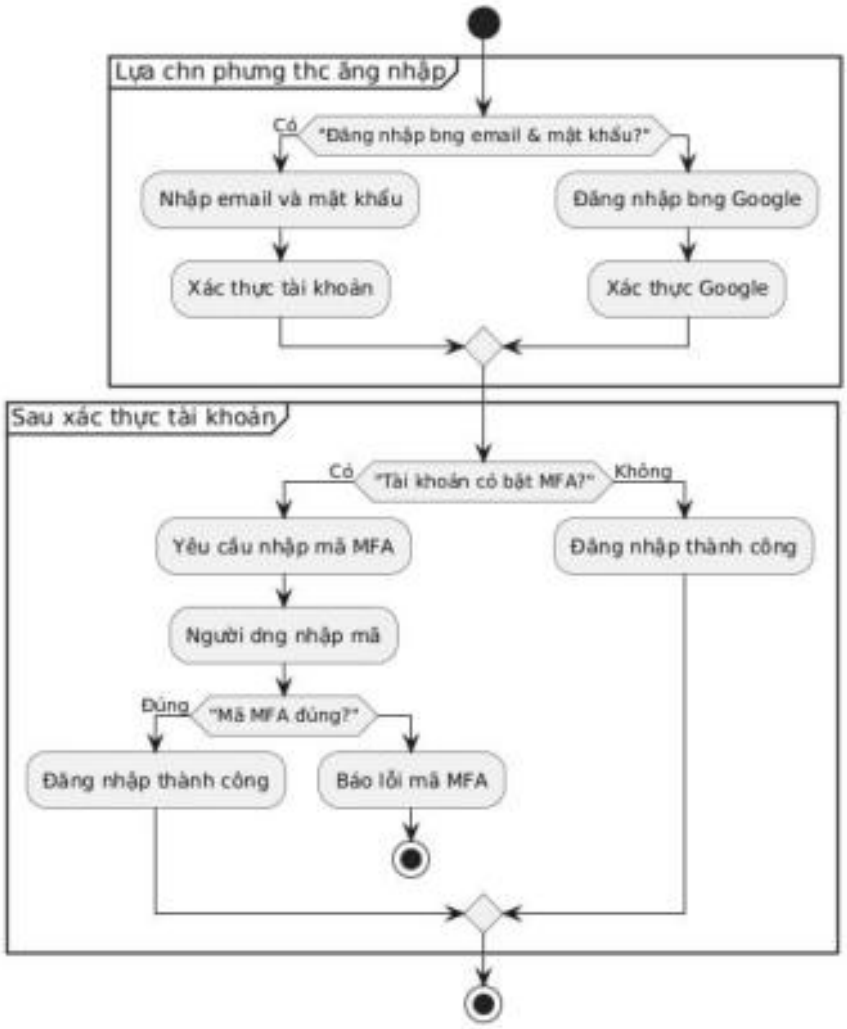
- ***Đăng ký***



Hình 3.5.1 Biểu đồ hoạt động đăng ký

Người dùng có thể chọn đăng ký bằng cách nhập thông tin thủ công hoặc sử dụng đăng nhập Google. Nếu là người dùng mới, họ sẽ tiếp tục khai báo thông tin cơ bản và tùy chọn thiết lập xác thực hai yếu tố (MFA). Trong trường hợp bật MFA, người dùng cần quét mã QR và nhập mã xác minh. Nếu mã hợp lệ, hệ thống hiển thị mã dự phòng và hoàn tất đăng ký. Ngược lại, nếu bỏ qua MFA hoặc mã xác minh sai, quy trình xử lý tương ứng sẽ diễn ra.

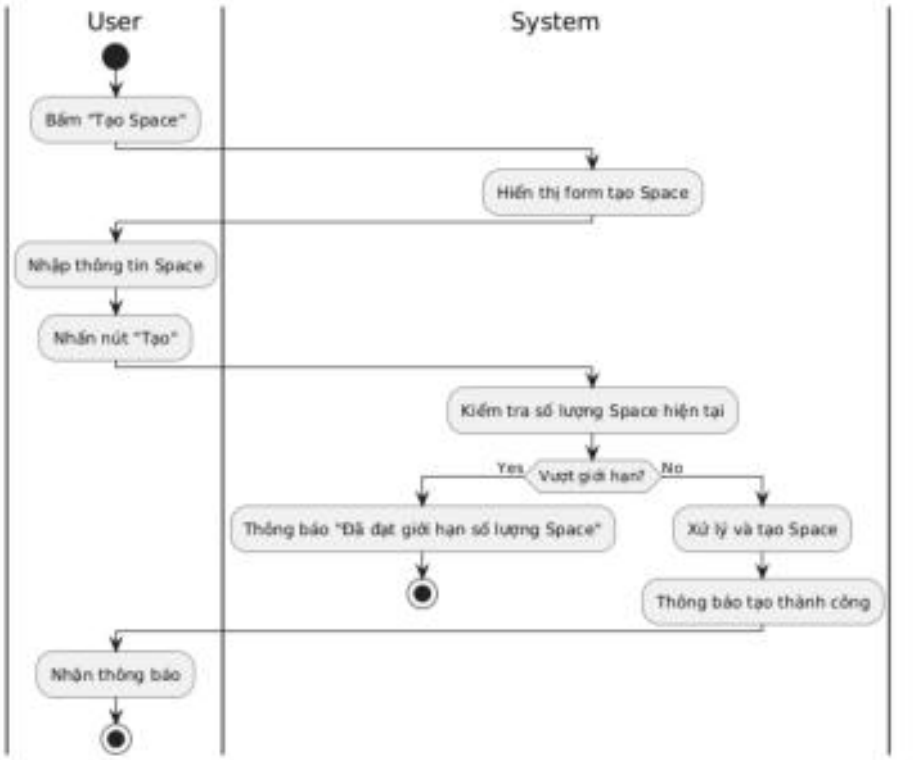
- ***Đăng nhập***



Hình 3.5.2 Biểu đồ hoạt động đăng nhập

Người dùng có thể chọn đăng nhập bằng email và mật khẩu hoặc thông qua Google. Sau khi xác thực tài khoản thành công, hệ thống kiểm tra xem tài khoản có bật xác thực đa yếu tố (MFA) hay không. Nếu có, người dùng được yêu cầu nhập mã MFA. Nếu mã hợp lệ, quá trình đăng nhập hoàn tất; nếu không, hệ thống báo lỗi. Nếu tài khoản không bật MFA, người dùng sẽ được đăng nhập ngay sau bước xác thực ban đầu.

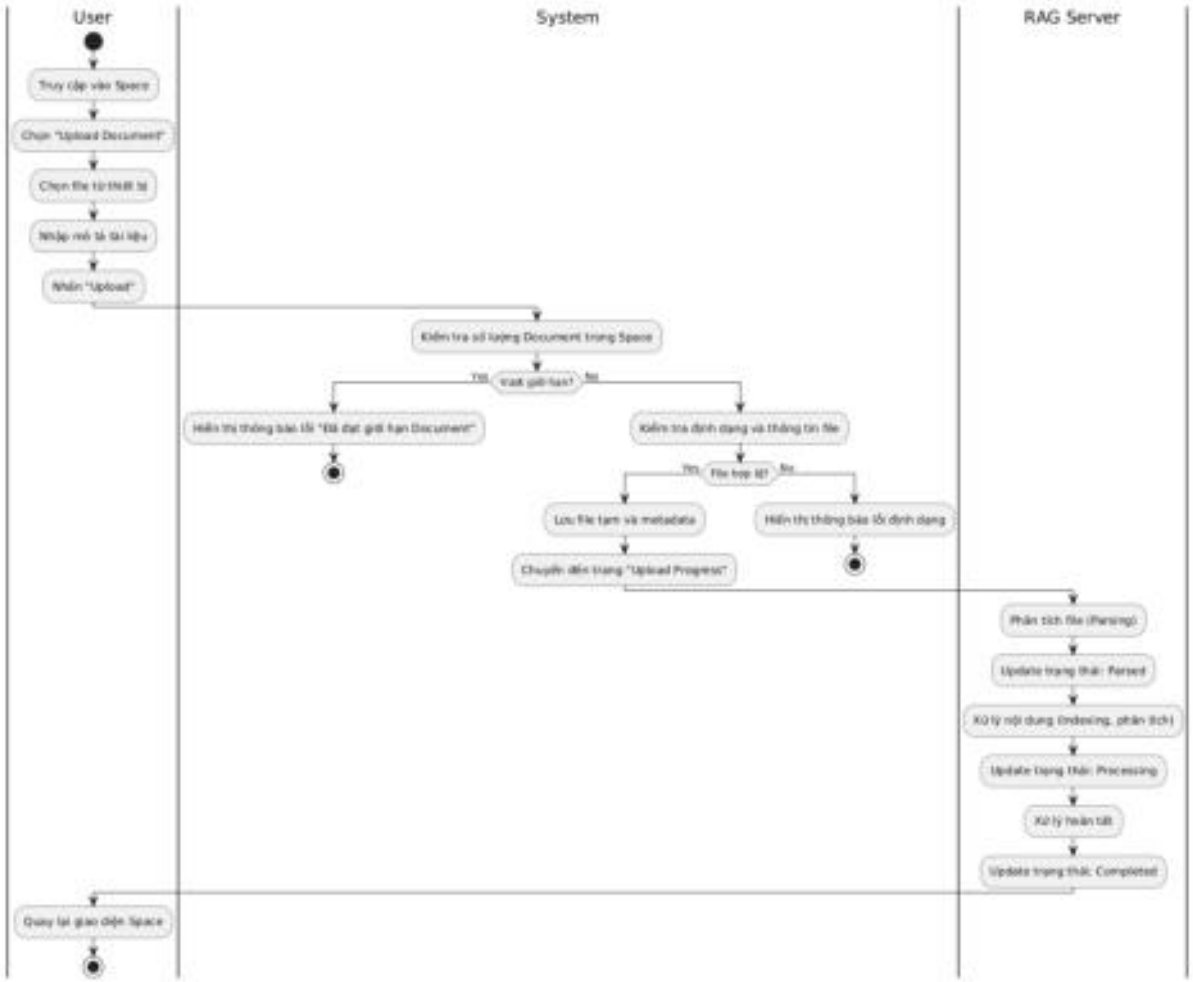
- ***Tạo Space***



Hình 3.5.3 Biểu đồ hoạt động tạo Space

Người dùng bắt đầu bằng cách bấm nút "Tạo Space", hệ thống sẽ hiển thị form nhập thông tin. Sau khi người dùng nhập dữ liệu và nhấn "Tạo", hệ thống sẽ kiểm tra xem người dùng đã đạt đến giới hạn số lượng Space cho phép hay chưa. Nếu đã đạt giới hạn, hệ thống thông báo và kết thúc quy trình. Nếu chưa, hệ thống xử lý yêu cầu, tạo Space mới và gửi thông báo thành công trở lại cho người dùng.

- ***Tải tài liệu lên hệ thống***

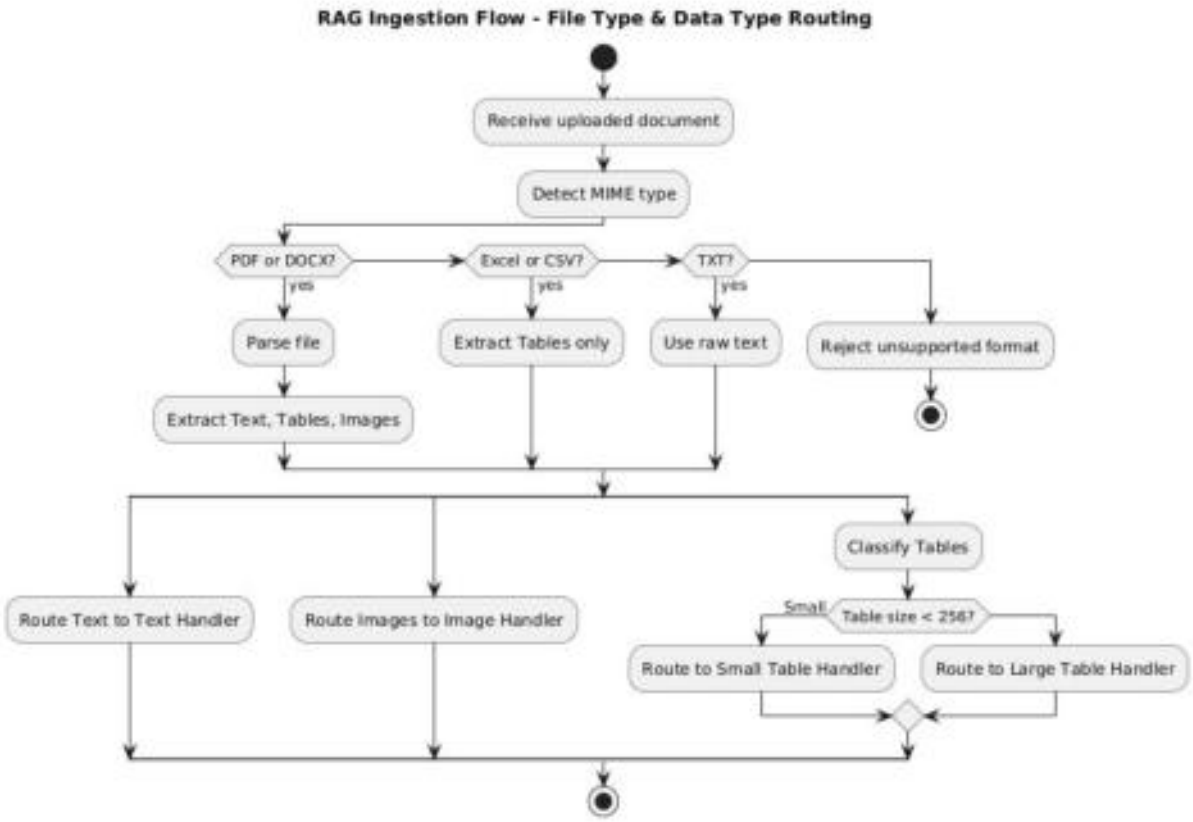


Hình 3.5.4 Biểu đồ hoạt động tải tài liệu lên hệ thống

Người dùng tải lên tài liệu vào Space, trong đó việc phân tích nội dung và xử lý tài liệu được chuyển sang RAG Server. Hệ thống kiểm tra số lượng tài liệu trong Space, nếu đã đạt giới hạn sẽ thông báo lỗi. Nếu không, hệ thống kiểm tra định dạng file, lưu tạm và chuyển trạng thái sang trang "Upload Progress". Lúc này, RAG Server đảm nhiệm việc phân tích tài liệu, xử lý nội dung và cập nhật các trạng thái tương ứng cho tới khi hoàn tất quá trình upload.

3.5.2. Hệ thống RAG

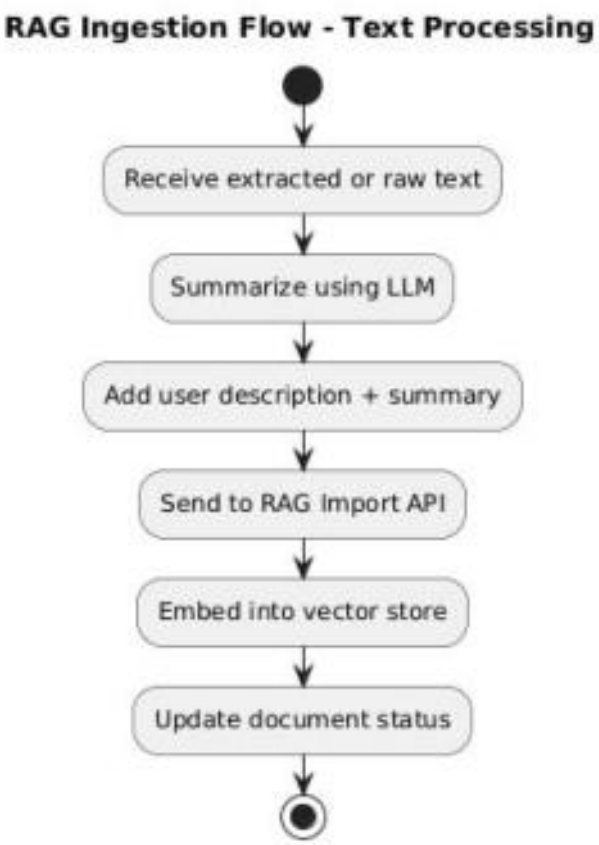
- ***Xử lý phân loại, chiết xuất các loại thông tin từ văn bản***



Hình 3.5.5 Biểu đồ hoạt động xử lý phân loại, chiết xuất các loại thông tin từ văn bản

Hệ thống sẽ dựa vào từng loại tệp dữ liệu được Upload lên và trích xuất các loại thông tin có trong tệp đó. Tuỳ từng loại dữ liệu hệ thống sẽ có cách phân tích và lưu trữ khác nhau để tối ưu hoá khả năng truy vấn.

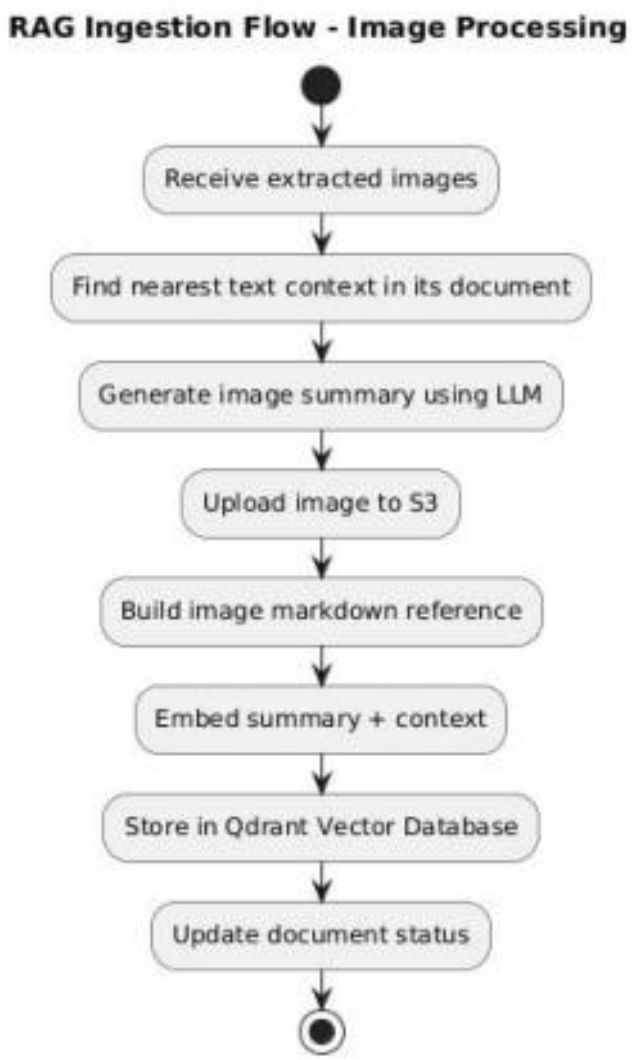
- ***Xử lý thông tin dạng văn bản***



Hình 3.5.6 Biểu đồ hoạt động xử lý thông tin dạng văn bản

Văn bản được trích xuất từ tài liệu (PDF, DOCX hoặc file văn bản). Hệ thống tóm tắt nội dung bằng mô hình ngôn ngữ (LLM), kết hợp mô tả từ người dùng, sau đó gửi nội dung đã tóm tắt đến backend RAG để lưu trữ và tìm kiếm ngữ nghĩa.

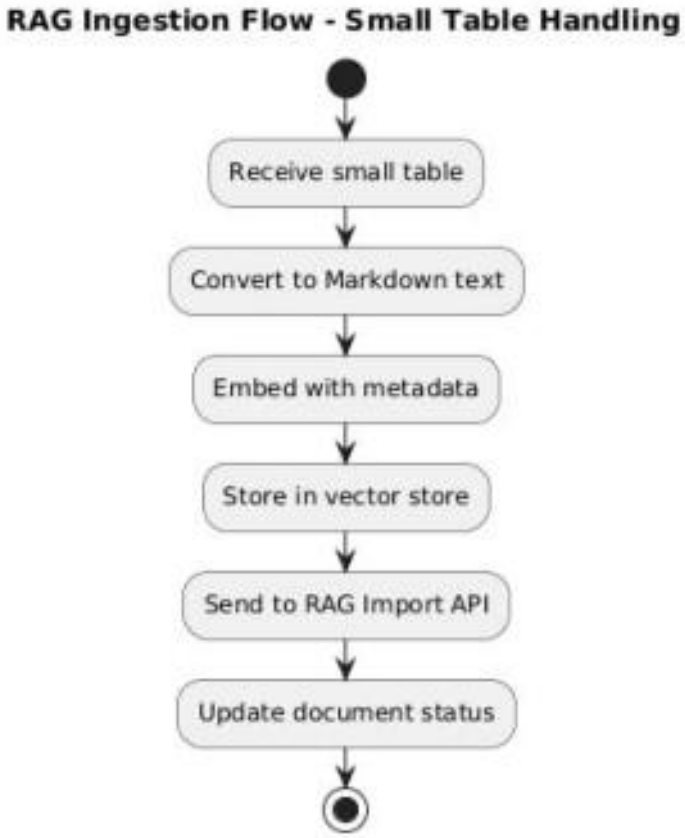
- ***Xử lý thông tin dạng hình ảnh***



Hình 3.5.7 Biểu đồ hoạt động xử lý thông tin dạng hình ảnh

Các hình ảnh trong tài liệu được xử lý bằng cách tìm ngữ cảnh xung quanh, phân tích nội dung hình bằng mô hình AI (GPT-4o), và sinh tóm tắt. Hình được tải lên S3, sau đó embedding của phần mô tả và ngữ cảnh được lưu trữ vào Qdrant để phục vụ truy vấn ngữ nghĩa theo hình ảnh.

