Tài liệu Đặc tả Thiết kế Giải pháp (Solution Design Specification - SDS)

Tự động hóa xử lý email yêu cầu hỗ trợ khách hàng bằng AI

Hoàng Anh Tuấn, Nguyễn Hữu Luân, Nguyễn Cao Bách

30/03/2025

Mục lục

Quản lý Tài liệu

Phiên bản	1.0
Trạng thái	Dự thảo (Draft)
Ngày	30/03/2024
Tác giả	Hoàng Anh Tuấn, Nguyễn Hữu Luân, Nguyễn Cao Bách
Giảng viên hướng dẫn	PGS.TS Nguyễn Ngọc Hoá

Bảng 1: Lịch sử Phiên bản

Phiên bản	**Ngày**	**Tác giả**	**Nội dung Thay đổi**
1.0	30/04/2024	H.A. Tuấn, N.H. Luân, N.C. Bách	Bổ sung tổng quan kiến trúc, chi tiết thành phần, luồng xử lý.

1 Giới thiệu

1.1 Mục đích

Tài liệu này cung cấp thiết kế kỹ thuật chi tiết cho dự án "Tự động hóa xử lý email yêu cầu hỗ trợ khách hàng bằng AI", bao gồm kiến trúc hệ thống, tương tác giữa các thành phần, luồng hoạt động và công nghệ sử dụng.

1.2 Tổng quan Dự án

Như đã định nghĩa trong Project Charter, dự án này nhằm mục đích tự động hóa việc xử lý email hỗ trợ khách hàng. Hệ thống sẽ tích hợp với Gmail, sử dụng AI (Gemini LLM và Embedding) để phân loại email và truy xuất thông tin liên quan từ cơ sở dữ liệu (FAQs, thông tin sản phẩm), tạo phản hồi, và chuyển các trường hợp phức tạp cho đội ngũ hỗ trợ qua Telegram.

1.3 Phạm vi

1.3.1 Trong Phạm vi

- Tích hợp với Gmail API để nhận email (thông qua Push Notifications) và gửi phản hồi.
- Xây dựng mô-đun LLMService sử dụng Gemini LLM để phân loại email (spam hay không) và tạo phản hồi.
- Xây dựng cơ chế truy xuất thông tin sử dụng Gemini Embeddings và cơ sở dữ liệu PostgreSQL chứa FAQs và thông tin sản phẩm.
- Tích hợp với Telegram Bot API để chuyển tiếp các email được phân loại là phức tạp đến một nhóm chat hỗ trợ được chỉ định.

1.3.2 Ngoài Phạm vi

- Giao diện quản lý thủ công trong hệ thống này.
- Tích hợp với các hệ thống CRM bên ngoài.
- Quản lý người dùng/quyền nâng cao ngoài việc xử lý khóa API cơ bản.
- Việc huấn luyện (training) hay tinh chỉnh (fine-tuning) mô hình LLM (sử dụng các mô hình Gemini có sẵn qua API).

1.4 Tài liệu Tham khảo

- Project Charter: Tự động hóa xử lý email yêu cầu hỗ trợ khách hàng bằng AI (ngày 25/3/2025)
- Tài liệu Gmail API: https://developers.google.com/gmail/api/
- Tài liệu Google AI Gemini API: https://ai.google.dev/docs
- Tài liệu Telegram Bot API: https://core.telegram.org/bots/api
- Tài liệu PostgreSQL: https://www.postgresql.org/docs/

2 Giả định và Ràng buộc

2.1 Giả định

- Thông tin xác thực truy cập (khóa API, chuỗi kết nối CSDL) cho Gmail, Google AI (Gemini), Telegram, và CSDL có sẵn và được bảo mật.
- Nhóm Telegram cho nhân viên hỗ trợ đã tồn tại và bot có quyền đăng tin nhắn.

2.2 Ràng buộc

- **Công nghệ:** Giới hạn trong Python, Gemini (LLM & Embedding), Gmail API, Telegram API, PostgreSQL như đã chỉ định.
- **Giới hạn API:** Chịu sự chi phối của giới hạn tần suất (rate limits) và quotas của các API bên ngoài (Gmail, Gemini, Telegram).