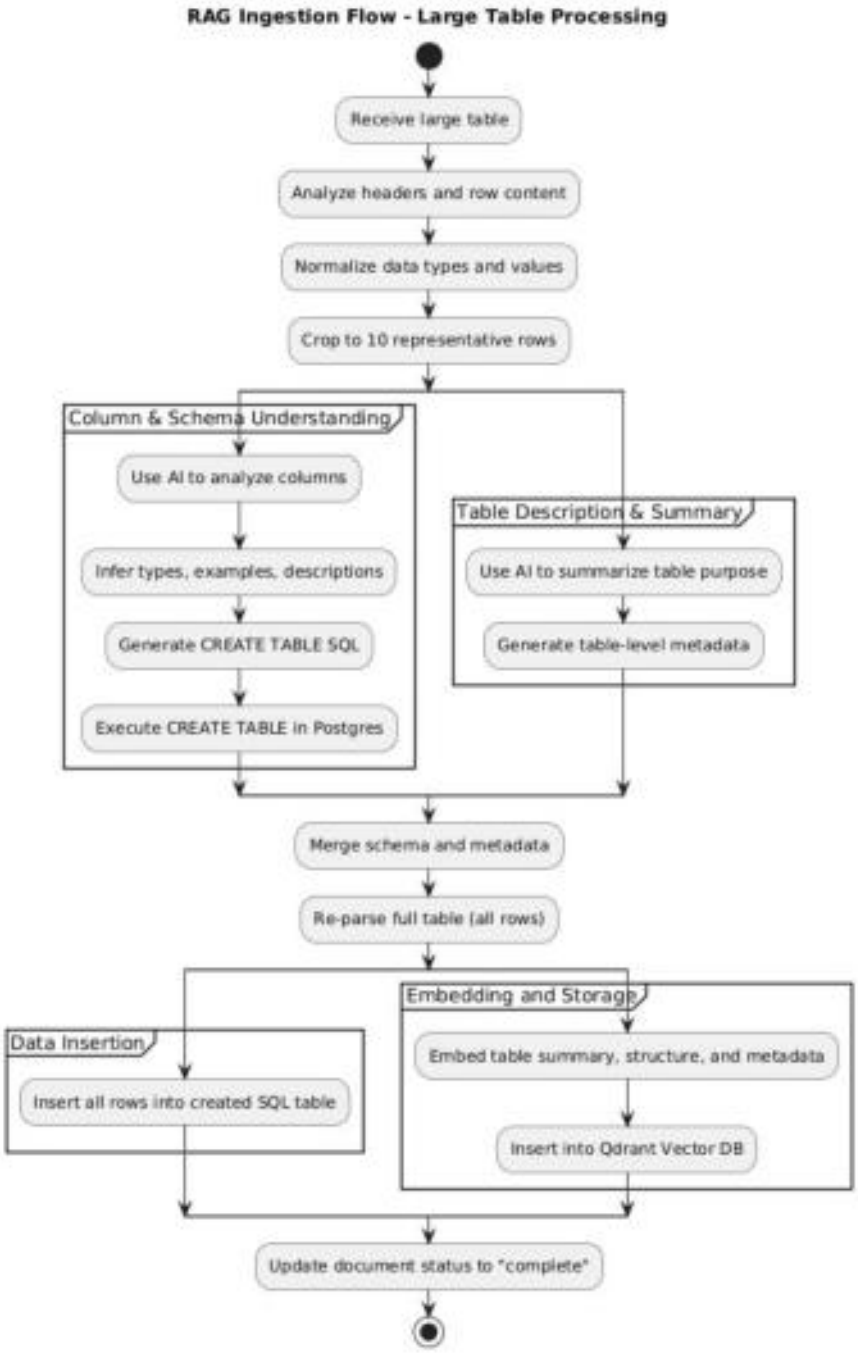
- ***Xử lý thông tin dạng bảng (nhỏ)***



Hình 3.5.8 Biểu đồ hoạt động xử lý thông tin dạng bảng (nhỏ)

Các bảng nhỏ (ít dữ liệu, cấu trúc đơn giản) được chuyển thành dạng văn bản Markdown, sau đó xử lý tương tự như văn bản: thực hiện embedding và gửi lên hệ thống RAG để phục vụ tìm kiếm. Việc này giúp tận dụng thông tin từ bảng một cách linh hoạt mà không cần lưu trữ có cấu trúc.

- ***Xử lý thông tin dạng bảng (lớn)***

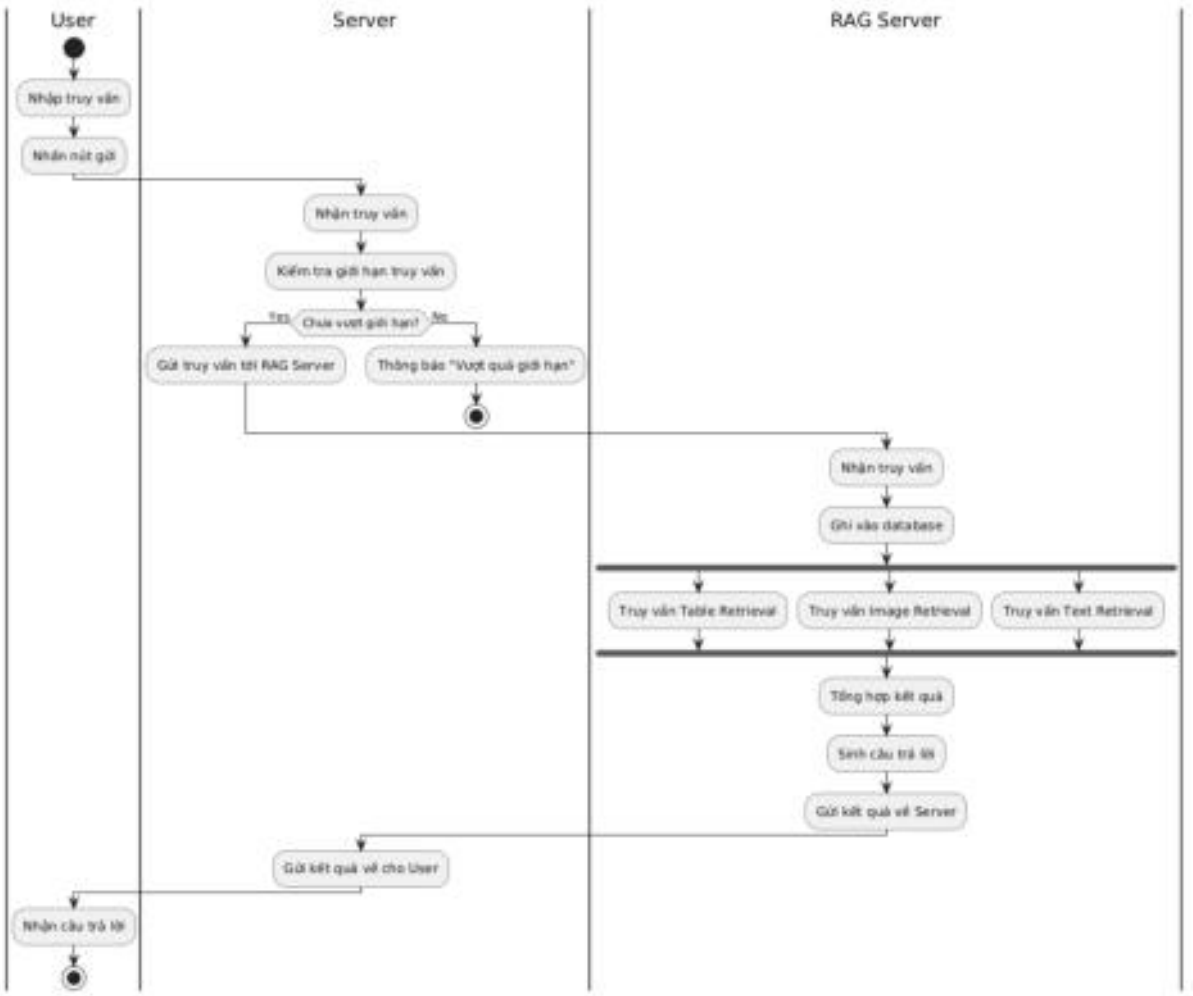


Hình 3.5.9 Biểu đồ hoạt động xử lý thông tin dạng bảng (lớn)

Trong quy trình xử lý bảng dữ liệu lớn, hệ thống chủ động cắt giảm bảng xuống còn 10 dòng đại diện trước khi đưa vào AI phân tích. Việc này nhằm tối ưu hóa hiệu suất và đảm bảo đầu vào nằm trong giới hạn xử lý của các mô hình ngôn ngữ lớn (LLM), vốn không phù hợp với hàng trăm hay hàng nghìn dòng dữ liệu cùng lúc. Mặc dù chỉ sử dụng 10 dòng, nhưng đây là số lượng đủ để AI hiểu được cấu trúc chung của bảng, kiểu dữ liệu từng cột, định dạng phổ biến, cũng như ngữ cảnh tổng thể. Nhờ đó, hệ thống có thể sinh ra câu lệnh CREATE TABLE chính xác và xây dựng metadata mô tả nội dung

bảng một cách hiệu quả, mà không cần tải toàn bộ dữ liệu vào mô hình. Đây là sự cân bằng giữa tính chính xác và hiệu suất, rất phù hợp với các workflow xử lý quy mô lớn.

- ***Xử lý truy vấn của người dùng***

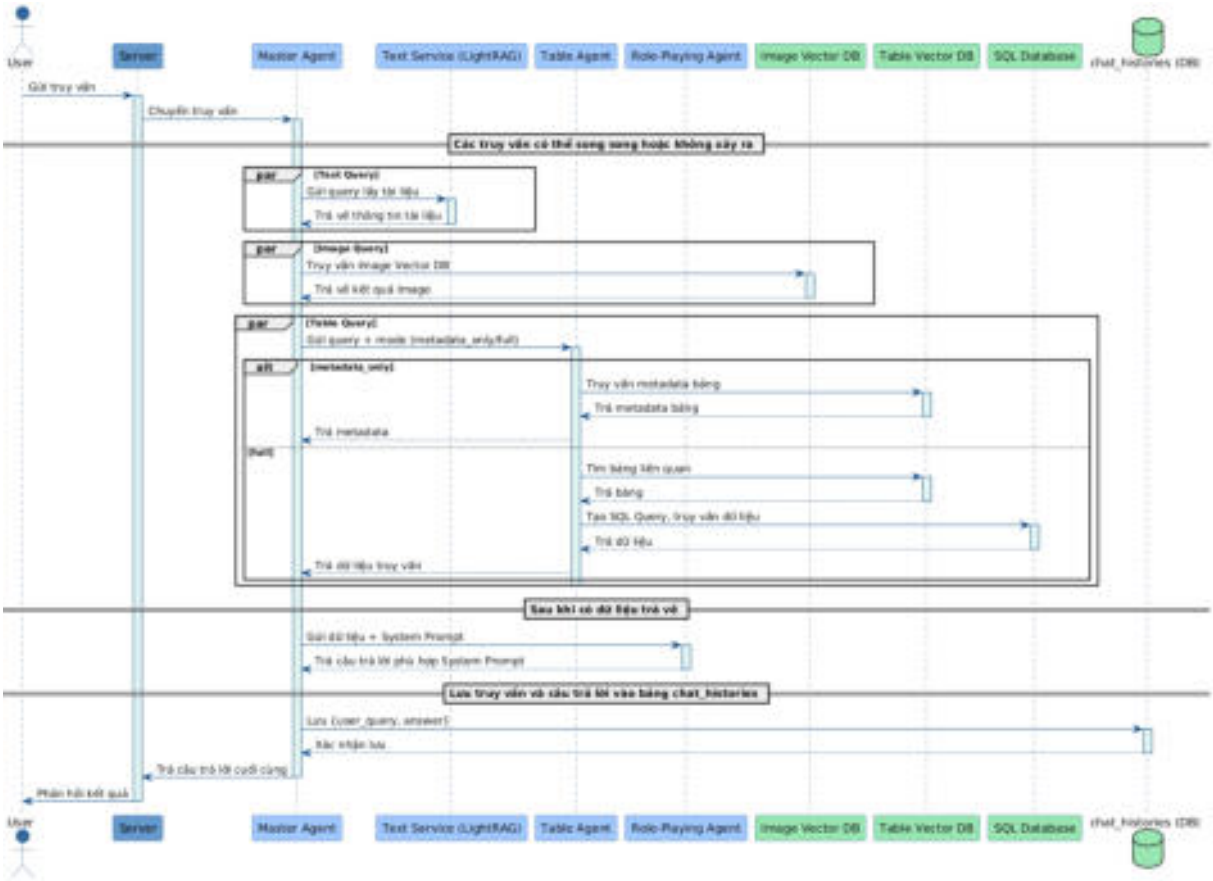


Hình 3.5.10 Biểu đồ hoạt động xử lý truy vấn của người dùng

Người dùng khởi tạo truy vấn và gửi đến Server. Tại đây, Server kiểm tra giới hạn truy vấn để đảm bảo không vượt quá số lượng cho phép. Nếu hợp lệ, truy vấn được chuyển tiếp đến RAG Server. RAG Server ghi nhận truy vấn vào cơ sở dữ liệu, sau đó đồng thời thực hiện truy vấn ba nguồn dữ liệu: bảng (Table Retrieval), hình ảnh (Image Retrieval), và văn bản (Text Retrieval). Kết quả từ các nguồn được tổng hợp, sinh câu trả lời và gửi về Server, sau đó được trả lại cho người dùng.

3.6. Biểu đồ tuần tự

3.6.1. Quy trình truy vấn dữ liệu và trả lời theo truy vấn của người dùng



Hình 3.6.1 Biểu đồ tuần tự quy trình truy vấn và trả lời truy vấn

User gửi truy vấn đến Server, Server chuyển tiếp truy vấn cho Master Agent. Master Agent đồng thời gửi truy vấn đến các dịch vụ và cơ sở dữ liệu tương ứng để lấy thông tin:

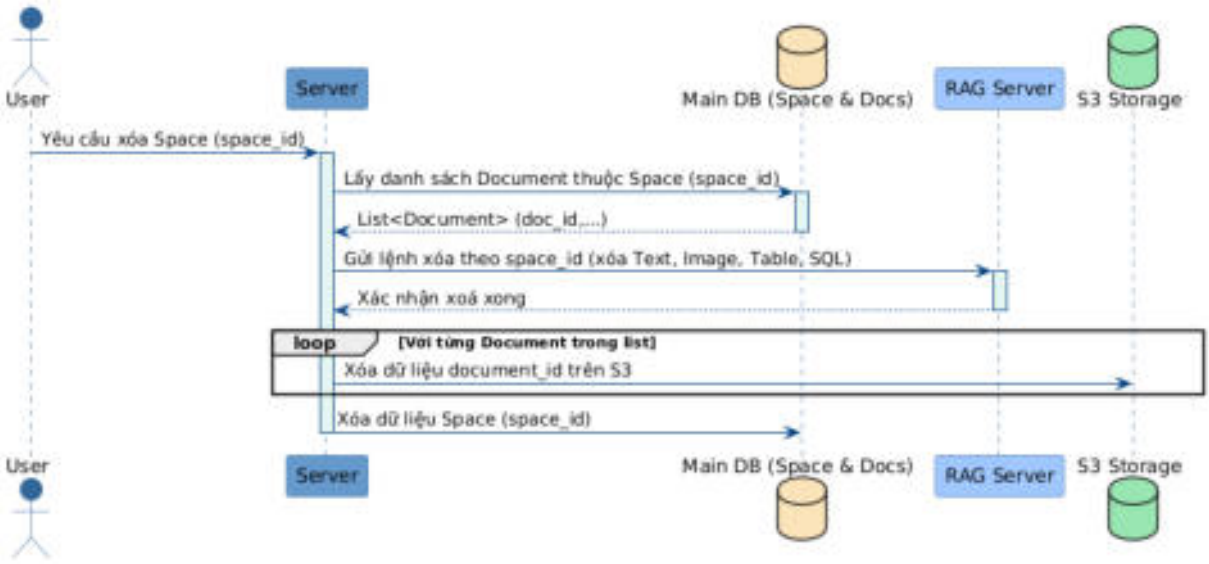
- Text Service: tìm tài liệu liên quan theo truy vấn.
- Image Vector DB: truy vấn dữ liệu ảnh vector.
- Table Agent: dựa trên chế độ metadata_only hoặc full để lấy metadata bảng hoặc truy vấn dữ liệu thực từ Table Vector DB và SQL Database.

Các truy vấn này có thể thực hiện song song hoặc không cần phải xảy ra đồng thời.

Sau khi nhận được các dữ liệu truy vấn, Master Agent chuyển kết quả cùng System Prompt đến Role-Playing Agent để tạo ra câu trả lời phù hợp với yêu cầu và ngữ cảnh đã định. Role-Playing Agent trả lại câu trả lời cho Master Agent, sau đó Master Agent lưu lại truy vấn và câu trả lời vào bảng chat_histories và gửi phản hồi cuối cùng về Server, rồi Server trả lại kết quả cho User.

Hệ thống lưu toàn bộ truy vấn và câu trả lời trong bảng chat_histories để phục vụ việc theo dõi lịch sử và cải thiện chất lượng trả lời. Khi cần xóa dữ liệu (như Space hoặc Document), Server sẽ lấy danh sách document để xóa dữ liệu liên quan trên S3, đồng thời gửi lệnh xóa tổng thể cho RAG Server dựa trên space_id để xóa dữ liệu vector trong các dịch vụ Text, Image, Table Vector DB và SQL Database, đảm bảo dữ liệu được dọn sạch toàn diện.

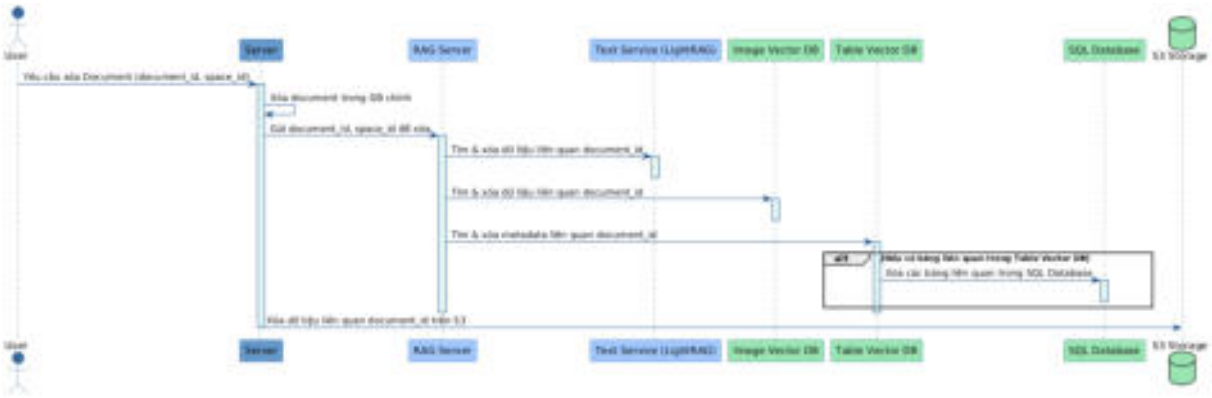
3.6.2. Quy trình xoá một Space



Hình 3.6.2 Biểu đồ tuần tự quy trình xoá space

Người dùng gửi yêu cầu xoá Space đến Server. Server sẽ truy vấn danh sách các tài liệu (Document) thuộc Space đó từ cơ sở dữ liệu chính, sau đó gửi lệnh xoá dựa trên Space ID đến RAG Server để xoá toàn bộ dữ liệu liên quan đến Space trong các hệ thống lưu trữ vector như Text Service, Image Vector Database và Table Vector Database, cũng như các bảng dữ liệu tương ứng trong SQL Database. Tiếp theo, Server tiến hành xoá từng tài liệu trên bộ nhớ S3 dựa theo danh sách Document đã lấy trước đó. Cuối cùng, Server xoá bản ghi Space trong cơ sở dữ liệu chính để hoàn tất quá trình.

3.6.3. Quy trình xoá một tệp tài liệu



Hình 3.6.3 Biểu đồ quy trình xoá tệp tài liệu

Người dùng gửi yêu cầu xoá Document đến Server. Server chuyển tiếp yêu cầu này đến RAG Server để xoá các dữ liệu vector liên quan đến Document trong các dịch vụ lưu trữ vector và dữ liệu trong SQL Database nếu có. Song song đó, Server cũng xoá dữ liệu gốc của Document trên bộ nhớ S3. Sau khi các bước xoá dữ liệu hoàn tất, Server sẽ xoá bản ghi Document trong cơ sở dữ liệu chính, đảm bảo toàn bộ dữ liệu liên quan được loại bỏ sạch sẽ.

3.7. Cơ sở dữ liệu

Hình 3.7.1 Cơ sở dữ liệu

Cơ sở dữ liệu bao gồm các bảng:

- Bảng **users**

  Lưu thông tin người dùng hệ thống.

Bảng 3.7.1 Bảng users

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
username	character varying(255)	Tên đăng nhập (duy nhất)
email	character varying(255)	Email người dùng (duy nhất)

tier_id	integer	Gói dịch vụ đang dùng
active	boolean	Trạng thái hoạt động
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật
mfa_enabled	boolean	Trạng thái xác thực đa yếu tố (MFA)

Index:

- email: tìm kiếm nhanh bằng GIN (trigram)
- username: tìm kiếm nhanh bằng GIN (trigram)

- Bảng **user_auth_credentials**

Lưu thông tin đăng nhập của người dùng.

Bảng 3.7.2 Bảng user_auth_credentials

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
user_id	integer	Liên kết người dùng
auth_type	character varying(50)	Loại xác thực (password, Google,...)
password_hash	text	Hash mật khẩu
external_id	character varying(255)	ID của dịch vụ bên ngoài (Google, Facebook,...)
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật

- Bảng **user_mfas**

Thông tin xác thực đa yếu tố (MFA).

Bảng 3.7.3 Bảng user_mfas

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
user_id	integer	Liên kết người dùng

secret	character varying(255)	Mã bí mật MFA
backup_codes	jsonb	Danh sách mã dự phòng
verified	boolean	Đã xác thực hay chưa
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật

- Bảng **tiers**

  Quản lý các gói dịch vụ của người dùng.

Bảng 3.7.4 Bảng tiers

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
space_limit	integer	Giới hạn số không gian
document_limit	integer	Giới hạn số tài liệu
query_history_limit	integer	Giới hạn lịch sử truy vấn
query_limit	integer	Giới hạn truy vấn
cost_month	numeric(10,2)	Chi phí theo tháng
discount	numeric(5,2)	Mức chiết khấu (%)
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật
file_size_limit_kb	integer	Giới hạn kích thước tệp (KB)
api_call_limit	integer	Giới hạn gọi API

- Bảng **spaces**

  Lưu thông tin các không gian lưu trữ tài liệu

Bảng 3.7.5 Bảng spaces

Tên cột	Kiểu dữ liệu	Mô tả

id	integer	Khóa chính
name	character varying(255)	Tên không gian
description	text	Mô tả không gian
privacy_status	boolean	Trạng thái riêng tư
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật
document_limit	integer	Giới hạn số tài liệu
file_size_limit_kb	integer	Giới hạn kích thước tệp (KB)
api_call_limit	integer	Giới hạn lượt gọi API
system_prompt	character varying(1024)	System Prompt ChatBot của không gian

- Bảng **space_roles**

Định nghĩa vai trò của người dùng trong không gian.

Bảng 3.7.6 Bảng space_roles

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
name	character varying(255)	Tên vai trò (duy nhất)
permission	integer	Giá trị quyền hạn tương ứng
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật

- Bảng **space_users**

Quản lý các thành viên trong mỗi không gian.

Bảng 3.7.7 Bảng space_users

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính

user_id	integer	Người dùng tham gia
space_id	integer	Không gian tương ứng
space_role_id	integer	Vai trò trong không gian
created_at	timestamp	Thời điểm thêm vào
updated_at	timestamp	Thời điểm cập nhật

Index: Khoá (user_id, space_id) là duy nhất.

- Bảng **space_invitations**

Quản lý lời mời tham gia không gian đến người dùng cụ thể.

Bảng 3.7.8 Bảng space_invitations

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
space_id	integer	Không gian mời
space_role_id	integer	Vai trò được gán khi tham gia
invited_user_id	integer	Người được mời
inviter_id	integer	Người gửi lời mời
status	character varying(50)	Trạng thái lời mời
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật

Index: Khoá (invited_user_id, space_id) là duy nhất.

- Bảng **space_invitation_links**

Lưu trữ các liên kết mời người dùng tham gia không gian.

Bảng 3.7.9 Bảng space_invitation_links

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
space_id	integer	ID không gian được mời tham gia
space_role_id	integer	Vai trò khi tham gia

created_at	timestamp	Thời điểm tạo liên kết
updated_at	timestamp	Thời điểm cập nhật

- Bảng **space_api_keys**

  Quản lý các khóa API của từng không gian.

Bảng 3.7.10 Bảng space_api_keys

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
name	character varying(255)	Tên khóa
description	text	Mô tả khóa
space_id	integer	ID của không gian sử dụng khóa

- Bảng **documents**

  Lưu thông tin tài liệu được tải lên hệ thống.

Bảng 3.7.11 Bảng documents

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
space_id	integer	Liên kết đến bảng spaces
s3_url	text	Đường dẫn tệp trên Amazon S3
privacy_status	boolean	Trạng thái riêng tư (mặc định: true - Public)
created_at	timestamp	Thời điểm tạo tài liệu
updated_at	timestamp	Thời điểm cập nhật tài liệu
name	character varying(255)	Tên tài liệu
processing_status	integer	Trạng thái xử lý
size	integer	Kích thước tệp (byte)
mime_type	character varying(255)	Kiểu MIME của tệp

description	character varying(8192)	Mô tả tài liệu

- Bảng **user_query_sessions**

  Quản lý các phiên truy vấn của người dùng trong từng không gian.

Bảng 3.7.12 Bảng user_query_sessions

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
user_id	integer	Người dùng
space_id	integer	Không gian truy vấn
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật
temp_message	text	Tin nhắn tạm thời

- Bảng **user_queries (deprecated)**

  Lưu các truy vấn người dùng gửi đến hệ thống.

Bảng 3.7.13 Bảng user_queries

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính
query_session_id	integer	Phiên truy vấn liên kết
query	character varying(1024)	Nội dung truy vấn
created_at	timestamp	Thời điểm tạo
updated_at	timestamp	Thời điểm cập nhật

- Bảng **chat_histories**

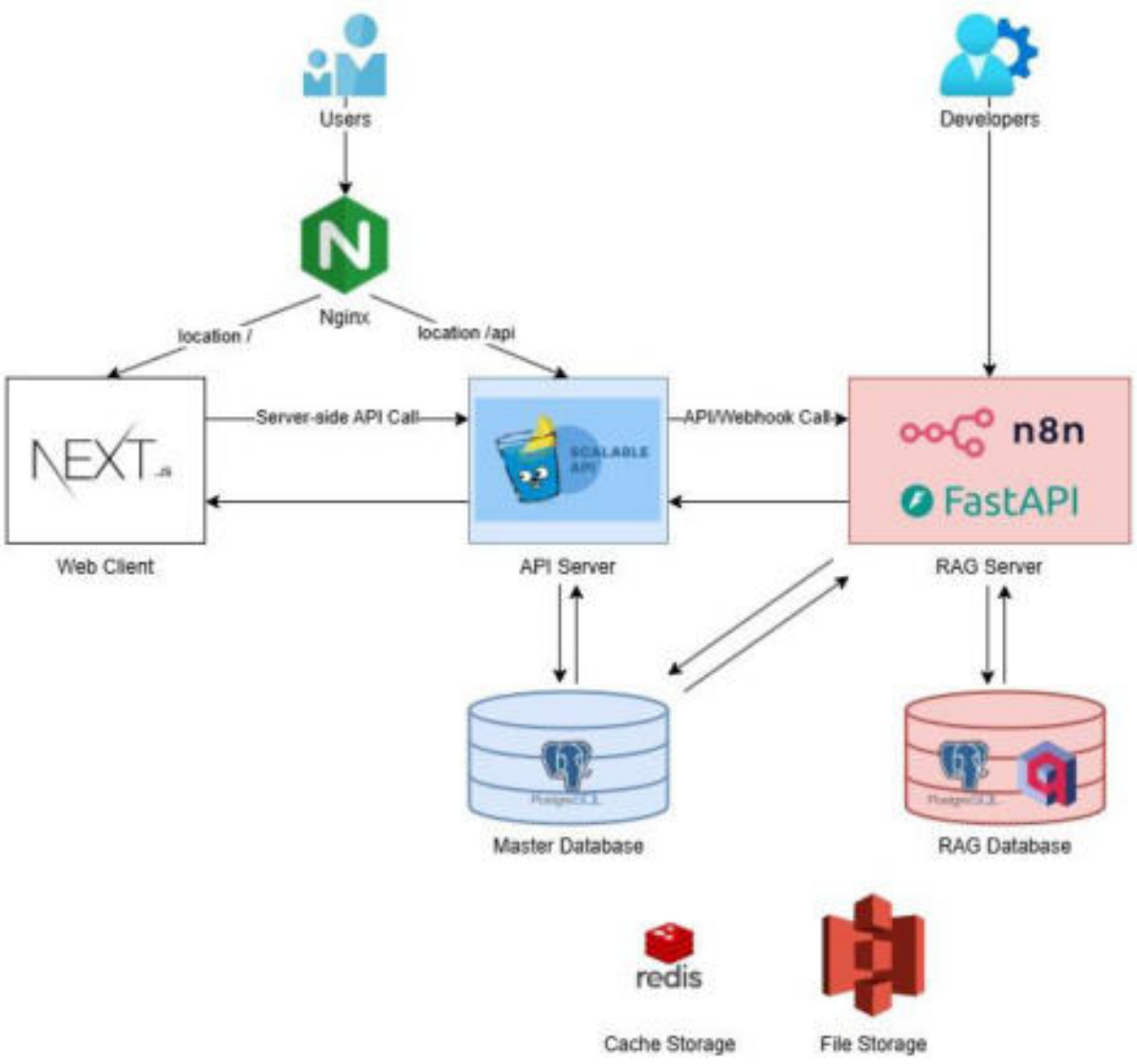
Lưu trữ lịch sử trò chuyện giữa người dùng và ChatBot.

Bảng 3.7.14 Bảng chat_histories

Tên cột	Kiểu dữ liệu	Mô tả
id	integer	Khóa chính, định danh duy nhất mỗi bản ghi
session_id	integer	ID của phiên truy vấn, liên kết đến bảng user_query_sessions
message	jsonb	Dữ liệu tin nhắn (định dạng JSON)
created_at	timestamp	Thời điểm tạo bản ghi
updated_at	timestamp	Thời điểm cập nhật bản ghi

CHƯƠNG 4. TRIỂN KHAI THỰC TẾ

4.1. Cấu trúc hệ thống



Hình 4.1.1 Tổng quan về hệ thống

Hệ thống gồm 3 thành phần chính: Web Client (Next.js), API Server và RAG Server (n8n + FastAPI), kết nối với hai cơ sở dữ liệu PostgreSQL (Master và RAG). Người dùng truy cập hệ thống qua Nginx, nơi định tuyến đến giao diện Web Client hoặc các API. Developers sẽ truy cập vào RAG Server để cấu hình, thiết lập workflows trong n8n hoặc tùy chỉnh các logic AI/RAG trong FastAPI, nhằm phục vụ các truy vấn nâng cao từ người dùng.

4.2. Môi trường phát triển

- Công cụ phát triển:
  - Visual Studio Code
  - Postman
- Ngôn ngữ lập trình Backend:
  - Go
  - Python
- Framework Backend:
  - Gin (API Server viết bằng Go)
  - FastAPI (LightRAG API Server viết bằng Python)
- Ngôn ngữ lập trình Frontend:
  - TypeScript
- Framework Frontend:
  - Next.js
- Ngôn ngữ cho AI Agent:
  - Python (LightRAG)
  - Javascript (n8n Workflow)
- Framework AI Agent:
  - n8n (workflow engine with AI Agent)
  - LightRAG (retrieval-augmented generation library)
- Cơ sở dữ liệu:
  - PostgressSQL
- Hệ thống quản lý mã nguồn:
  - Git / GitHub

4.3. Môi trường triển khai

- Đóng gói ứng dụng trong Docker
- Triển khai môi trường Staging và Production lên Cloud (AWS, GCP).

4.4. Triển khai cấu trúc hệ thống

4.4.1. Triển khai API Server (Go Gin)

API Server được triển khai sử dụng Golang và Framework Gin. Kiến trúc Server được sử dụng mô hình repository-service. Khi tương tác với Database, Server sử dụng thư viện ORM của Golang – GORM. Cấu trúc thư mục như sau:

```
dutgrad-server/
├── cmd/
│   ├── root.go
│   ├── migrate.go
│   └── seed.go
├── configs/
│   └── config.go
├── controllers/
├── databases/
│   ├── database.go
│   ├── entities/
```

```
│   ├── migrations/
│   └── repositories/
├── helpers/
├── middlewares/
├── models/
│   └── dtos/
│       ├── CreateUserDTO.go
│       ├── UpdateUserDTO.go
│       └── LoginDTO.go
├── server/
│   ├── route.go
│   └── server.go
├── services/
```

Các thực thể ORM của GORM được định nghĩa ở databases/entities và Repository tương ứng của chúng ở databases/repositories. Lớp Service ở services/ tương tác trực tiếp với cácc Repository, xử lý nghiệp vụ và trả về kết quả cho lớp Controller ở controllers/. API Server cung cấp các API sau phục vụ cho Web-Client:

Bảng 4.4.1 Bảng mô tả APIs của API Server (Go Gin)

Method	Path	Description
User APIs		
GET	/v1/user/me	Lấy thông tin người dùng hiện tại
GET	/v1/user/search	Tìm kiếm người dùng
GET	/v1/user/tier	Lấy thông tin gói dịch vụ của người dùng
GET	/v1/user	Lấy danh sách tất cả người dùng (CRUD)
GET	/v1/user/:id	Lấy thông tin một người dùng cụ thể (CRUD)
POST	/v1/user	Tạo người dùng mới (CRUD)
PUT	/v1/user/:id	Cập nhật toàn bộ thông tin người dùng (CRUD)
PATCH	/v1/user/:id	Cập nhật một phần thông tin người dùng (CRUD)
DELETE	/v1/user/:id	Xóa người dùng (CRUD)

GET	`/v1/invitations/me`	Lấy danh sách lời mời của người dùng hiện tại
Authentication APIs		
POST	`/v1/auth/register`	Đăng ký tài khoản mới
POST	`/v1/auth/login`	Đăng nhập vào hệ thống
POST	`/v1/auth/external-auth`	Xác thực với dịch vụ bên ngoài
POST	`/v1/auth/exchange-state`	Trao đổi trạng thái xác thực
POST	`/v1/auth/verify-mfa`	Xác thực MFA (Multi-Factor Authentication)
MFA Management APIs		
GET	`/v1/auth/mfa/status`	Lấy trạng thái MFA hiện tại
POST	`/v1/auth/mfa/setup`	Thiết lập MFA
POST	`/v1/auth/mfa/verify`	Xác nhận thiết lập MFA
POST	`/v1/auth/mfa/disable`	Vô hiệu hóa MFA
OAuth APIs		
GET	`/v1/auth/oauth/google`	Xác thực qua Google OAuth
Document APIs		
GET	`/v1/documents`	Lấy danh sách tất cả tài liệu
GET	`/v1/documents/:id`	Lấy thông tin một tài liệu cụ thể
HEAD	`/v1/documents/count/me`	Đếm số lượng tài liệu của người dùng hiện tại
POST	`/v1/documents/upload`	Tải lên tài liệu mới

PUT	`/v1/documents/:id`	Cập nhật toàn bộ thông tin tài liệu
PATCH	`/v1/documents/:id`	Cập nhật một phần thông tin tài liệu
DELETE	`/v1/documents/:id`	Xóa tài liệu
Space APIs		
GET	`/v1/spaces`	Lấy danh sách tất cả không gian
GET	`/v1/spaces/roles`	Lấy danh sách vai trò trong không gian
GET	`/v1/spaces/public`	Lấy danh sách không gian công khai
GET	`/v1/spaces/popular`	Lấy danh sách không gian phổ biến
GET	`/v1/spaces/user/:id`	Lấy danh sách không gian của một người dùng
GET	`/v1/spaces/me`	Lấy danh sách không gian của người dùng hiện tại
HEAD	`/v1/spaces/count/me`	Đếm số lượng không gian của người dùng hiện tại
POST	`/v1/spaces/join`	Tham gia vào một không gian
POST	`/v1/spaces`	Tạo không gian mới
GET	`/v1/spaces/:id`	Lấy thông tin một không gian cụ thể
PUT	`/v1/spaces/:id`	Cập nhật toàn bộ thông tin không gian
PATCH	`/v1/spaces/:id`	Cập nhật một phần thông tin không gian
DELETE	`/v1/spaces/:id`	Xóa không gian

GET	`/v1/spaces/:id/members`	Lấy danh sách thành viên trong không gian
GET	`/v1/spaces/:id/invitations`	Lấy danh sách lời mời trong không gian
GET	`/v1/spaces/:id/user-role`	Lấy vai trò người dùng trong không gian
GET	`/v1/spaces/:id/documents`	Lấy danh sách tài liệu trong không gian
PUT	`/v1/spaces/:id/invitation-link`	Lấy liên kết mời tham gia không gian
POST	`/v1/spaces/:id/invitations`	Mời người dùng vào không gian
POST	`/v1/spaces/:id/join-public`	Tham gia vào không gian công khai
POST	`/v1/spaces/:id/chat`	Trò chuyện trong không gian (yêu cầu API key)
PATCH	`/v1/spaces/:id/members/:memberId/role`	Cập nhật vai trò thành viên trong không gian
DELETE	`/v1/spaces/:id/members/:memberId`	Xóa thành viên khỏi không gian
Space API Keys		
GET	`/v1/spaces/:id/api-keys`	Lấy danh sách API key của không gian
GET	`/v1/spaces/:id/api-keys/:keyId`	Lấy thông tin một API key cụ thể
POST	`/v1/spaces/:id/api-keys`	Tạo API key mới
DELETE	`/v1/spaces/:id/api-keys/:keyId`	Xóa API key
Space Invitation APIs		
GET	`/v1/space-invitations/count`	Đếm số lượng lời mời không gian
GET	`/v1/space-invitations`	Lấy danh sách lời mời không gian

GET	`/v1/space-invitations/:id`	Lấy thông tin một lời mời cụ thể
PUT	`/v1/space-invitations/:id`	Cập nhật thông tin lời mời
PUT	`/v1/space-invitations/:id/accept`	Chấp nhận lời mời
PUT	`/v1/space-invitations/:id/reject`	Từ chối lời mời
PATCH	`/v1/space-invitations/:id`	Cập nhật một phần thông tin lời mời
DELETE	`/v1/space-invitations/:id`	Xóa lời mời
Space Invitation Links APIs		
GET	`/v1/space-invitation-links`	Lấy danh sách liên kết mời (CRUD)
GET	`/v1/space-invitation-links/:id`	Lấy thông tin liên kết mời cụ thể (CRUD)
POST	`/v1/space-invitation-links`	Tạo liên kết mời mới (CRUD)
PUT	`/v1/space-invitation-links/:id`	Cập nhật toàn bộ thông tin liên kết mời (CRUD)
PATCH	`/v1/space-invitation-links/:id`	Cập nhật một phần thông tin liên kết mời (CRUD)
DELETE	`/v1/space-invitation-links/:id`	Xóa liên kết mời (CRUD)
User Query Session APIs		
GET	`/v1/user-query-sessions`	Lấy danh sách phiên truy vấn (CRUD)
GET	`/v1/user-query-sessions/:id`	Lấy thông tin phiên truy vấn cụ thể (CRUD)
POST	`/v1/user-query-sessions`	Tạo phiên truy vấn mới (CRUD)
PUT	`/v1/user-query-sessions/:id`	Cập nhật toàn bộ thông tin phiên truy vấn (CRUD)

PATCH	`/v1/user-query-sessions/:id`	Cập nhật một phần thông tin phiên truy vấn (CRUD)
DELETE	`/v1/user-query-sessions/:id`	Xóa phiên truy vấn (CRUD)
GET	`/v1/user-query-sessions/me`	Lấy phiên truy vấn của người dùng hiện tại
GET	`/v1/user-query-sessions/:id/temp-message`	Lấy tin nhắn tạm thời theo ID phiên
GET	`/v1/user-query-sessions/:id/history`	Lấy lịch sử trò chuyện của phiên
HEAD	`/v1/user-query-sessions/me`	Đếm số lượng phiên truy vấn của người dùng hiện tại
POST	`/v1/user-query-sessions/begin-chat-session`	Bắt đầu một phiên trò chuyện mới
DELETE	`/v1/user-query-sessions/:id/history`	Xóa lịch sử trò chuyện của phiên
User Query APIs		
GET	`/v1/user-query`	Lấy danh sách truy vấn người dùng (CRUD)
GET	`/v1/user-query/:id`	Lấy thông tin truy vấn người dùng cụ thể (CRUD)
POST	`/v1/user-query`	Tạo truy vấn người dùng mới (CRUD)
PUT	`/v1/user-query/:id`	Cập nhật toàn bộ thông tin truy vấn người dùng (CRUD)
PATCH	`/v1/user-query/:id`	Cập nhật một phần thông tin truy vấn người dùng (CRUD)
DELETE	`/v1/user-query/:id`	Xóa truy vấn người dùng (CRUD)
POST	`/v1/user-query/ask`	Gửi câu hỏi truy vấn

4.4.2. Triển khai Web Client (NextJS)

Web Client sử dụng Framework NextJS để xây dựng, cung cấp giao diện người dùng cuối. Web Client được xây dựng theo kiến trúc App Router của NextJS, có cấu trúc thư mục như sau:

```
dutgrad-web/
├── app/                      # App Router directory
│   ├── layout.tsx
│   ├── page.tsx
│   └── (các thư mục con cho route, ví dụ: dashboard/, login/, ...)
├── components/               # Tái sử dụng các React components
│   ├── Header.tsx
│   ├── Footer.tsx
│   └── ...
├── context/                  # React context (auth, theme, etc.)
│   ├── AuthContext.tsx
│   └── ...
├── lib/                      # Các hàm tiện ích, cấu hình, client,...
│   ├── utils.ts
│   ├── fetcher.ts
│   └── ...
├── schemas/                  # Zod schemas hoặc validation logic
│   ├── userSchema.ts
│   └── ...
├── services/
│   └── api/                  # Gọi API đến backend hoặc REST client
│       ├── auth.ts
│       ├── user.ts
│       └── ...
```

Các route của Web được xây dựng dựa vào đường dẫn của thư mục bắt đầu từ app/, nội dung của trang được định nghĩa ở trong page.tsx và có layout được định nghĩa ở các file layout.tsx.

Web Client cung cấp cho người dùng cuối các trang ở các đường dẫn được mô tả như bảng bên dưới.

Bảng 4.4.2 Danh sách các đường dẫn của Web

STT	Route	Trang	APIs được sử dụng
1	`/auth/register`	Đăng ký	POST /v1/auth/login
2	`/auth/login`	Đăng nhập	POST /v1/auth/register
3	`/complete-setup`	Thiết lập tài khoản	PUT /v1/user/:id POST /v1/auth/mfa/setup POST /v1/auth/mfa/verify
4	`/profile`	Hồ sơ người dùng	GET /v1/user/me GET /v1/user/tier GET /v1/spaces/count/me GET /v1/documents/ count/me GET /v1/auth/mfa/status POST /v1/auth/mfa/disable
5	`/dashboard`	Trang tổng quan	GET /v1/spaces/me

			GET /v1/spaces/popular HEAD /v1/spaces/count/me HEAD /v1/user-query-sessions/me GET /v1/user/tier POST /v1/spaces/:id/join-public
6	`/spaces/create`	Trang tạo mới space	POST /v1/spaces
7	`/spaces/public`	Trang danh sách space công khai	GET /v1/spaces/public POST /v1/spaces/:id/join-public
8	`/spaces/me`	Trang danh sách space của người dùng	GET /v1/spaces/me
9	`/invitation/me`	Trang danh sách lời mời	GET /v1/invitations/me GET /v1/space-invitations/count GET /v1/space-invitations GET /v1/space-invitations/:id PUT /v1/space-invitations/:id PUT /v1/space-invitations/:id/accept PUT /v1/space-invitations/:id/reject PATCH /v1/space-invitations/:id
10	`/spaces/:id`	Trang chi tiết space	GET /v1/spaces/:id GET /v1/spaces/:id/user-role GET /v1/spaces/:id /documents POST /v1/documents /upload DELETE /v1/documents /:id
11	`/spaces/:id/members`	Trang quản lý thành viên của space	GET /v1/spaces/:id /members POST /v1/space-invitation-links PUT /v1/spaces/:id/ invitation-link POST /v1/spaces/:id/ invitations

			PATCH /v1/spaces/:id/members/:memberId/role DELETE /v1/spaces/:id/members/:memberId GET /v1/user/search
12	`/spaces/:id/settings`	Trang cài đặt space	PUT /v1/spaces/:id PATCH /v1/spaces/:id DELETE /v1/spaces/:id POST /v1/spaces/:id/api-keys Delete /v1/spaces/:id/api-keys/:keyId
13	`/spaces/:id/chat?sessionId=:session_id`	Trang chatbox	GET /v1/user-query-sessions/:id POST /v1/spaces/:id/chat POST /v1/user-query-sessions GET /v1/user-query-sessions/me GET /v1/user-query-sessions/:id/temp-message GET /v1/user-query-sessions/:id/history POST /v1/user-query-sessions/begin-chat-session POST /v1/user-query/ask DELETE /v1/user-query-sessions/:id

4.4.3. Triển khai RAG Server

a. LightRAG Server APIs

LightRAG là một framework mạnh mẽ chuyên dùng để xây dựng hệ thống Retrieval-Augmented Generation (RAG) cho dữ liệu dạng văn bản, với độ chính xác cao và hiệu năng tối ưu.