3 Yêu cầu

Phần này mô tả các yêu cầu chức năng và phi chức năng mà hệ thống tự động hóa xử lý email hỗ trợ khách hàng cần đáp ứng.

3.1 Yêu cầu Chức năng (Functional Requirements)

Các yêu cầu này xác định các chức năng cụ thể mà hệ thống phải thực hiện:

- Hệ thống phải có khả năng tiếp nhận email mới từ một tài khoản Gmail được chỉ định thông qua cơ chế Push Notifications của Gmail API.
- Hệ thống phải có khả năng trích xuất các thông tin cơ bản từ email nhận được, bao gồm người gửi, tiêu đề và nội dung chính (ưu tiên dạng văn bản thuần túy - plain text).
- Hệ thống phải sử dụng mô hình ngôn ngữ lớn (Gemini LLM) để phân loại email nhận được thành các loại chính: Spam/Bỏ qua, Xử lý
- Hệ thống phải bỏ qua (không xử lý tiếp) các email được phân loại là Spam.
- Đối với các email được phân loại là xử lý, hệ thống phải tạo vector embedding cho nội dung email bằng Gemini Embedding API.
- Hệ thống phải truy vấn cơ sở dữ liệu PostgreSQL (chứa FAQs, thông tin sản phẩm và embedding tương ứng) để tìm kiếm thông tin liên quan nhất dựa trên sự tương đồng vector (ví dụ: cosine similarity) với embedding của email đến.
- Hệ thống phải đánh giá mức độ liên quan của thông tin truy xuất được dựa trên một ngưỡng tương đồng (similarity threshold) có thể cấu hình.
- Nếu thông tin truy xuất được đánh giá là đủ liên quan (vượt ngưỡng), hệ thống phải sử dụng Gemini LLM để tạo nội dung email phản hồi dựa trên nội dung email gốc của khách hàng và thông tin đã truy xuất từ cơ sở dữ liệu.
- Hệ thống phải gửi email phản hồi đã tạo đến người gửi ban đầu thông qua Gmail API, sử dụng tài khoản Gmail đã chỉ định.
- Nếu thông tin truy xuất từ cơ sở dữ liệu không đủ liên quan (dưới ngưỡng), hệ thống phải chuyển tiếp thông tin chính của email (người gửi, tiêu đề, nội dung) đến một nhóm chat Telegram được chỉ định thông qua Telegram Bot API.

- Hệ thống phải ghi log các bước xử lý chính, các quyết định quan trọng (phân loại, gửi phản hồi, chuyển tiếp) và các lỗi xảy ra trong quá trình hoạt động.

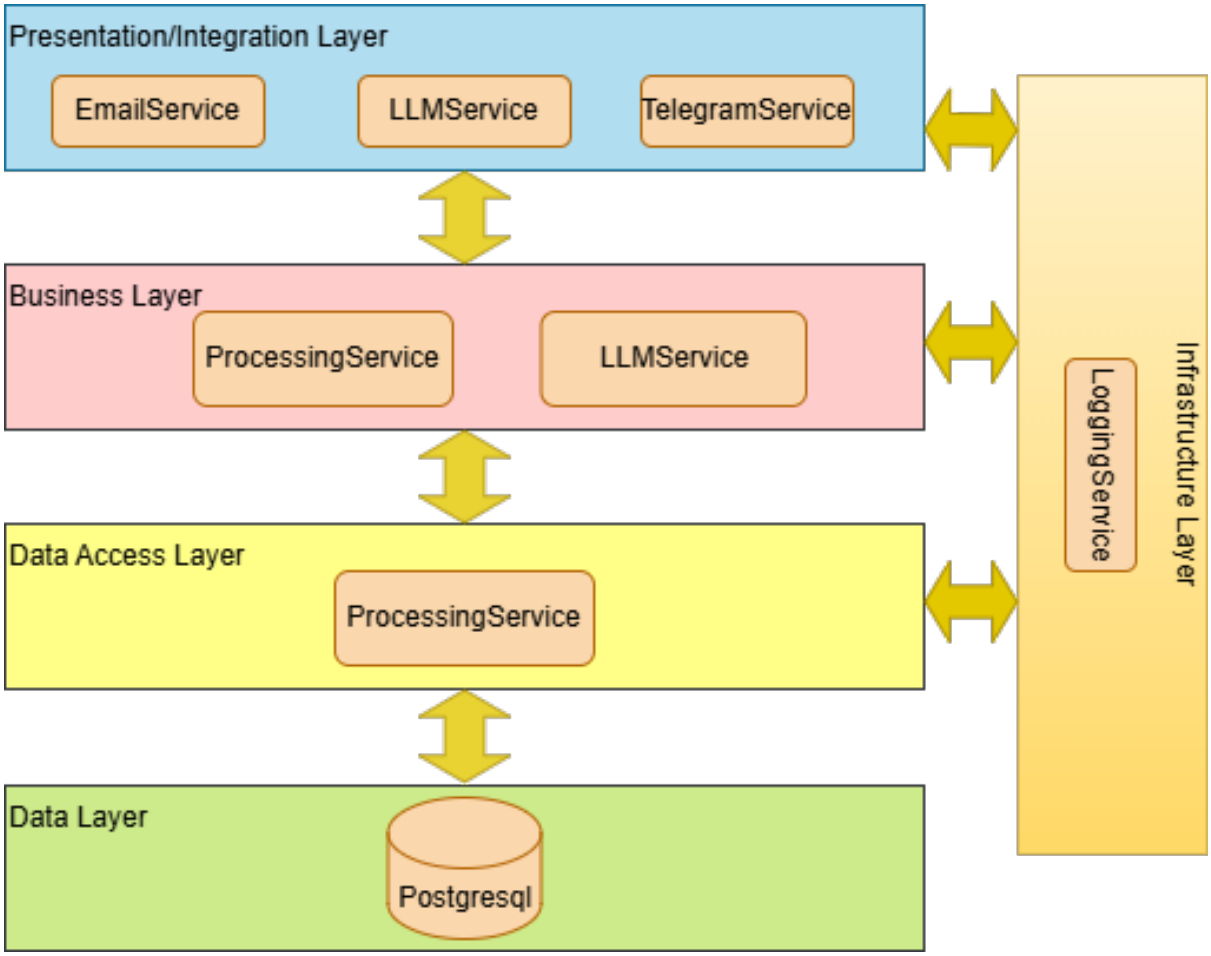
3.2 Yêu cầu Phi chức năng (Non-functional Requirements)

Các yêu cầu này xác định các tiêu chuẩn về chất lượng và cách thức hoạt động của hệ thống:

- **Performance:** Thời gian trung bình từ lúc nhận email đến lúc gửi phản hồi tự động (đối với các trường hợp xử lý tự động thành công) không nên vượt quá 30 giây.
- **Scalability:** Kiến trúc hệ thống phải cho phép mở rộng (scale-out) để xử lý lượng email tăng lên trong tương lai, tận dụng cơ chế của Google Cloud Pub/Sub và khả năng chạy nhiều tiến trình xử lý song song.
- **Security:** Mọi thông tin nhạy cảm như khóa API, thông tin đăng nhập cơ sở dữ liệu phải được lưu trữ an toàn (không hardcode trong mã nguồn) và truy cập một cách bảo mật (ví dụ: qua biến môi trường hoặc dịch vụ quản lý bí mật).
- **Maintainability:** Mã nguồn phải được tổ chức rõ ràng, có bình luận đầy đủ. Các cấu hình hệ thống (ngưỡng, ID, khóa API) phải dễ dàng thay đổi mà không cần sửa đổi mã nguồn.
- **Accuracy:**
  - Tỷ lệ phân loại email chính xác cần đạt mức tối thiểu 90%.
  - Chất lượng của phản hồi tự động được tạo ra phải đảm bảo tính hữu ích, lịch sự và phù hợp với ngữ cảnh của yêu cầu khách hàng (đánh giá định tính).
- **Usability:** Thông báo được gửi đến nhóm Telegram phải rõ ràng, đầy đủ thông tin cần thiết (người gửi, tiêu đề, nội dung tóm tắt/đầy đủ) để nhân viên hỗ trợ có thể nhanh chóng nắm bắt và xử lý.
- **Resource Constraints:** Hệ thống phải hoạt động trong giới hạn về tần suất yêu cầu (rate limits) và quotas của các dịch vụ API được sử dụng (Gmail, Gemini, Telegram).