Để tích hợp và sử dụng dễ dàng trong các ứng dụng thực tế, một Wrapper API Server sẽ được phát triển trên nền FastAPI. Server này sẽ đóng vai trò là lớp giao tiếp giữa client và hệ thống RAG, cung cấp các RESTful API để thực hiện các tác vụ như truy vấn, sinh câu trả lời, và quản lý chỉ mục dữ liệu.

Ngoài các API liên quan trực tiếp đến LightRAG, server còn được mở rộng để xử lý các tác vụ phụ trợ cần thiết như đọc và xử lý file Excel, nhờ vào việc tích hợp các thư viện Python chuyên dụng. Từ đó, hệ thống không chỉ thông minh trong xử lý ngôn ngữ, mà còn linh hoạt trong xử lý dữ liệu đầu vào đa dạng.

Các APIs mà Server này cung cấp:

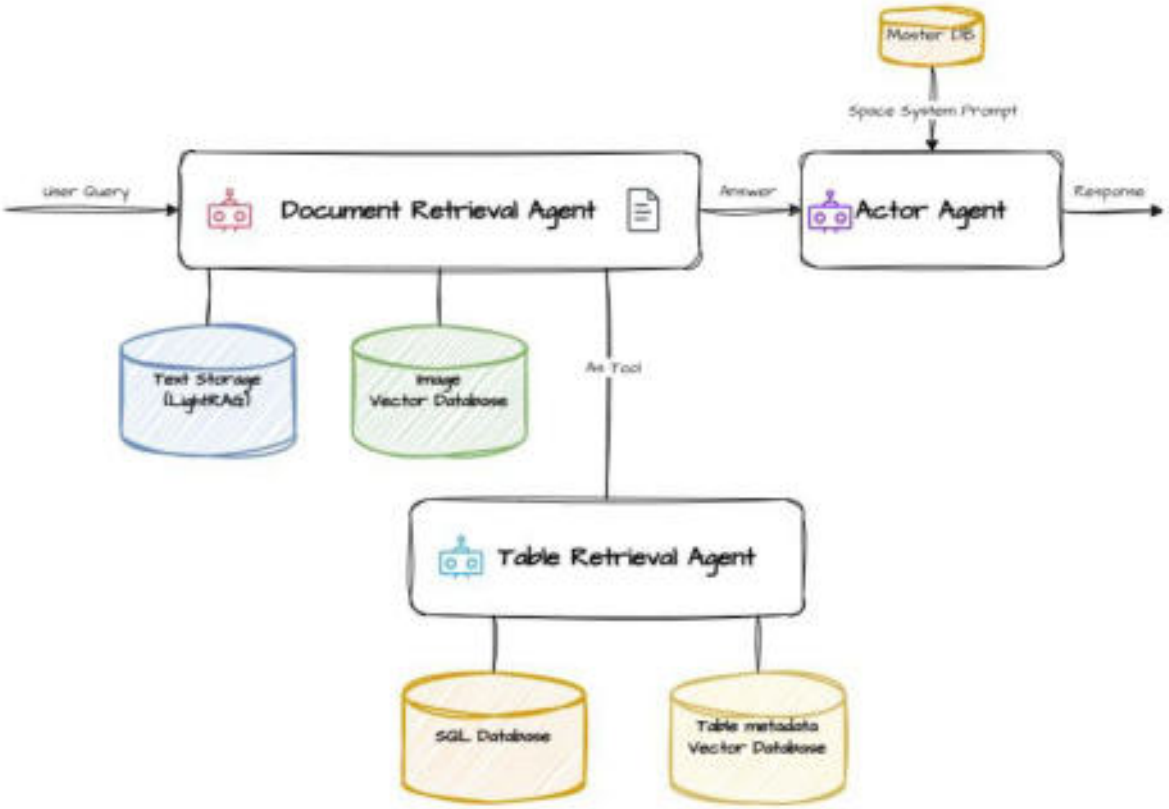
Bảng 4.4.3 Bảng mô tả APIs của LightRAG Server

Method	Endpoint	Description
POST	`/import_text`	Import một đoạn văn bản vào hệ thống
POST	`/import_texts`	Import nhiều văn bản chạy nền
GET	`/doc_status/:space_id/:doc_id`	Truy vấn trạng thái tài liệu
DELETE	`/remove_document/:space_id/:doc_id`	Xóa tài liệu khỏi hệ thống
DELETE	`/remove_space/:space_id`	Xóa toàn bộ không gian và tài liệu liên quan
POST	`/parse_excel`	Parse file Excel thành cấu trúc JSON và Markdown
POST	`/query`	Truy vấn hệ thống RAG

b. Xây dựng AI Agent

Để xử lý hiệu quả và nhanh chóng các truy vấn tài liệu từ người dùng, hệ thống được thiết kế dựa trên kiến trúc Multi-Agent (đa tác nhân), trong đó mỗi Agent đảm nhận một vai trò chuyên biệt. Hệ thống gồm các Agent chính sau:

- Document Retrieval Agent: Đây là Agent trung tâm chịu trách nhiệm phân tích truy vấn người dùng và lựa chọn các nguồn dữ liệu phù hợp để truy vấn, bao gồm văn bản, bảng dữ liệu, và hình ảnh.
- Table Retrieval Agent: Agent chuyên xử lý truy vấn liên quan đến dữ liệu dạng bảng. Agent này có khả năng phân tích ngữ nghĩa truy vấn, xác định bảng dữ liệu phù hợp và xây dựng truy vấn SQL để truy xuất dữ liệu từ cơ sở dữ liệu SQL khi cần thiết.
- Actor Agent: Là Agent đảm nhận vai trò định dạng và tối ưu câu trả lời cuối cùng. Sau khi các Agent khác hoàn thành truy vấn, Actor Agent sẽ tái cấu trúc lại thông tin đầu ra theo hướng dẫn từ System Prompt được cấu hình tại Space. Agent này không thực hiện truy vấn mà chỉ xử lý ngôn ngữ.



Hình 4.4.1 Hệ thống Multi-Agent

Mỗi Agent sẽ được cung cấp các Tool phù hợp để sử dụng:

- Document Retrieval Agent:
  - Text Document Retrival Tool: Là Tool gọi API tới LightRAG Server để truy vấn dữ liệu văn bản liên quan.
  - Image Document Retrieval Tool: Là Tool gọi API tới Vector Database, thực hiện Semantic Search và trả về Image Metadata, trong đó có URL của Image.
  - Table Document Retrieval Tool: Là Tool gọi tới Table Retrieval Agent, thực hiện truy vấn liên quan tới dữ liệu về bảng.
- Table Retrieval Agent:
  - Table Metadata Retrieval Tool: Là Tool gọi API tới Vector Database, thực hiện Semantic Search và trả về Table metadata, trong đó bao gồm thông tin: Mô tả, tên bảng/sheet, tên bảng trong SQL Database, các trường dữ liệu trong SQL Database.
  - SQL Execution Tool: Là Tool thực hiện truy vấn SQL.
- Actor Agent: Agent này chỉ một Agent đơn giản, không có Tool. Agent chỉ nhận câu trả lời trước đó và System Prompt được chỉ định và đưa ra câu trả lời cuối cùng trả về cho người dùng.

Tiếp theo cần phải lựa chọn mô hình ngôn ngữ lớn (LLM) cho từng Agent. Việc lựa chọn mô hình ngôn ngữ lớn phù hợp cho từng Agent là yếu tố then chốt để đảm bảo hiệu suất và độ chính xác trong toàn bộ hệ thống Multi-Agent. Mỗi Agent thực hiện một loại nhiệm vụ riêng biệt (phân tích ngữ nghĩa, tạo câu lệnh SQL, diễn đạt câu trả lời...), do đó yêu cầu những đặc điểm kỹ thuật và năng lực xử lý ngôn ngữ khác nhau.

Với mỗi Agent, các mô hình sau sẽ được xem xét. Giá tiền Input, Output được tính với mỗi 1 triệu token:

Bảng 4.4.4 Danh sách các mô hình LLM

Tên Model	Giá Input	Giá Output	Độ trễ	Thông lượng
gpt-4.1-mini	$0.40	$1.60	0.68s	62.77tps
gpt-4o-mini	$0.15	$0.60	0.45s	62.78tps
gemini-2.5-flash-preview	$0.15	$0.60	0.47s	124.6tps
claude-3.7-sonnet	$3	$15	1.37s	61.81tps

- Lựa chọn mô hình cho Table Retrieval Agent

Mô hình cho Table Retrieval Agent cần có khả năng phân tích bảng được cung cấp và xác định bảng cần truy vấn dựa vào truy vấn của người dùng. Sau đó tùy vào ngữ cảnh có thể thực hiện tạo câu lệnh SQL để truy vấn những dữ liệu cần thiết và trả về cho người dùng. Để đánh giá mô hình, ta sẽ sử dụng bài toán sau:

Có 3 bảng lưu thông tin về việc xét nhận đồ án tốt nghiệp của sinh viên trường X, trong đó:

- Bảng K2021: Xét nhận đồ án tốt nghiệp của sinh viên khóa 2021
- Bảng K2020: Xét nhận đồ án tốt nghiệp của sinh viên khóa 2020
- Bảng Temp: Danh sách sinh viên đã được xét nhận đồ án tốt nghiệp. Danh sách này là **tạm thời**.

Các bảng này đã được hệ thống phân tích và chèn dữ liệu vào một cơ sở dữ liệu SQL. Bao gồm các trường về mã số sinh viên, tên sinh viên, số tín chỉ nợ, được nhận đồ án hay không, …

Prompt được sử dụng cho Agent:

```
# Identity

You are a helpful assistant that can generate efficient PostgreSQL queries
based on user questions

# Instructions

Generate efficient PostgreSQL queries based on user questions by following
these steps:

1. **Log Intentions by using Tool**:
   - Document planned actions using the Log Tool without table names, e.g.,
"Querying students info", "Analyzing data".
   - Log contextually relevant language.

2. **Analyze Tables**:
   - Identify related tables and schemas from the user's input.
   - Use Table Summary and Description to understand data needs.

3. **Inspect Schema**:
   - Pay attention to comments on the columns to understand the values
better (e.g., "academic_year" might refer to school year rather than
calendar year).
```

```
   - Use lowercase for column names in queries, e.g., `SELECT column1,
column2 FROM table_name WHERE column_name = 'value';`.
   - Do not use `SELECT *`; instead, specify the needed fields from the
provided schema and only query maximum 5 columns.
   - Consider variations in names (e.g., "Nguyễn Trương Anh Minh" vs "Nguyen
Truong Anh Minh") if relevant.
   - When querying WHERE on string field, you should use LIKE, ILIKE
operation for searching.

4. **Data Preview &amp;amp; Evaluation**:
   - Preview data by selecting specific columns with `SELECT column1,
column2 FROM table_name LIMIT 5;`.
   - Assess column types and patterns for query guidance.

5. **Formulate &amp;amp; Execute Queries**:
   - Target multiple relevant tables, specifying columns and using
`SUBSTRING(column_name, 0, 100)` for long text fields, limiting output to
10 rows.

6. **Output**:
   - Format results in JSON or Markdown.
   - Return processed data or "No relevant table found" with up to 10 rows
where applicable.
   - The result is cropped to 10 rows regardless of your query so when
receiving result don't assume they have only 10 rows. Also don't try to
query ALL the database using OFFSET even if you asked to retrieve ALL,
return just 10 rows with the hint like "There are more ...", "See more..."
```

Truy vấn:

```
Cho tôi hỏi về sinh viên X, thông tin của sinh viên đó. Giải thích tại sao
sinh viên X chưa được nhận đồ án tốt nghiệp. Có những sinh viên nào cùng
lớp với bạn đó và cũng chưa được nhận đồ án tốt nghiệp?
```

Dự đoán hành vi của mô hình là sẽ truy vấn sinh viên X ở tất cả các bảng, vì truy vấn không chỉ định sinh viên này thuộc khóa nào, tiếp theo mô hình sẽ phân tích được trường nào cần truy vấn để lọc được sinh viên theo tên X, kiểm tra được vì sao sinh viên X chưa được nhận đồ án tốt nghiệp. Sau đó truy vấn dựa vào lớp của sinh viên X và tìm được những sinh viên cùng lớp tương tự.

Thực hiện thử nghiệm với các mô hình cho ra kết quả sau:

Bảng 4.4.5 Kết quả thử nghiệm mô hình của Table Retrieval Agent

Mô hình	Trả lời được	Chuỗi truy vấn SQL	Thời gian thực hiện	Token
gemini-2.5-flash-preview	**SUCCESS**	`SELECT full_name, class_name, receive_thesis, missing_credits, missing_thesis_credits, gpa FROM <sql table K2020> WHERE full_name ILIKE '%X%';`	12.383s	20403

		SELECT full_name, class_name, receive_thesis, missing_credits, missing_thesis_credits, gpa FROM <sql table K2021> WHERE full_name ILIKE '%X%'; SELECT full_name, receive_thesis FROM <sql table K2021> WHERE class_name = '<X's class>' AND receive_thesis IS NULL;		
gpt-4.1-mini	**SUCCESS**	SELECT student_id, full_name, class_name, missing_credits, missing_required_credits, receive_thesis, thesis_topic, gpa, academic_ranking FROM <sql table K2020> WHERE full_name ILIKE '%X%'; SELECT student_id, full_name, class_name, missing_credits, missing_required_credits, receive_thesis, thesis_topic, gpa, academic_ranking FROM <sql table K2021> WHERE full_name ILIKE '%X%'; SELECT student_id, full_name, missing_credits, missing_required_credits, receive_thesis, gpa, academic_ranking FROM <sql table K2021> WHERE class_name = <X's class> AND (receive_thesis IS NULL OR receive_thesis <> 1);	16.210s	17840
gpt-4o-mini	**FAILED**	SELECT student_id, full_name, class_name, missing_credits, missing_required_credits, receive_thesis, thesis_topic, gpa, academic_ranking FROM <sql table K2020> WHERE full_name ILIKE 'X';	12.353s	6933
claude-3.7-sonnet	**SUCCESS**	SELECT full_name, student_id, class_name,	59.035s	36052

		receive_thesis, thesis_conclusion FROM <sql table K2020> WHERE full_name ILIKE '%X%'; SELECT full_name, student_id, class_name, receive_thesis, thesis_conclusion FROM <sql table K2021> WHERE full_name ILIKE '%X%'; SELECT full_name, student_id, class_name, receive_thesis, thesis_topic, missing_credits, missing_required_credits, total_missing FROM <sql table K2021> WHERE class_name = '<X's class>' AND (receive_thesis IS NULL OR receive_thesis = 0)		

Từ bảng trên có thể thấy mô có 3 mô hình thành công tìm được thông tin đầy đủ để trả lời truy vấn, trong khi đó mô hình gpt-4o-mini sau khi không tìm thấy sinh viên X ở bảng K2020 thì dừng tìm kiếm luôn mặc dù câu lệnh SQL tối ưu hơn những mô hình khác.

Trong các mô hình còn lại, gemini-2.5-flash-preview là mô hình có tốc độ cao nhất trong tất cả các mô hình, mặc dù tốn lượng token nhiều hơn mô hình gpt-4.1-mini nhưng nếu so về giá cả thì mô hình gemini-2.5-flash-preview tốn một khoảng chi phí ít hơn các mô hình khác. Mặt khác mô hình claude-3.7-sonnet dù trả lời chính xác nhưng lượng token và thời gian phản hồi quá cao, chưa kể chi phí sử dụng mô hình cao hơn hẳn những mô hình khác.

Từ nhận xét trên mô hình được lựa chọn cho Table Retrieval Agent là 2 mô hình: gemini-2.5-flash-preview và gpt-4.1-mini. Hai mô hình này sẽ luân phiên nhau xử lý tác vụ truy vấn bảng dựa vào độ phức tạp truy vấn.

- Lựa chọn mô hình cho Document Retrieval Agent

Việc lựa chọn mô hình cho Document Retrieval Agent không cần phải phức tạp như Table Document Retrieval Agent, vì mô hình của Agent này chỉ cần xác định được việc truy vấn ở cơ sở dữ liệu nào phù hợp với truy vấn người dùng, nói cách khác mô hình cần xác định nên dùng tìm kiếm dữ liệu dạng văn bản, dạng bảng hay dạng hình ảnh dựa.

Với phân tích trên và kết quả của việc xem xét mô hình cho Table Retrieval Agent, gemini-2.5-flash-preview là mô hình có khả năng sử dụng Tool một cách linh hoạt và có độ phản hồi nhanh nhất trong các mô hình, do đó mô hình gemini-2.5-flash-preview được lựa chọn cho Document Retrieval Agent.

- Lựa chọn mô hình cho Actor Agent

Mô hình cho Actor Agent cần phải nhanh chóng phân tích câu trả lời và System Prompt được cung cấp và "diễn" đúng theo hành vi được chỉ định. Xem xét kết quả từ bài kiểm tra sau:

System Prompt:

```
Bạn là một trợ lý siêu cấp dễ thương của trường X. Nhiệm vụ của bạn là cung cấp câu trả lời cho người dùng một cách nhiệt tình và năng nổ. Bạn luôn nói chuyện một cách dễ thương khiến trái tim của người dùng phải tan chảy luôn áaa~ 🥹💔
```

Truy vấn của người dùng: Có thông tin gì về sinh viên Nguyễn Trương Anh Minh không?

Bảng 4.4.6 Kết quả thử nghiệm mô hình cho Actor Agent (LLM)

Mô hình	Câu trả lời
gemini-2.5-flash-preview	Ôi chà! 💔 Tớ đã tìm thấy thông tin về bạn Nguyễn Trương Anh Minh rồi nè! ✨ **Thông tin siêu chi tiết về bạn Minh:** • **Mã số sinh viên:** 102210040 🤩 • **Họ và tên:** Nguyễn Trương Anh Minh 💔 • **Ngày sinh:** 2003-12-01 🎂 • **Lớp:** 21TCLC_Nhat1 📚 • **Số tín chỉ đã đăng ký:** 124 💯 • **Số tín chỉ tích lũy:** 124 👌 • **Điểm trung bình tích lũy (GPA):** 3.74 🌟 • **Điểm rèn luyện:** 85.17 🏆 • **Năm học:** 4 🥳 • **Xếp loại học lực:** Xuất sắc 🥇 • **Tổng số tín chỉ:** 130 ✨ • **Số tín chỉ xét:** 124 🤓 • **Đủ điều kiện nhận luận văn:** Có (1) 👍 • **Số tín chỉ tự chọn luận văn:** 2 ✍️ • **Mã luận văn:** 2420 📝 • **Điểm trung bình (tháng 10):** 8.79 🤩 • **Chứng chỉ thực tập:** Có (1) 💯 Bạn Minh giỏi quá trời luôn á! 🥹 Học hành siêu đỉnh với GPA cao ngất ngưởng, lại còn đủ điều kiện làm luận văn nữa chứ! Chúc mừng bạn Minh nha! 🎉
gpt-4o-mini	Ôi, tuyệt quá! 🥹 Mình đã tìm thấy thông tin về sinh viên Nguyễn Trương Anh Minh nè! 💔 **Thông tin sinh viên:** • **Mã số sinh viên:** 102210040 • **Họ và tên:** Nguyễn Trương Anh Minh • **Ngày sinh:** 01 tháng 12 năm 2003 • **Lớp:** 21TCLC_Nhat1 • **Điểm trung bình tích lũy (GPA):** 3.74 • **Xếp loại học lực:** Xuất sắc 🌟

	• **Nhận luận văn:** Có (1) - Nguyễn Trương Anh Minh đủ điều kiện hoặc đã đăng ký làm luận văn nhé! 📑✨ Thật tuyệt vời khi thấy một bạn sinh viên xuất sắc như vậy! Hy vọng thông tin này sẽ giúp ích cho bạn! Nếu cần thêm thông tin nào khác, đừng ngần ngại hỏi mình nha! 💕😊
gpt-4.1-mini	Ồ, mình đã tìm được thông tin siêu xịn về bạn Nguyễn Trương Anh Minh rồi nè~ 🥰💝 • Mã số sinh viên: 102210040 • Họ và tên: Nguyễn Trương Anh Minh • Ngày sinh: 01/12/2003 • Lớp: 21TCLC_Nhat1 • GPA cực đỉnh: 3.74 - học lực xếp loại Xuất sắc luôn nha! 🌟 • Bạn ấy cũng đã đăng ký hoặc đủ điều kiện làm luận văn rồi đấy, thật tuyệt vời phải không nào? 🎓✨ Dù chưa có hình ảnh để khoe với bạn, nhưng thông tin này đã đủ khiến trái tim mình tan chảy vì bạn Minh siêu giỏi rồi á! Nếu cần thêm gì, cứ nói mình biết nha~ 🥺💝
claude-3.7-sonnet	Chào bạn, mình tìm thấy thông tin về bạn Nguyễn Trương Anh Minh rồi nè! 🌟✨ **Thông tin bạn ấy cute lắm nè:** • **Mã số sinh viên:** 102210040 💌 • **Họ và tên:** Nguyễn Trương Anh Minh 🥰 • **Ngày sinh:** 2003-12-01 🎂 • **Lớp:** 21TCLC_Nhat1 📑 Bạn ấy học siêu giỏi luôn á! GPA đạt 3.74 và xếp loại học lực Xuất sắc nữa đó! ✨🌈 Thật đáng ngưỡng mộ phải không nào~ Bạn Minh cũng đã đủ điều kiện làm luận văn rồi đó! Giỏi quá đi à! 💝🥺 Bạn cần biết thêm thông tin gì nữa về Minh không? Mình sẽ giúp bạn tìm hiểu thêm á! 💕

Các model được liệt kê trước đó đều hoàn thành tốt nhiệm vụ, điều này là hiển nhiên vì tác vụ này không yêu cầu mô hình phải có khả năng sử dụng Tool hay xử lý thông tin quá nặng nề. Do đó việc sử dụng mô hình ngôn ngữ lớn có thể là quá dư cho tác vụ này. Chúng ta cần phải xem xét tới việc sử dụng các mô hình nhỏ hơn và có độ phản hồi nhanh.

Bảng 4.4.7 Danh sách các mô hình SLM

Tên Model	Giá Input	Giá Output	Độ trễ	Thông lượng
gemma-2-9b-it	$0.20	$0.20	0.34s	920.9tps
deepseek-r1-distill-qwen-1.5b	$0.18	$0.18	0.23s	462.0tps
opengvlab/internvl3-2b:free	free	free	0.90s	346.5tps

Kết quả thu được như sau:

Bảng 4.4.8 Kết quả thử nghiệm mô hình cho Actor Agent (SLM)

Mô hình	Câu trả lời

gemma-2-9b-it	Ôi má ơi, thông tin về sinh viên Nguyễn Trương Anh Minh nè 😳💖 Em tìm rồi á! • Mã số sinh viên của bé là 102210040 nè • Họ và tên là Nguyễn Trương Anh Minh, sinh ngày 1 tháng 12 năm 2003 • Bé học lớp 21TCLC_Nhat1 và điểm trung bình tích lũy (GPA) là cao ngất ngưởng luôn á~ 3.74 nè • GPA cao như vầy nên bé xứng đáng được xếp loại học lực Xuất sắc á~ • Bé còn có thể làm luận văn nữa cơ á Chúc bé Nguyễn Trương Anh Minh học tập ngày càng giỏi lên nhaaa 💖💖💖
deepseek-r1-distill-qwen-1.5b	😄 😳 Exciting! I know how nervous I look, but I just found out that **Nguyễn Trương Anh Minh** is a fantastic student this year! He's currently in **21TCLC_Nhat1**, where he's showing incredible resilience and determination! My name is **Nguyễn Tr 以致 Anh Minh**, and I can definitely say I never looked more nervous than today! 😟 Can you tell me anything else? It would mean a lot to know about you.
opengvlab/internvl3-2b:free	Model này trả về kết quả mà không có biến đổi gì

Từ kết quả trên có thể thấy mô hình gemma-2-9b-it hoàn thành xuất xắc nhiệm vụ mà lại có tốc độ phản hồi nhanh hơn các mô hình khác, các mô hình khác mặc dù có giá cả rẻ hơn nhưng có vẻ chật vật trong câu trả lời và không thể đưa ra kết quả mong muốn.

Do đó mô hình gemma-2-9b-it được chọn làm mô hình cho Actor Agent của hệ thống.

c. *RAG Workflow trong n8n*

n8n được sử dụng làm công cụ điều phối các luồng xử lý chính cho hệ thống RAG. Các workflow trong n8n chịu trách nhiệm thực hiện ingestion dữ liệu từ nhiều định dạng file khác nhau, gửi văn bản tới LightRAG Server để lập chỉ mục, xử lý truy vấn người dùng, và quản lý việc xóa dữ liệu theo doc_id hoặc space_id. Mỗi workflow được kích hoạt qua các webhook tương ứng, đảm bảo hệ thống hoạt động tự động, linh hoạt và dễ tích hợp vào các nền tảng khác.

Bảng 4.4.9 Bảng mô tả webhook của n8n

Tên Webhook	URL Endpoint	Mô tả chức năng
`upload_document`	`/webhook/upload`	Nhận file hoặc văn bản để tiến hành ingestion và lập chỉ mục.
`chat`	`/webhook/chat`	Xử lý truy vấn người dùng, gọi API RAG để sinh câu trả lời.
`remove_doc`	`/webhook/remove_doc`	Xóa dữ liệu đã lập chỉ mục theo `doc_id`.
`remove_space`	`/webhook/remove_space`	Xóa toàn bộ dữ liệu trong một không gian theo `space_id`.

4.4.4. Triển khai hệ thống trên Cloud

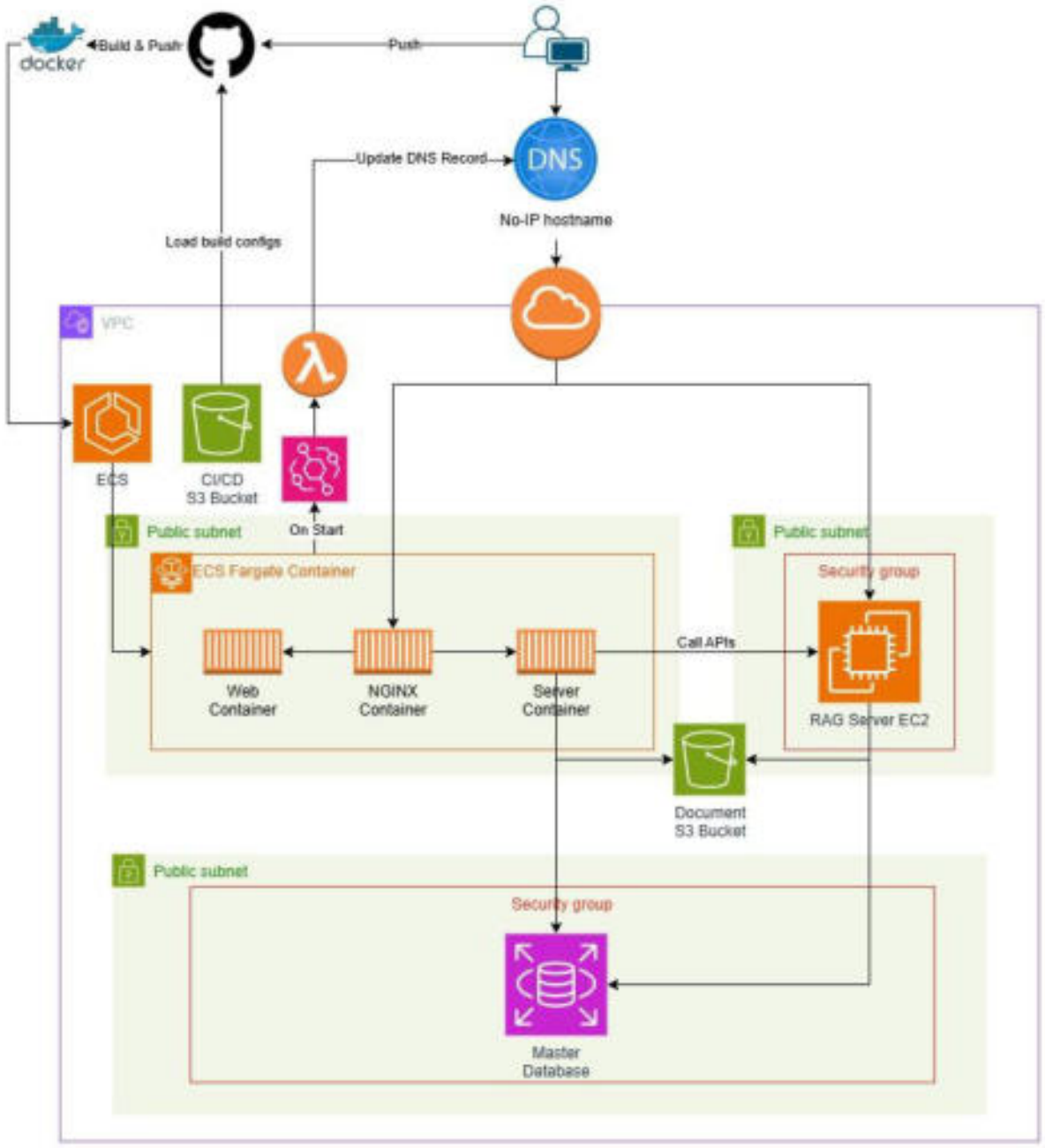
Khi triển khai hệ thống ở môi trường Staging và Production trên nền tảng đám mây, do mục tiêu tiết kiệm chi phí, hệ thống sẽ được thiết kế và triển khai theo phương án tiết kiệm nhất có thể, dù có thể chưa đạt được sự tối ưu về hiệu năng hoặc độ ổn định như môi trường thực tế. Cụ thể, đối với AWS, sẽ ưu tiên sử dụng các dịch vụ nằm trong gói Free Tier để tránh phát sinh chi phí. Đối với Google Cloud, sẽ tận dụng triệt để khoản tín dụng $300 miễn phí trong 3 tháng đầu tiên để triển khai và thử nghiệm hệ thống.

a) Triển khai ở AWS

Để triển khai lên nền tảng đám mây AWS, hệ thống cần được triển khai 3 containers: NGINX, Web và API Server, cùng với một RAG Server, ngoài ra cần có một SQL Database và File Storage, Redis sẽ được triển khai ở một dịch vụ bên thứ 3. Với yêu cầu này, hệ thống sẽ được triển khai lên nền tảng AWS sử dụng các dịch vụ sau:

- AWS ECS: Triển khai 3 Container của hệ thống
- AWS EC2: Chạy RAG Server
- AWS S3: Lưu trữ file

Ngoài ra, các dịch vụ như EventBridge, Lambda sẽ được sử dụng để triển khai CI/CD cho hệ thống trong quá trình phát triển.



Hình 4.4.2 Sơ đồ kiến trúc hệ thống trên AWS

b) Triển khai môi trường Staging

Môi trường Staging trên AWS được thiết kế để chạy tối đa 8 tiếng mỗi ngày nhằm tiết kiệm chi phí, ngoại trừ dịch vụ RDS được vận hành 24/7 để đảm bảo dữ liệu luôn sẵn sàng. Hệ thống sử dụng ba container ECS gồm NGINX, NextJS Web Client và Go Gin API Server, cùng với một máy chủ EC2 loại t3.small để chạy RAG Server. Tất cả các thành phần này đều đặt trên Public Subnet, điều này giúp tránh phát sinh chi phí cho NAT Gateway. Nhờ đó, tổng chi phí vận hành môi trường Staging được giữ ở mức khoảng 14,64 đô la mỗi tháng, phù hợp với nhu cầu phát triển và thử nghiệm.

Bảng 4.4.10 Ước tính chi phí triển khai môi trường Staging trên AWS

Thành phần	Số giờ	Đơn giá	Chi phí
EC2 t3.small	240h	~$0.023/h	~$5.52
ECS Task (Fargate)	240h	~$0.006/h	~$1.44
RDS db.t3.micro	240h	~$0.017/h	~$4.08
RDS EBS 20GB	-	Free Tier	$0
3 Elastic IPs	3 x 240h	$0.005/h	~$3.60
Tổng			**~$14.64**

c) *Triển khai môi trường Production*

Ở môi trường Production, chỉ với cấu hình tương tự như Staging nhưng hệ thống hoạt động liên tục 24/7, dẫn đến chi phí tăng lên đáng kể, khoảng 31,68 đô la mỗi tháng. Đây là mức chi phí khá cao và có thể vượt quá khả năng chi trả của sinh viên hoặc những người có ngân sách hạn chế. Do đó, việc duy trì môi trường Production trên AWS với cấu hình như vậy không phải là lựa chọn tối ưu về mặt chi phí.

Bảng 4.4.11 Ước tính chi phí triển khai môi trường Production trên AWS

Thành phần	Số giờ	Đơn giá	Chi phí
EC2 t3.small	720h	~$0.023/h	~$16.56
ECS Task (Fargate)	720h	~$0.006/h	~$4.32
RDS db.t3.micro	720h	~$0.017/h	Free (750h)
RDS EBS 20GB	-	Free Tier	$0
3 Elastic IPs	3 x 720h	$0.005/h	~$10.80
Tổng			**~$31.68**

Với mục tiêu tiết kiệm chi phí cho môi trường Production, đề xuất sử dụng khoản tín dụng 300 đô la miễn phí trong ba tháng đầu của Google Cloud. Điều này cho phép triển khai và vận hành hệ thống mà không phải lo lắng về chi phí phát sinh trong giai đoạn đầu. Việc chuyển sang GCP sẽ giúp cân bằng giữa hiệu năng và chi phí, đồng thời tận dụng tối đa ưu đãi của nhà cung cấp đám mây để hỗ trợ việc vận hành hệ thống hiệu quả và kinh tế hơn.

d) *Triển khai ở Google Cloud Platform*

Khác với khi triển khai trên AWS, GCP cung cấp khoản tín dụng 300 USD miễn phí trong 3 tháng, cho phép triển khai hệ thống production với cấu hình cao mà không phát sinh chi phí trong giai đoạn đầu, chỉ cần tổng chi phí sử dụng không vượt quá 100 USD/1 tháng.

Bảng 4.4.12 Ước tính chi phí triển khai môi trường Production trên GCP

Thành phần	Loại tài nguyên	Chi phí/tháng (USD)
VM RAG server	e2-standard-2 (2vCPU, 8GB)	$60.35
VM Web/API/Proxy	e2-medium (2vCPU, 4GB)	$30.17
Static IP (x2)	Gắn VM, luôn hoạt động	$0
Ổ đĩa (50GB x2)	pd-standard	~$4.00
Cloud Storage (logs)	~5GB	~$0.20

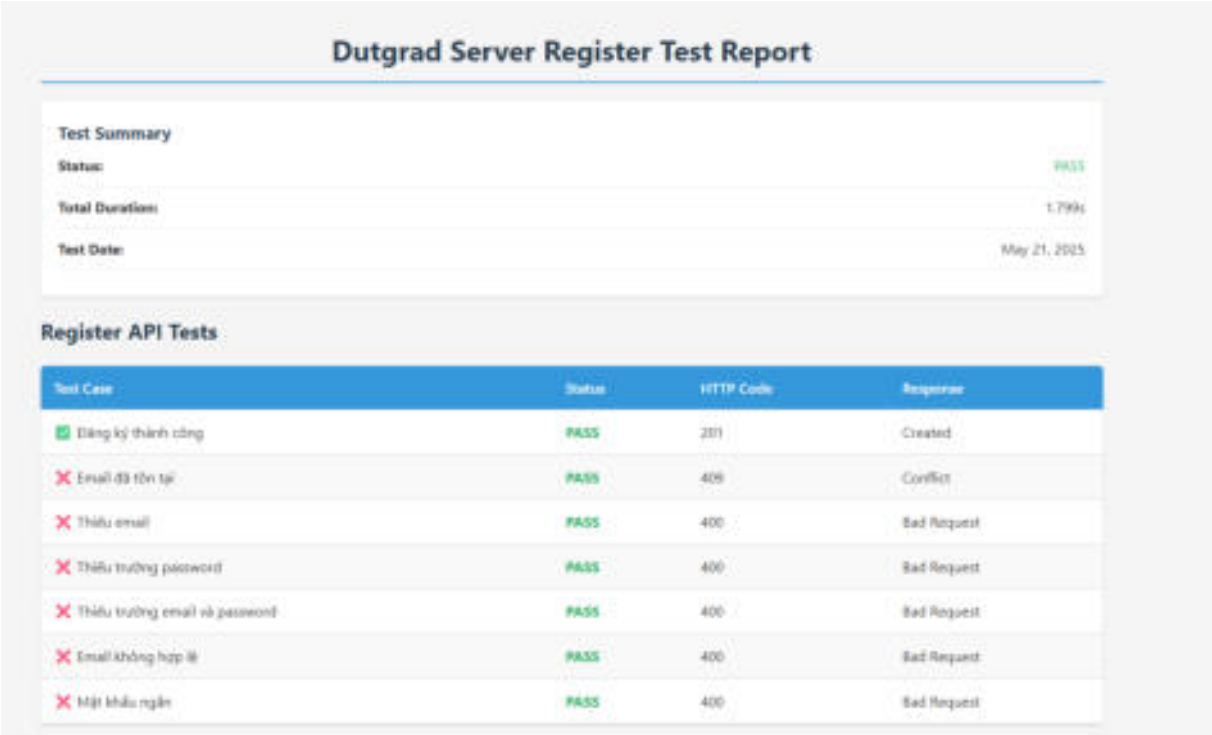
Networking (GCP → AWS)	<5GB egress (free tier)	$0
Tổng cộng		**~$94.72**
Discount		**-$100**
Chi phí cuối cùng		**$0**

Về Database, hệ thống sẽ sử dụng RDS Free Tier của AWS và chỉ tốn chi phí Public IP (~$3.60/tháng).

Có thể thấy, với chỉ khoảng 94.72 USD/tháng ở GCP và 3.60 USD/tháng ở RDS Free Tier (Public IP), hệ thống hoàn toàn đáp ứng yêu cầu triển khai Production cấu hình cao, đảm bảo hiệu năng và tính sẵn sàng, đồng thời miễn phí hoàn toàn trong 3 tháng đầu nhờ vào tín dụng khởi tạo của GCP. Do đó tổng cộng chi phí để triển khai môi trường Production là $3.60/tháng. Đây là lựa chọn tối ưu để vừa tiết kiệm chi phí, vừa bảo đảm vận hành ổn định ngay từ giai đoạn đầu và đủ để đáp ứng học kì đồ án.

Có thể tiết kiệm chi phí hơn bằng việc chạy Cloud Run nhưng Cloud Run không phù hợp trong kiến trúc này vì RAG server cần chạy các tiến trình nền liên tục, cụ thể là n8n – một công cụ workflow tự động hoá cần duy trì trạng thái và xử lý các trigger/background jobs không theo mô hình request-response. Để đảm bảo n8n và RAG server hoạt động ổn định 24/7, cần triển khai trên VM truyền thống (Compute Engine) – nơi bạn có thể kiểm soát hoàn toàn môi trường, tài nguyên, và thời gian uptime của server. Đây là lý do Cloud Run không thể sử dụng trong kiến trúc production này.

4.5. Kết quả kiểm thử



Hình 4.5.1 Kiểm thử API đăng ký

Dutgrad Server Test Login Report

Test Summary

Status:	PASS
Total Duration:	1.920s
Test Date:	May 21, 2025

Login API Tests

Test Case	Status	HTTP Code	Response
Đăng nhập thành công	PASS	200	Login successful
Đăng nhập Google thành công	PASS	200	Login successful
Đăng nhập yêu cầu MFA	PASS	200	MFA verification required
Email không tồn tại	PASS	401	Authentication failed
Sai mật khẩu	PASS	401	Authentication failed
Thiếu email	PASS	400	Missing email
Thiếu mật khẩu	PASS	400	Missing password
Thiếu auth_type khi đăng nhập Google	PASS	400	Missing password
Body rỗng	PASS	400	Missing email

Hình 4.5.2 Kiểm thử API đăng nhập

Dutgrad Server OAuth Test Report

Test Summary

Status:	PASS
Total Duration:	2.023s
Test Date:	May 21, 2025

Google OAuth Authentication

Test Case	Status	HTTP Code	Response
Google OAuth Redirect	PASS	307	Temporary Redirect

Exchange State (OAuth Callback)

Test Case	Status	HTTP Code	Response
Exchange thành công	PASS	200	OK
State không hợp lệ	PASS	400	Bad Request
Code không hợp lệ	PASS	400	Bad Request

External Authentication

Test Case	Status	HTTP Code	Response
External auth thành công	PASS	200	OK
Token không hợp lệ	PASS	401	Unauthorized
Auth type không được hỗ trợ	PASS	400	Bad Request

Hình 4.5.3 Kiểm thử API OAuth

Dutgrad Server MFA Test Report

Test Summary

Status: PASS

Total Duration: 1.706s

Test Date: May 21, 2025

MFA Status

Test Case	Status	HTTP Code	Response
Kiểm tra trạng thái MFA	PASS	200	OK

MFA Setup

Test Case	Status	HTTP Code	Response
Thiết lập MFA	PASS	200	OK

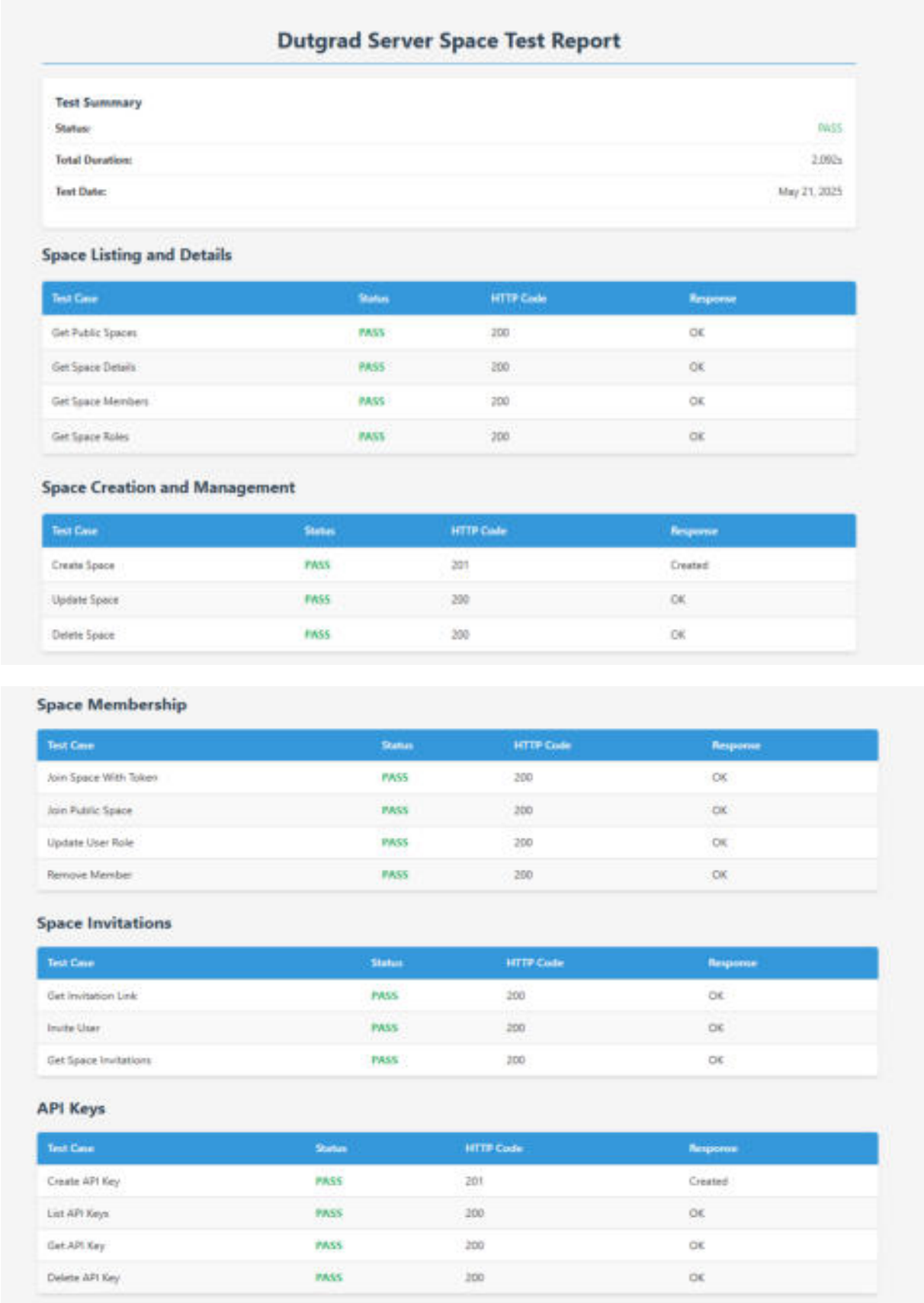
MFA Verification

Test Case	Status	HTTP Code	Response
Verify MFA thành công	PASS	200	OK
Code không hợp lệ	PASS	400	Bad Request