4 Kiến trúc Giải pháp

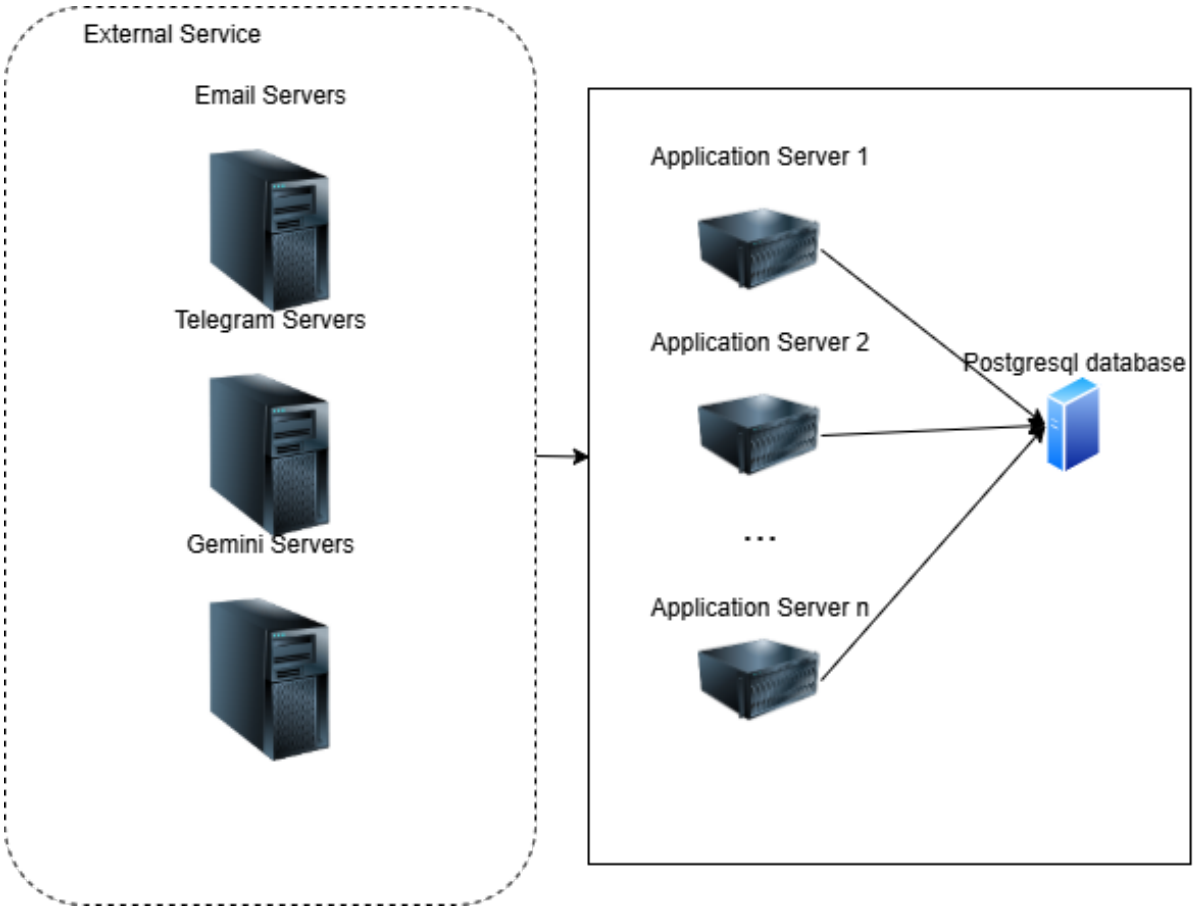
4.1 Sơ đồ kiến trúc logic



Hình 1: Sơ đồ kiến trúc logic

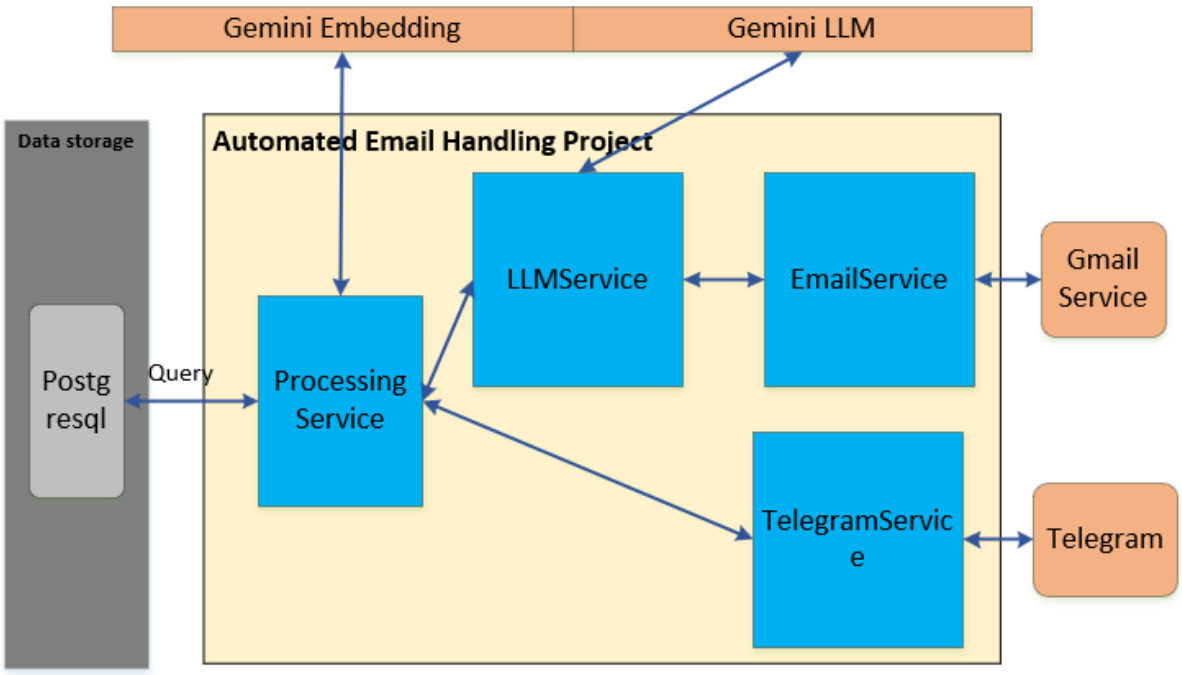
4.2 Sơ đồ kiến trúc vật lý

Với phạm vi môn học, kiến trúc đơn tầng được lựa chọn để nhanh chóng xây dựng sản phẩm một cách đơn giản, toàn bộ service được triển khai monolith trên 1 máy. Tuy nhiên, hệ thống vẫn có thể scale bằng cách chạy trên nhiều máy để tiếp nhận nhiều yêu cầu khi dịch vụ mail của google có khả năng load balance đến các dịch vụ khác nhau cùng thực hiện subscribe đến 1 topic.