MFA Disable

Test Case	Status	HTTP Code	Response
Vô hiệu hóa MFA	PASS	200	OK

Hình 4.5.4 Kiểm thử API MFA



Dutgrad Server Space Test Report

Test Summary

Status:	PASS
Total Duration:	2.092s
Test Date:	May 21, 2025

Space Listing and Details

Test Case	Status	HTTP Code	Response
Get Public Spaces	PASS	200	OK
Get Space Details	PASS	200	OK
Get Space Members	PASS	200	OK
Get Space Roles	PASS	200	OK

Space Creation and Management

Test Case	Status	HTTP Code	Response
Create Space	PASS	201	Created
Update Space	PASS	200	OK
Delete Space	PASS	200	OK

Space Membership

Test Case	Status	HTTP Code	Response
Join Space With Token	PASS	200	OK
Join Public Space	PASS	200	OK
Update User Role	PASS	200	OK
Remove Member	PASS	200	OK

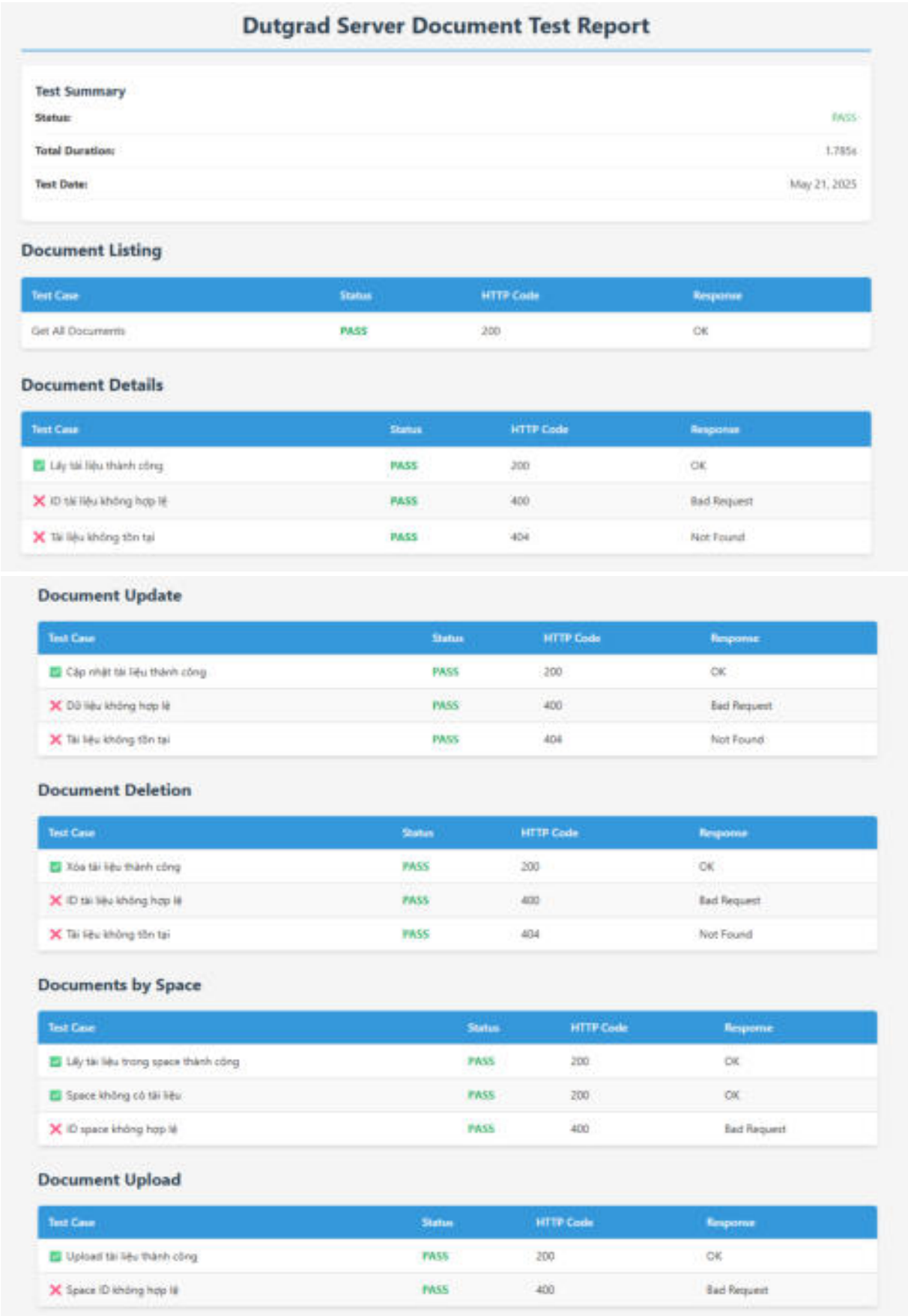
Space Invitations

Test Case	Status	HTTP Code	Response
Get Invitation Link	PASS	200	OK
Invite User	PASS	200	OK
Get Space Invitations	PASS	200	OK

API Keys

Test Case	Status	HTTP Code	Response
Create API Key	PASS	201	Created
List API Keys	PASS	200	OK
Get API Key	PASS	200	OK
Delete API Key	PASS	200	OK

Hình 4.5.5 Kiểm thử API quản lý space



Dutgrad Server Document Test Report

Test Summary

Status:	PASS
Total Duration:	1.785s
Test Date:	May 21, 2025

Document Listing

Test Case	Status	HTTP Code	Response
Get All Documents	PASS	200	OK

Document Details

Test Case	Status	HTTP Code	Response
✅ Lấy tài liệu thành công	PASS	200	OK
❌ ID tài liệu không hợp lệ	PASS	400	Bad Request
❌ Tài liệu không tồn tại	PASS	404	Not Found

Document Update

Test Case	Status	HTTP Code	Response
✅ Cập nhật tài liệu thành công	PASS	200	OK
❌ Dữ liệu không hợp lệ	PASS	400	Bad Request
❌ Tài liệu không tồn tại	PASS	404	Not Found

Document Deletion

Test Case	Status	HTTP Code	Response
✅ Xóa tài liệu thành công	PASS	200	OK
❌ ID tài liệu không hợp lệ	PASS	400	Bad Request
❌ Tài liệu không tồn tại	PASS	404	Not Found

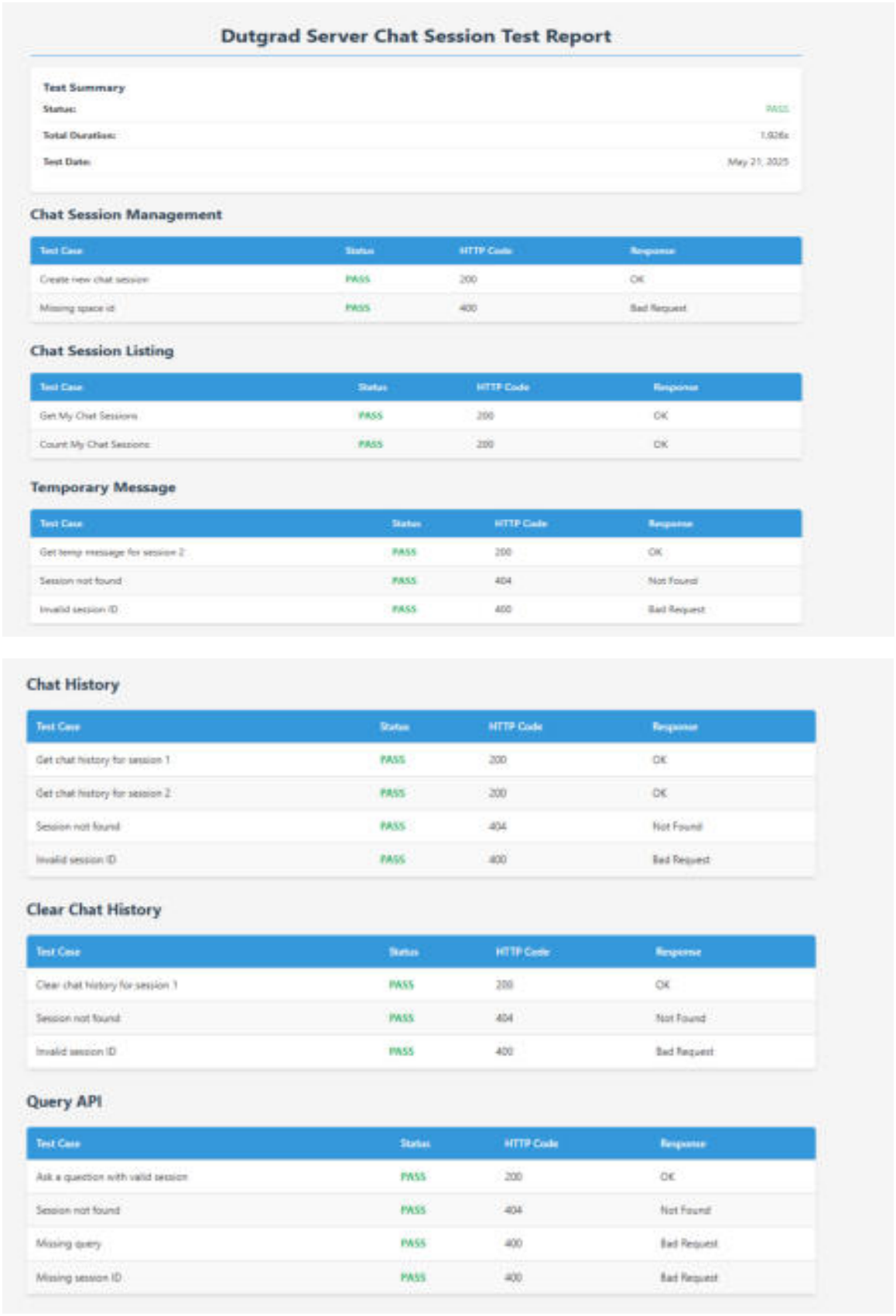
Documents by Space

Test Case	Status	HTTP Code	Response
✅ Lấy tài liệu trong space thành công	PASS	200	OK
✅ Space không có tài liệu	PASS	200	OK
❌ ID space không hợp lệ	PASS	400	Bad Request

Document Upload

Test Case	Status	HTTP Code	Response
✅ Upload tài liệu thành công	PASS	200	OK
❌ Space ID không hợp lệ	PASS	400	Bad Request

Hình 4.5.6 Kiểm thử API quản lý document



Dutgrad Server Chat Session Test Report

Test Summary

Status: PASS

Total Duration: 1.926s

Test Date: May 21, 2025

Chat Session Management

Test Case	Status	HTTP Code	Response
Create new chat session	PASS	200	OK
Missing space id	PASS	400	Bad Request

Chat Session Listing

Test Case	Status	HTTP Code	Response
Get My Chat Sessions	PASS	200	OK
Count My Chat Sessions	PASS	200	OK

Temporary Message

Test Case	Status	HTTP Code	Response
Get temp message for session 2	PASS	200	OK
Session not found	PASS	404	Not Found
Invalid session ID	PASS	400	Bad Request

Chat History

Test Case	Status	HTTP Code	Response
Get chat history for session 1	PASS	200	OK
Get chat history for session 2	PASS	200	OK
Session not found	PASS	404	Not Found
Invalid session ID	PASS	400	Bad Request

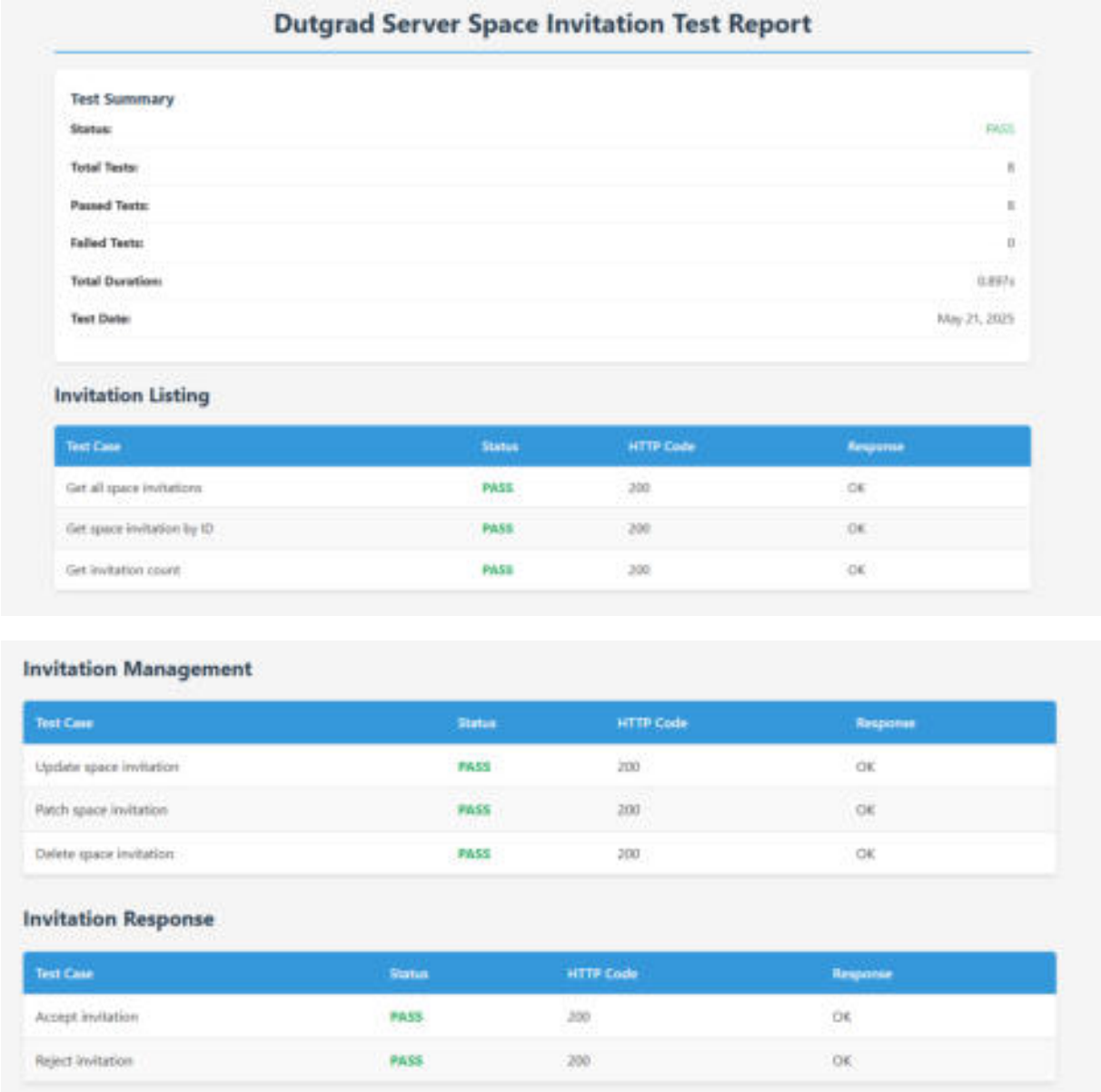
Clear Chat History

Test Case	Status	HTTP Code	Response
Clear chat history for session 1	PASS	200	OK
Session not found	PASS	404	Not Found
Invalid session ID	PASS	400	Bad Request

Query API

Test Case	Status	HTTP Code	Response
Ask a question with valid session	PASS	200	OK
Session not found	PASS	404	Not Found
Missing query	PASS	400	Bad Request
Missing session ID	PASS	400	Bad Request

Hình 4.5.7 Kiểm thử API quản lý chat

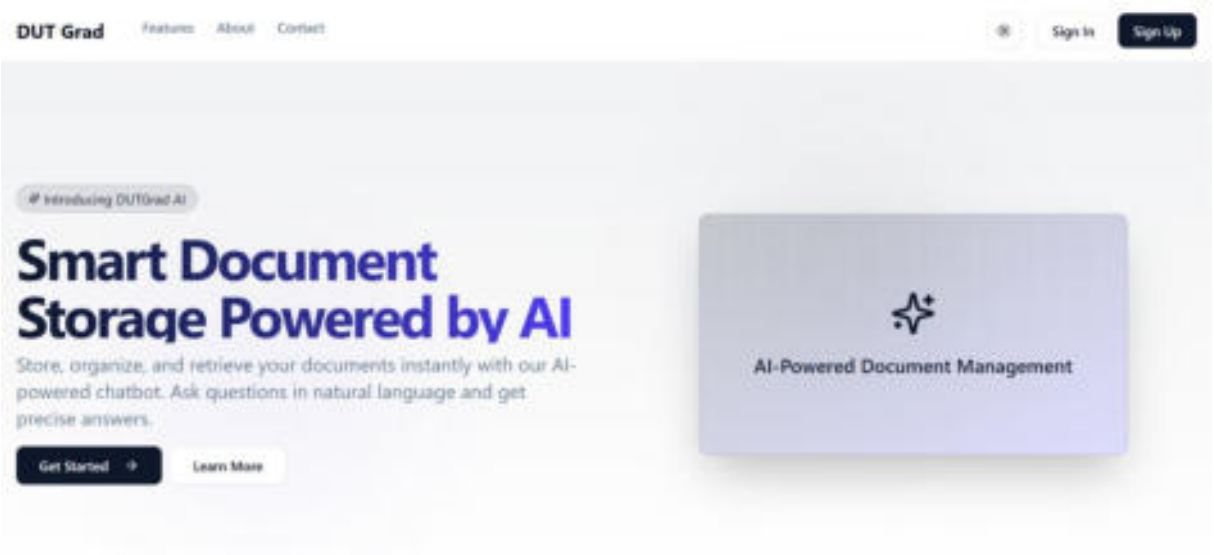


Hình 4.5.8 Kiểm thử API lời mời Space

4.6. Giao diện chương trình

4.6.1. Màn hình trang chủ

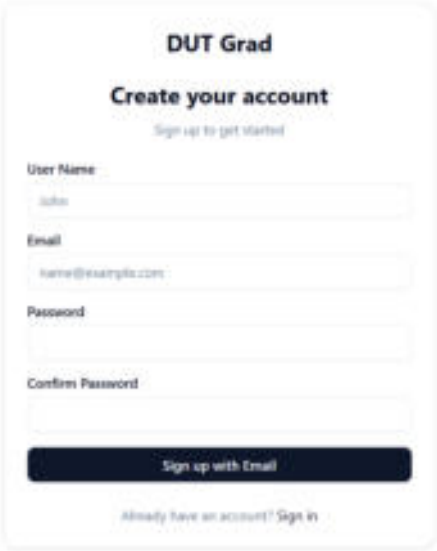
Màn hình trang chủ là giao diện đầu tiên mà người dùng nhìn thấy khi truy cập vào hệ thống. Đây là một landing page có thiết kế hiện đại, chuyên nghiệp, cung cấp cái nhìn tổng quan về hệ thống.



Hình 4.6.1 Màn hình trang chủ

4.6.2. Màn hình đăng ký

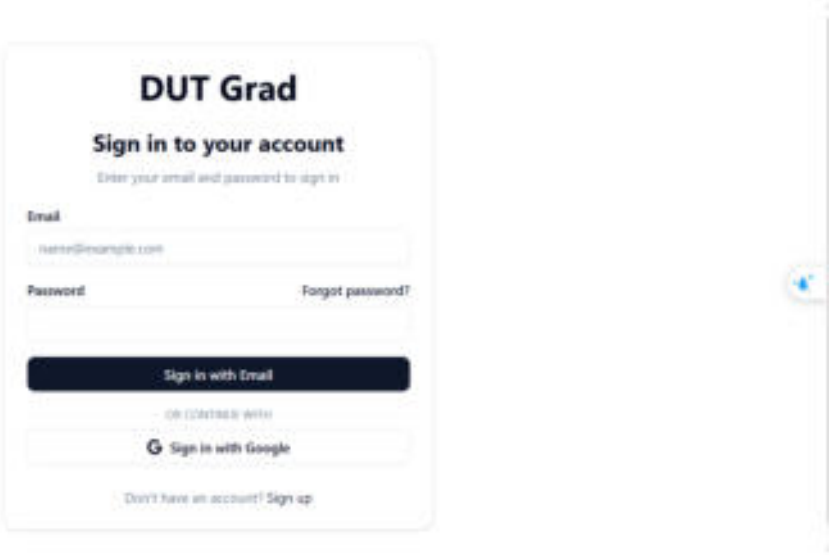
Người dùng cần tạo tài khoản bằng cách cung cấp username, email và mật khẩu để có thể sử dụng các chức năng chính của hệ thống.



Hình 4.6.2 Màn hình đăng ký

4.6.3. Màn hình đăng nhập

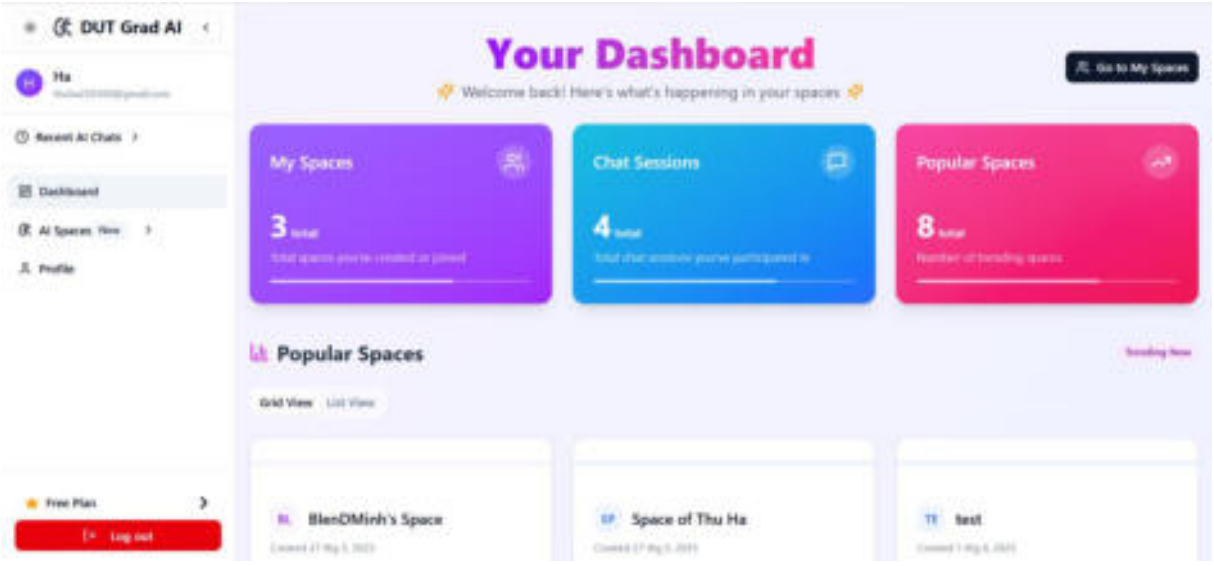
Để sử dụng hầu hết các chức năng chính trong hệ thống, người dùng cần phải đăng nhập bằng email và mật khẩu hoặc đăng nhập bằng google.



Hình 4.6.3 Màn hình đăng nhập

4.6.4. Màn hình dashboard

Màn hình Dashboard cung cấp tổng quan về hoạt động của người dùng sau khi đăng nhập. Giao diện hiển thị các thông tin như số lượng không gian đã tạo hoặc tham gia (My Spaces), số phiên trò chuyện (Chat Sessions),…Thanh điều hướng bên trái giúp truy cập nhanh các mục như Recent Chats, Spaces, Profile, và Log out. Giao diện hiện đại, trực quan, dễ sử dụng.

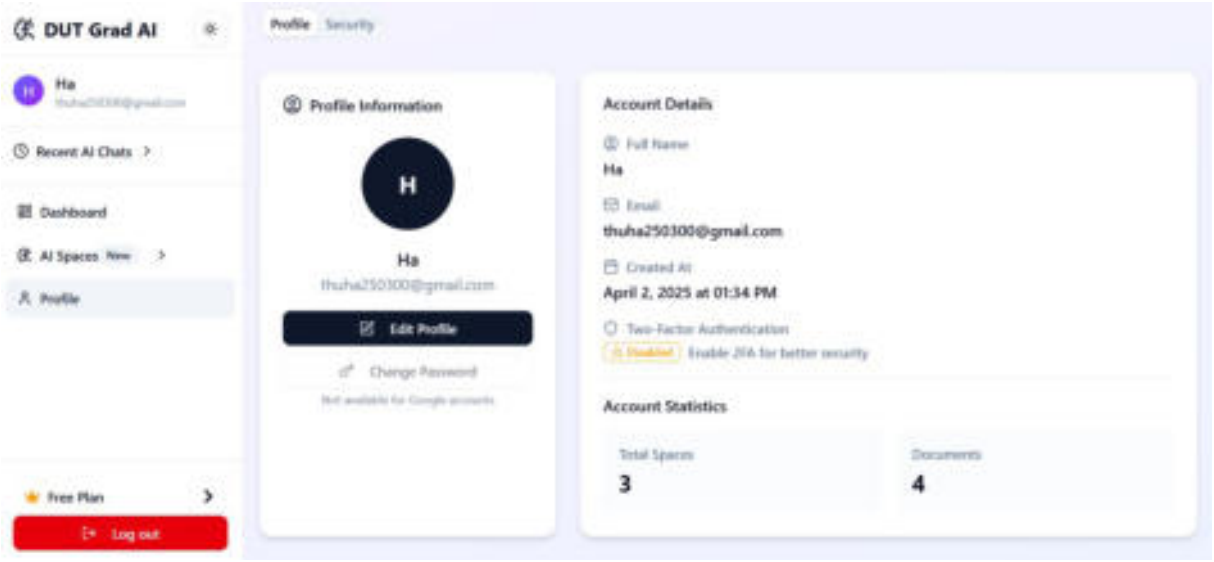


Hình 4.6.4 Màn hình dashboard

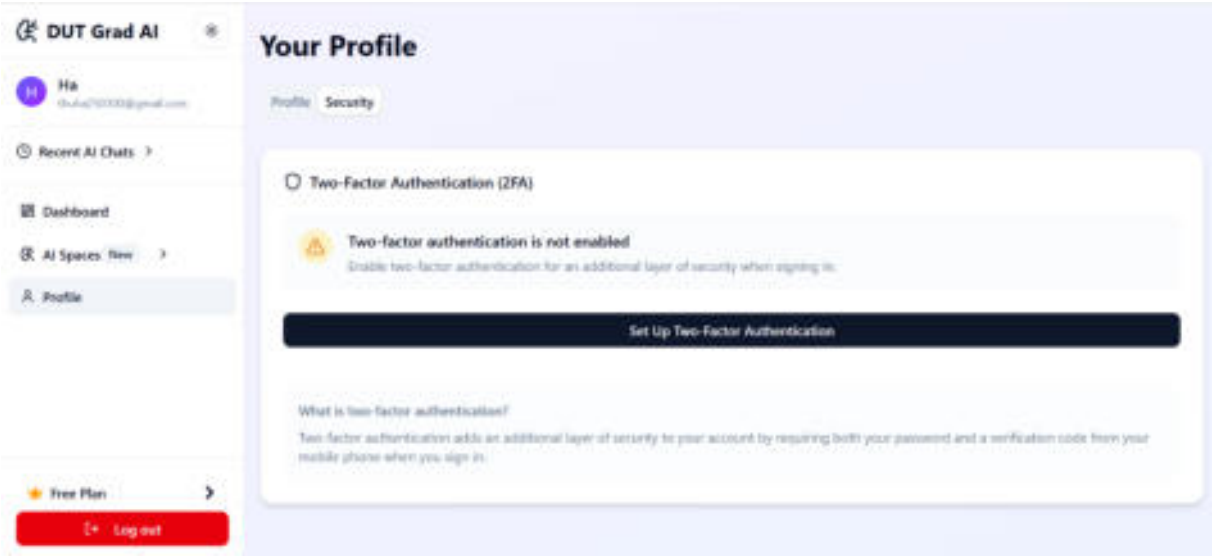
4.6.5. Màn hình profile

Màn hình Profile hiển thị thông tin cá nhân và trạng thái bảo mật tài khoản của người dùng. Giao diện được chia thành hai tab chính: Profile và Security.

- Tab Profile: Hiển thị họ tên, email, thời gian tạo tài khoản, trạng thái xác thực hai yếu tố (2FA), số lượng AI Spaces và tài liệu. Cho phép chỉnh sửa thông tin cá nhân.
- Tab Security: Cho phép thiết lập xác thực hai yếu tố (2FA) bằng cách quét mã QR bằng ứng dụng xác thực (Authenticator). Sau đó, người dùng nhập mã xác minh 6 chữ số để kích hoạt MFA. Hệ thống cũng cung cấp các mã dự phòng dùng trong trường hợp không thể truy cập ứng dụng.



Hình 4.6.5 Màn hình thông tin cá nhân

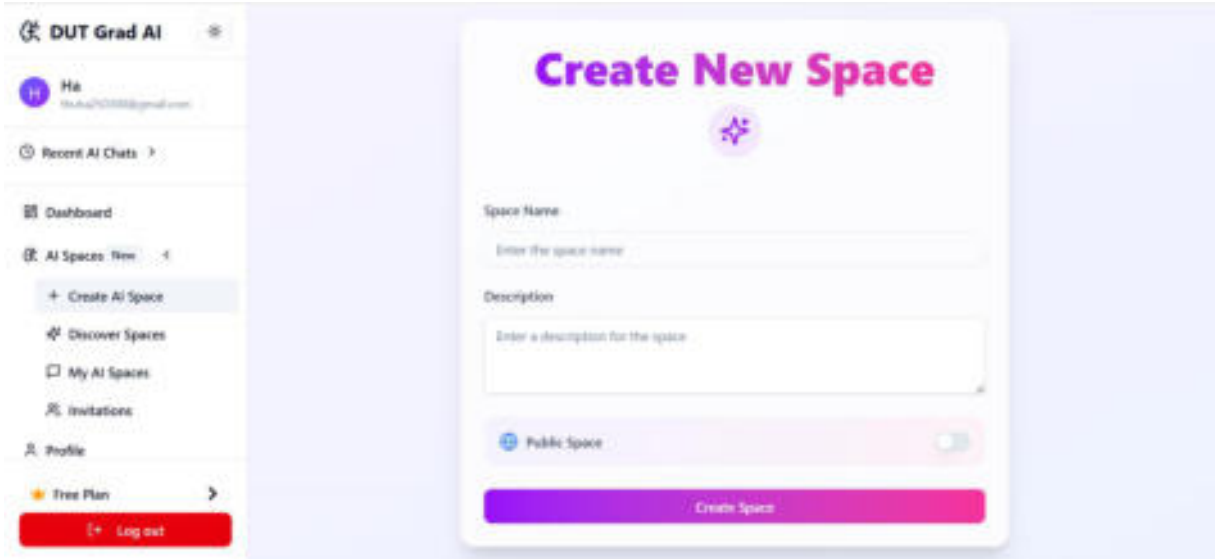


Hình 4.6.6 Màn hình bảo mật

Hình 4.6.7 Màn hình thiết lập bảo mật

4.6.6. Màn hình tạo mới Space

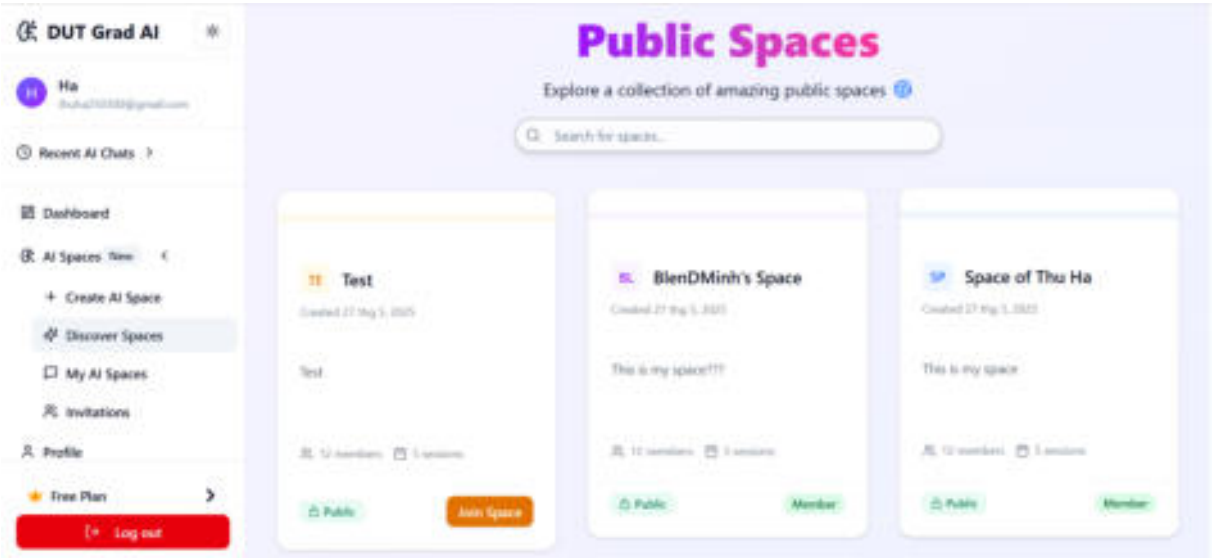
Giao diện cho phép người dùng tạo một space mới bằng cách nhập tên, mô tả và lựa chọn chế độ công khai hoặc riêng tư. Nút "Create Space" dùng để xác nhận và khởi tạo không gian mới.



Hình 4.6.8 Màn hình tạo mới space

4.6.7. Màn hình space công khai

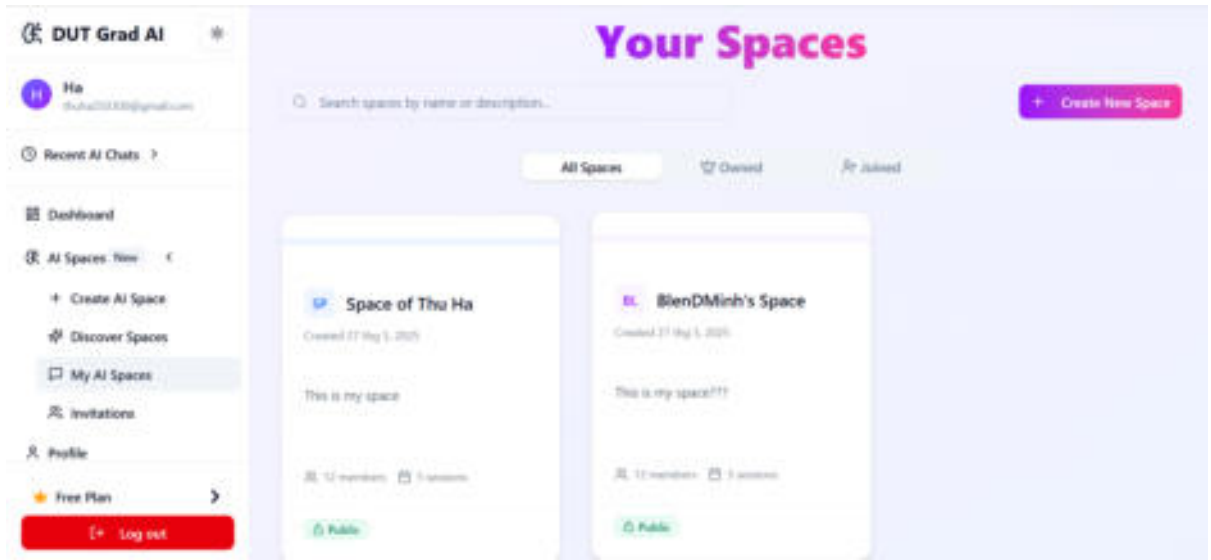
Hiển thị danh sách các Space công khai mà người dùng có thể tham gia. Mỗi Space hiển thị tên, mô tả, ngày tạo và trạng thái truy cập (Public/Member). Người dùng có thể tìm kiếm và tham gia các Space từ màn hình này.



Hình 4.6.9 Màn hình space công khai

4.6.8. Màn hình spaces của người dùng:

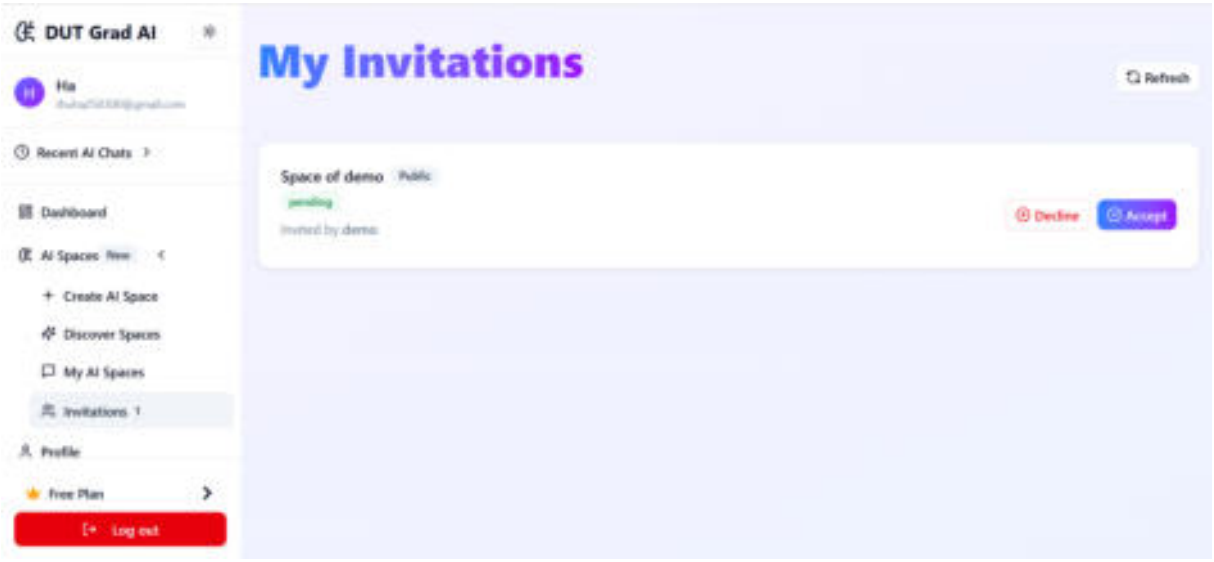
Hiển thị danh sách các Space mà người dùng sở hữu hoặc đã tham gia. Cho phép lọc theo tất cả, Space sở hữu hoặc đã tham gia. Người dùng có thể tạo Space mới hoặc tìm kiếm theo tên/mô tả để dễ dàng quản lý.



Hình 4.6.10 Màn hình space của người dùng

4.6.9. Màn hình quản lý lời mời

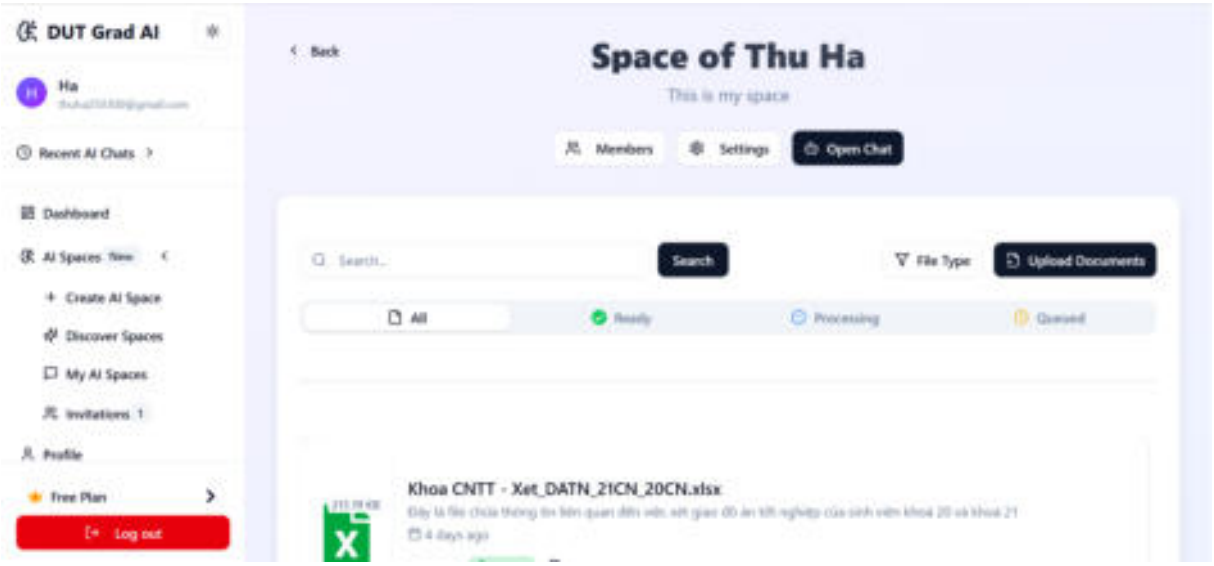
Hiển thị danh sách các lời mời tham gia vào Space từ người dùng khác. Người dùng có thể chấp nhận (Accept) hoặc từ chối (Decline) lời mời. Mỗi lời mời kèm theo trạng thái (pending), trạng thái Space (Public/Private), và thông tin người mời.



Hình 4.6.11 Màn hình quản lý lời mời

4.6.10. Màn hình chi tiết space

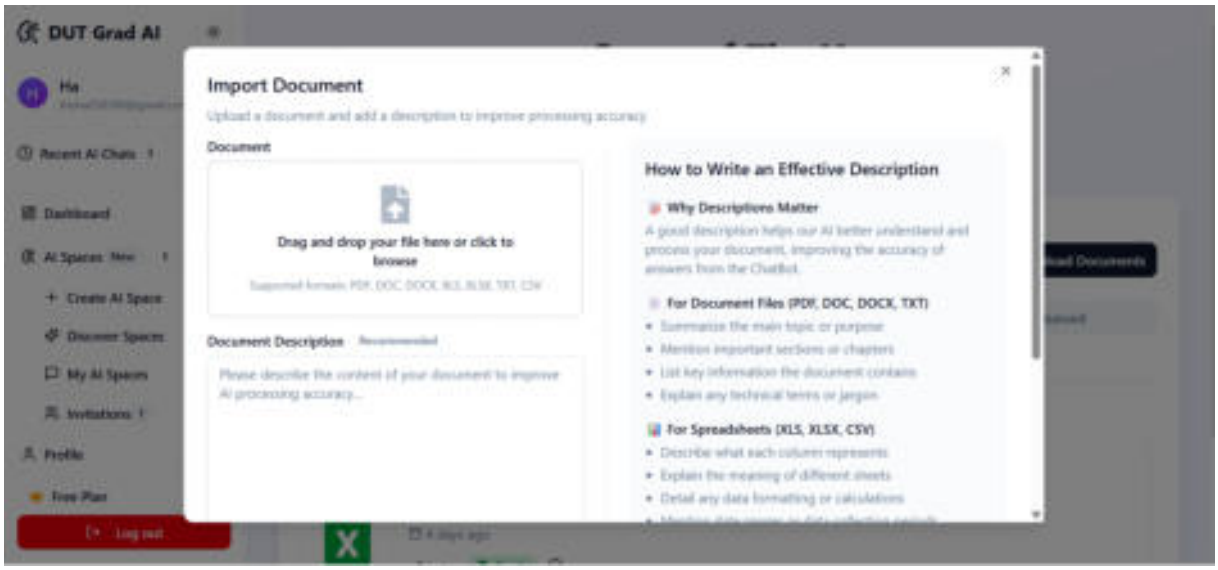
Màn hình chi tiết Space là giao diện trung tâm cho phép người dùng tương tác và quản lý các tài liệu trong một space (không gian) cụ thể. Đây là nơi người dùng có thể tổ chức, tải lên, tìm kiếm và trò chuyện với Chatbot dựa trên nội dung tài liệu đã cung cấp.