Hình 2: Sơ đồ kiến trúc vật lý

4.3 Sơ đồ Kiến trúc các thành phần

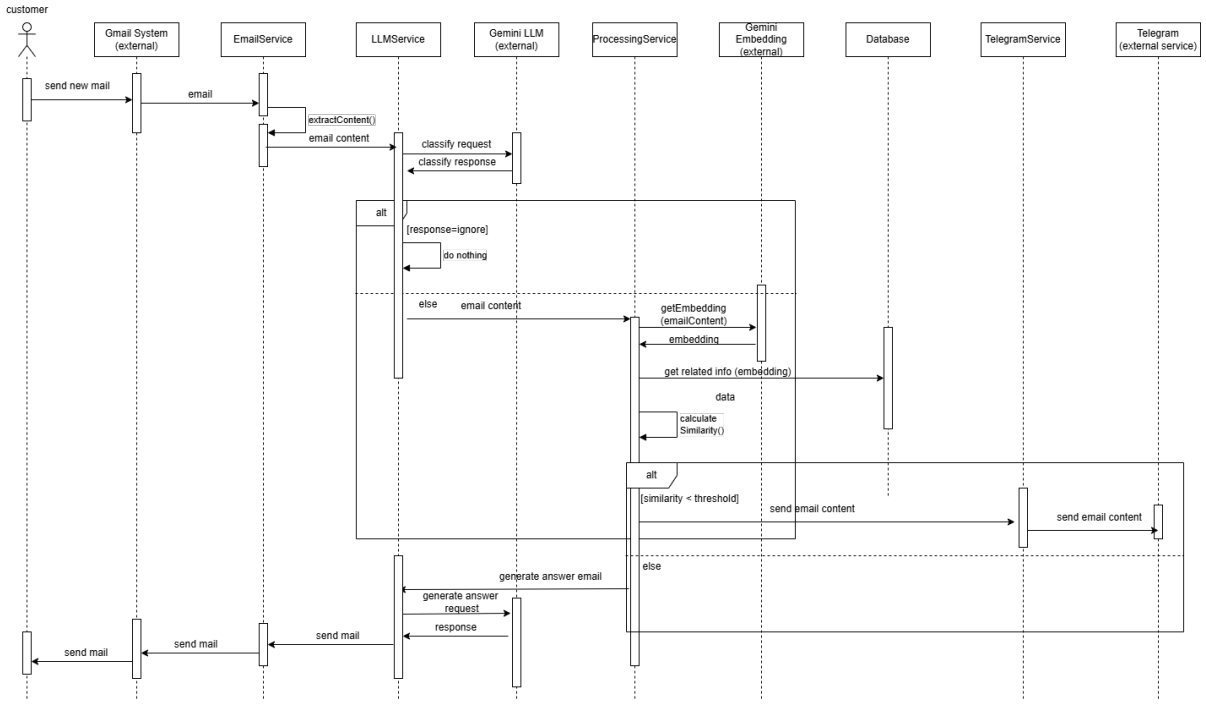


Hình 3: Thành phần kiến trúc tổng thể

- *Hệ thống bên ngoài: Máy chủ Gmail, Máy chủ Telegram, Google AI (Gemini)*
- *Thành phần ứng dụng lõi: Email Service (Gửi/nhận Email), LLM Service (phân loại Email, tạo phản hồi), Processing Service (Truy xuất dữ liệu, cung cấp ngữ cảnh để tạo phản hồi), Telegram Service*
- *Kho dữ liệu: CSDL PostgreSQL (FAQs, Sản phẩm)*
- *Tương tác chính: Push Notification -> EmailService -> LLMService -> Gemini LLM -> ProcessingService -> DB -> (Gmail/Telegram)*

5 Thiết kế Chi tiết

5.1 Sequence Diagram



Hình 4: Luồng hoạt động hệ thống

- `customer` gửi một thư mới đến `Gmail System (external)`.
- `Gmail System (external)` chuyển tiếp `email` đó đến `EmailService`.
- `EmailService` trích xuất nội dung (`email content`) từ `email`.
- `EmailService` gửi `email content` đến `LLMService` để phân loại.
- `LLMService` gửi yêu cầu phân loại (`classify request`) đến `Gemini LLM (external)`.
- `Gemini LLM (external)` trả về kết quả phân loại (`classify response`) cho `LLMService`.
- `LLMService` kiểm tra kết quả phân loại:
  - **Nếu** kết quả là bỏ qua (`response=ignore`):
    - `LLMService` không làm gì tiếp.
  - **Ngược lại** (nếu không phải bỏ qua):
    - `LLMService` gửi `email content` đến `ProcessingService`.
    - `ProcessingService` yêu cầu `Gemini Embedding (external)` tạo embedding cho `email content`.
    - `Gemini Embedding (external)` trả về `embedding` cho `ProcessingService`.
    - `ProcessingService` sử dụng `embedding` để truy vấn thông tin liên quan từ `Database`.
    - `Database` trả về dữ liệu (`data`) liên quan cho `ProcessingService`.
    - `ProcessingService` tính toán độ tương tự (`Similarity`).