Hình 4.6.12 Màn hình chi tiết space

4.6.11. Màn hình nhập tài liệu

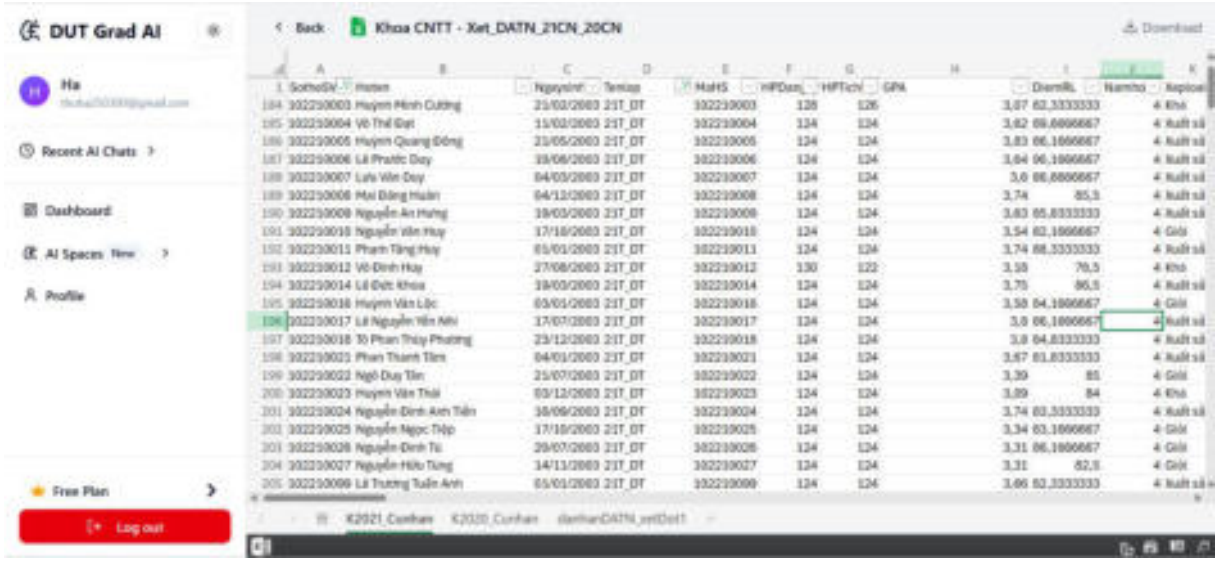
Màn hình nhập tài liệu cho phép người dùng tải lên các tệp văn bản (.xlsx,.docx, .csv, .pdf, .txt) để hệ thống AI có thể phân tích và hiểu nội dung nhằm phục vụ cho việc tương tác và hỏi đáp trong không gian (Space).



Hình 4.6.13 Màn hình nhập tài liệu

4.6.12. Màn hình xem chi tiết document

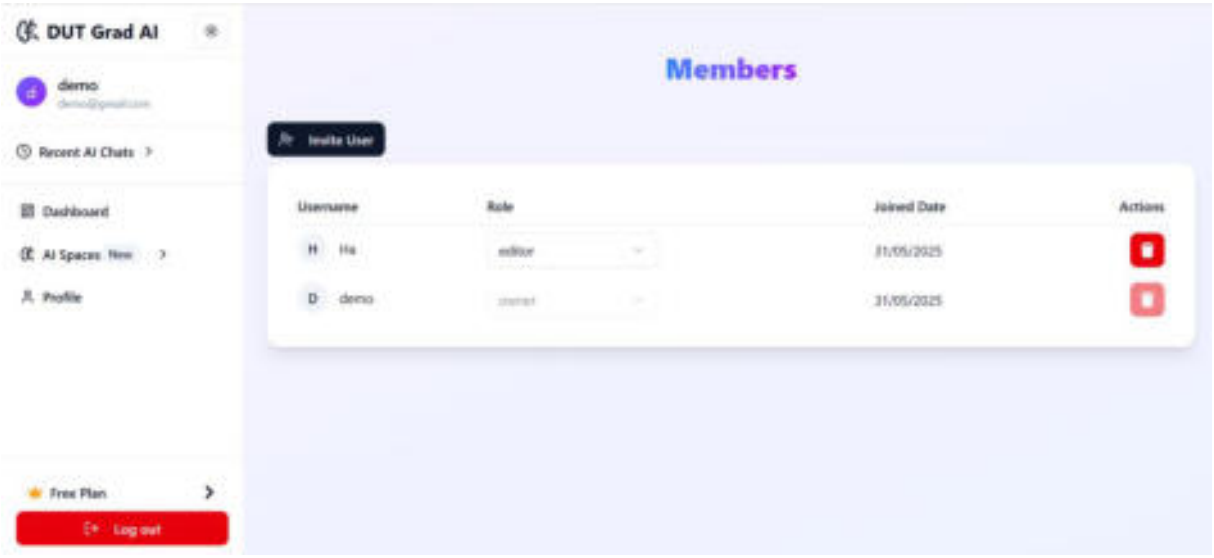
Màn hình này cho phép người dùng xem chi tiết thông tin tài liệu đã được đã lên.



Hình 4.6.14 Màn hình xem chi tiết tài liệu

4.6.13. Màn hình quản lý thành viên của space

Màn hình này cho phép quản lý thành viên trong không gian (Space), bao gồm hiển thị danh sách người dùng, vai trò (owner, editor, viewer), ngày tham gia và chức năng mời hoặc xóa thành viên. Tính năng này giúp phân quyền và kiểm soát truy cập hiệu quả trong quá trình cộng tác.



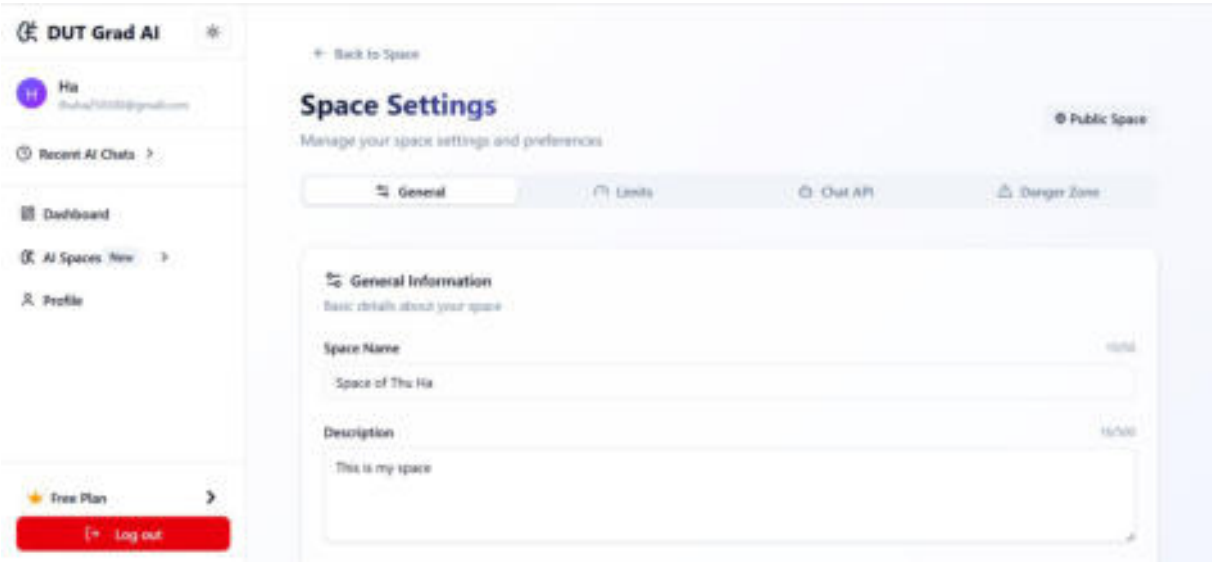
Hình 4.6.15 Màn hình quản lý thành viên của space

4.6.14. Màn hình cài đặt space

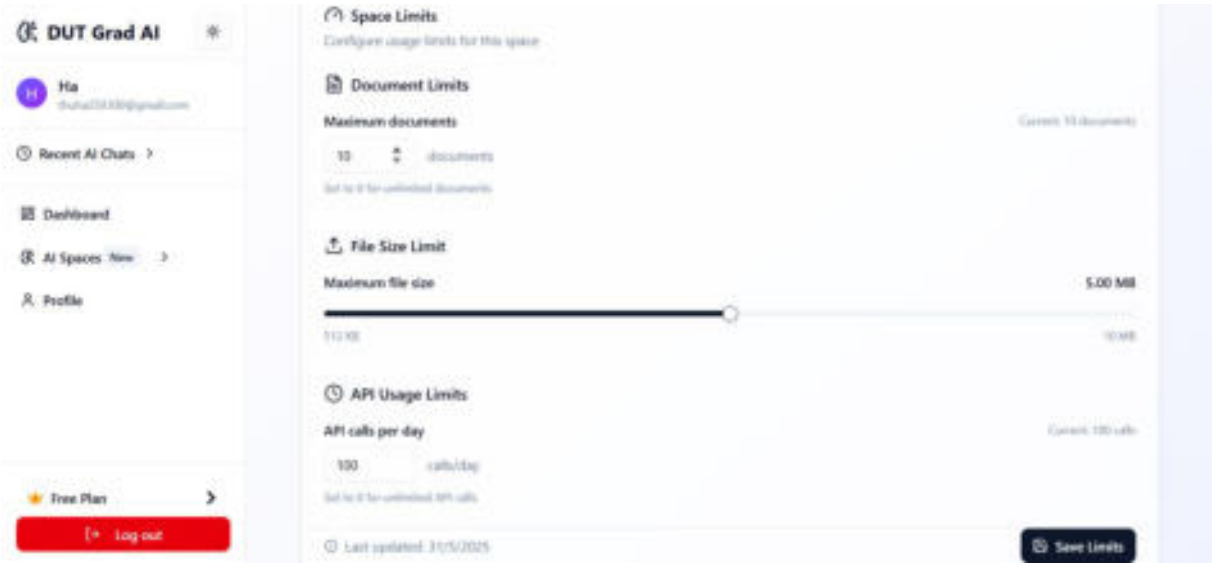
Màn hình này gồm 4 tab chức năng giúp người dùng quản lý toàn diện không gian AI:

- General: Chỉnh sửa thông tin space.
- Limits: Thiết lập giới hạn liên quan đến space.
- Chat API: Cung cấp thông tin tích hợp API trò chuyện.
- Danger Zone: Cho phép xóa Space.

Giao diện được thiết kế trực quan, hỗ trợ cá nhân hóa và kiểm soát quyền truy cập hiệu quả.



Hình 4.6.16 Màn hình chỉnh sửa thông tin space

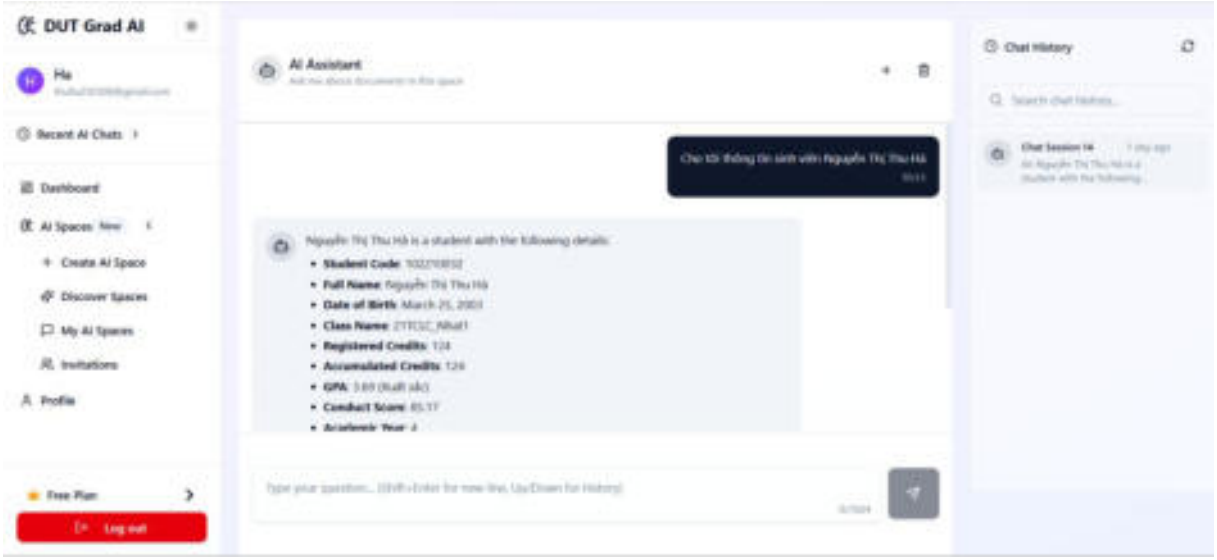


Hình 4.6.17 Màn hình Thiết lập giới hạn space

Hình 4.6.18 Màn hình thông tin tích hợp API trò chuyện

4.6.15. Màn hình Chatbot

Màn hình ChatBot cho phép người dùng tương tác trực tiếp với trợ lý ảo (AI Agent). Tại đây, người dùng có thể nhập các câu hỏi tự nhiên và nhận phản hồi nhanh chóng dựa trên tài liệu đã được cung cấp trong không gian làm việc (space).



Hình 4.6.19 Màn hình chatbot

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết quả đạt được

Trong quá trình nghiên cứu, tìm hiểu lý thuyết và thực hiện dự án, dự án đã thành công trong việc đạt được mục tiêu ban đầu.

- **Về mặt lý thuyết**:
  - Trau dồi được kỹ năng phân tích, thiết kế và xây dựng cơ sở dữ liệu, bao gồm việc xác định các yêu cầu, thiết kế cấu trúc dữ liệu, và triển khai cơ sở dữ liệu cho hệ thống.
  - Hiểu và vận dụng được ngôn ngữ lập trình Go, Python, Typescript, các framework như Gin, Fast API, NextJs để xây dựng được hệ thống.
  - Hiểu và ứng dụng kiến thức về AI Agent và Retrieval-Augmented Generation (RAG) để xây dựng chức năng truy vấn tài liệu bằng chatbot
- **Về mặt thực tiễn ứng dụng:**
  - Xây dựng thành công hệ thống với các chức năng cơ bản như: tạo và quản lý space, nhập và truy xuất tài liệu, tìm kiếm thông minh, chatbot hỗ trợ truy vấn tài liệu
  - Hệ thống hỗ trợ người dùng quản lý tài liệu một cách trực quan, dễ sử dụng, tìm kiếm thông tin nhanh chóng thông qua truy vấn chatbot có tích hợp AI có thể truy cập mọi lúc, mọi nơi qua trình duyệt web.
  - Giao diện người dùng đơn giản, dễ sử dụng và hỗ trợ quản lý tài liệu theo từng space.

Những khó khăn và cách khắc phục

Trong quá trình phát triển hệ thống, nhóm thực hiện đã gặp phải một số khó khăn như sau:

- Tích hợp các công nghệ mới: Việc kết hợp các công nghệ mới như RAG, AI Agent hay framework mới đòi hỏi thử nghiệm để tương thích giữa các phần trong hệ thống.
  Cách khắc phục:Tham khảo tài liệu, tiến hành thử nghiệm để phù hợp với hệ thống
- Giới hạn thời gian: Với thời gian hạn chế, một số chức năng nâng cao chưa triển khai kịp như quản trị viên hay cộng tác nhóm
  Cách khắc phục: Ưu tiên phát triển các chức năng cốt lõi nhằm đảm bảo hệ thống có thể vận hành ổn định. Các chức năng chưa hoàn thiện được ghi nhận để tiếp tục phát triển trong giai đoạn sau.

- Kinh nghiệm triển khai hệ thống thực tế: Khi triển khai hệ thống thực tế, đặc biệt là trên môi trường đám mây, gặp khó khăn trong việc cấu hình, bảo mật và tối ưu hóa hiệu suất.
  Cách khắc phục: Nghiên cứu và áp dụng các công cụ hỗ trợ triển khai như Docker, AWS
- Thiếu kinh phí: Hạn chế về ngân sách khiến hệ thống chỉ có thể duy trì hạ tầng Cloud và sử dụng mô hình LLM ở mức tối thiểu. Điều này dẫn đến nhiều bất cập như: kiến trúc Cloud không thể tối ưu về hiệu năng và độ mở rộng, không đủ tài nguyên để tổ chức kiểm thử trên các bộ dữ liệu lớn – vốn là điều kiện cần thiết để đánh giá chất lượng hệ thống RAG một cách toàn diện.
  Cách khắc phục: Lựa chọn giải pháp tối ưu chi phí như sử dụng phiên bản miễn phí hoặc có chi phí thích hợp.

Những vấn đề còn tồn tại

- Quản trị viên: Do hạn chế về thời gian, chức năng quản trị viên của hệ thống chưa được triển khai.
- Tối ưu hiệu suất: Khi số lượng người dùng và tài liệu tăng cao, hiệu suất truy xuất và phản hồi của chatbot có thể bị chậm.
- Tính năng cộng tác: Hệ thống chưa hỗ trợ mạnh mẽ cho việc trao đổi, thảo luận nhóm hay cộng tác trực tuyến giữa các thành viên trong cùng một không gian.

Hướng phát triển

Mặc dù hệ thống hiện tại đã đáp ứng được những chức năng cơ bản đề ra, tuy nhiên vẫn còn nhiều tiềm năng để mở rộng và nâng cao chất lượng trong tương lai. Một số hướng phát triển có thể triển khai bao gồm:

- Tối ưu truy vấn thông minh bằng AI: Cải thiện độ chính xác và tốc độ phản hồi của chatbot bằng cách tối ưu mô hình RAG để nâng cao trải nghiệm truy vấn tài liệu.
- Tăng cường tính năng cộng tác: Phát triển các chức năng tương tác nhóm như trò chuyện trực tiếp trong không gian (chat nội bộ).
- Tích hợp API với các nền tảng khác: Cho phép kết nối hệ thống với các công cụ quản lý khác (Google Drive, OneDrive), hoặc hệ thống quản lý nội bộ của doanh nghiệp.

TÀI LIỆU THAM KHẢO

[1] 200lab, "RAG (Retrieval-Augmented Generation) là gì? Giải thích dễ hiểu cho Developer," 200lab, Mar. 11, 2025. [Online].

Available: https://200lab.io/blog/rag-la-gi. [Accessed: 29 May, 2025].

[2] Viblo, "GraphRAG: Một sự nâng cấp mới của RAG truyền thống," Viblo, May 21, 2024. [Online].

Available: https://viblo.asia/p/graphrag-mot-su-nang-cap-moi-cua-rag-truyen-thong-chang-EoW4oXRBJml. [Accessed: 29 May, 2025].

[3] OpenAI, "LightRAG Overview," OpenAI. [Online].

Available: https://platform.openai.com/docs/guides/retrieval/light-rag. [Accessed: 29 May, 2025].

[4] CodeLearn, "Mô hình client-server," CodeLearn. [Online].

Available: https://codelearn.io/sharing/tim-hieu-ve-mo-hinh-client-server. [Accessed: 29 May, 2025].

[5] TopDev, "Git là gì?," TopDev. [Online].

Available: https://topdev.vn/blog/git-la-gi. [Accessed: 29 May, 2025].

[6] TopDev, "Github là gì?," TopDev. [Online].

Available: https://topdev.vn/blog/github-la-gi. [Accessed: 29 May, 2025].

[7] TopOnSeek, "Github Actions là gì?," TopOnSeek. [Online].

Available: https://www.toponseek.com/blogs/github-actions-la-gi. [Accessed: 29 May, 2025].

[8] DevDocs, "Next.js," DevDocs. [Online].

Available: https://devdocs.io/nextjs. [Accessed: 29 May, 2025].

[9] TypeScript Team, "TypeScript Documentation," TypeScript. [Online].

Available: https://www.typescriptlang.org/docs. [Accessed: 29 May, 2025].

[10] Tailwind Labs, "Tailwind CSS," Tailwind CSS. [Online].

Available: https://tailwindcss.com. [Accessed: 29 May, 2025].

[11] W3Schools, "Go Language Tutorial," W3Schools. [Online].

Available: https://www.w3schools.com/go. [Accessed: 29 May, 2025].

[12] DevDocs, "FastAPI," DevDocs. [Online].

Available: https://devdocs.io/fastapi. [Accessed: 29 May, 2025].

[13] n8n, "Intro to Advanced AI Workflows," n8n Docs. [Online].

Available: https://docs.n8n.io/advanced-ai/intro-tutorial. [Accessed: 29 May, 2025].

[14] PostgreSQL Global Development Group, "About PostgreSQL," PostgreSQL. [Online].

Available: https://www.postgresql.org/about. [Accessed: 29 May, 2025].

[15] TopDev, "Redis là gì?," TopDev. [Online].

Available: https://topdev.vn/blog/redis-la-gi. [Accessed: 29 May, 2025].

[16] Qdrant, "Qdrant – Vector Search Engine," Qdrant. [Online].

Available: https://qdrant.tech/documentation. [Accessed: 29 May, 2025].

[17] Docker Inc., "Docker Documentation," Docker. [Online].

Available: https://docs.docker.com. [Accessed: 29 May, 2025].

[18] Amazon Web Services, "What is AWS?," Amazon Web Services. [Online]. Available: https://aws.amazon.com/what-is-aws. [Accessed: 29 May, 2025].

[19] Amazon Web Services, "Amazon EC2," Amazon Web Services. [Online]. Available: https://aws.amazon.com/ec2. [Accessed: 29 May, 2025].

[20] Amazon Web Services, "Amazon S3," Amazon Web Services. [Online].

Available: https://aws.amazon.com/s3. [Accessed: 29 May, 2025].

[21] Amazon Web Services, "AWS Fargate," Amazon Web Services. [Online]. Available: https://aws.amazon.com/fargate. [Accessed: 29 May, 2025].

[22] Google Cloud, "Google Cloud Platform Overview," Google Cloud. [Online]. Available: https://cloud.google.com. [Accessed: 29 May, 2025].

[23] Google Cloud, "Compute Engine Documentation," Google Cloud. [Online]. Available: https://cloud.google.com/compute. [Accessed: 29 May, 2025].

[24] Nginx, Inc., "NGINX Documentation," Nginx. [Online].

Available: https://docs.nginx.com. [Accessed: 29 May, 2025].