* `ProcessingService` kiểm tra độ tương tự:
  · **Nếu** độ tương tự thấp hơn ngưỡng (`similarity < threshold`):
  · `ProcessingService` gửi `email content` đến `TelegramService`.
  · `TelegramService` gửi `email content` đến `Telegram (external service)`.
  · **Ngược lại** (nếu độ tương tự không thấp hơn ngưỡng):
  · `ProcessingService` yêu cầu `LLMService` tạo email trả lời.
  · `LLMService` tạo nội dung email trả lời.
  · `LLMService` gửi email trả lời đã tạo đến `EmailService`.
  · `EmailService` gửi email trả lời qua `Gmail System (external)`.
  · `Gmail System (external)` gửi email trả lời đến `customer` ban đầu.

5.2 EmailService (Gửi, nhận email, phân tích)

- **Công nghệ:** Python, Thư viện Google API Client cho Python.
- **Kích hoạt:** Chủ đề (topic) Google Cloud Pub/Sub được cấu hình cho Gmail Push Notifications trên hộp thư đích. Một subscription sẽ đẩy tin nhắn đến một điểm cuối (endpoint) do thành phần này host (ví dụ: một web server nhỏ như Flask/FastAPI hoặc một worker chạy nền).
- **Chức năng:**
  - Nhận tin nhắn thông báo từ Pub/Sub..
  - Sử dụng Gmail API để lấy chi tiết email đầy đủ (người gửi, tiêu đề, nội dung - ưu tiên dạng plain text).
  - Thực hiện phân tích/làm sạch cơ bản nội dung email (ví dụ: loại bỏ chữ ký, các trả lời trước đó nếu có thể).
  - Chuyển dữ liệu email đã xử lý (người gửi, tiêu đề, nội dung, message ID) đến LLM Service
  - Xử lý các lỗi API tiềm ẩn (xác thực, giới hạn tần suất, không tìm thấy tin nhắn).
  - Nhận phản hồi từ LLMService để tự động gửi email phản hồi người dùng
- **Giao diện Tương tác:** Nhận tin nhắn Pub/Sub, Gọi Gmail API, Gọi/Nhận LLMService.

5.3 LLM Service (Phân loại & Phản hồi)

- **Công nghệ:** Python, Google AI Python SDK (cho Gemini).
- **Chức năng:**
  - **Phân loại:**
  - Nhận nội dung email từ Dịch vụ Xử lý.
  - Xây dựng prompt cho Gemini LLM API được thiết kế đặc biệt để phân loại ý định của email
  - Gọi Gemini LLM API.
  - Phân tích phản hồi API để trích xuất danh mục phân loại.

- Trả kết quả phân loại về cho Dịch vụ Xử lý.
- Xử lý lỗi API và các phân loại không rõ ràng tiềm ẩn.
- **Tạo Phản hồi:**
- Nhận thông tin, ngữ cảnh từ ProcessingService.
- Xây dựng prompt cho Gemini LLM API hướng dẫn nó tạo ra một phản hồi hữu ích dựa trên ngữ cảnh, giải đáp truy vấn của người dùng.
- Gọi Gemini LLM API.
- Phân tích phản hồi để lấy nội dung email được tạo ra.
- Trả phản hồi được tạo ra về cho EmailService.
- Xử lý lỗi API.

- **Giao diện Tương tác:** Được gọi bởi Dịch vụ Xử lý, Gọi Gemini LLM API.

5.4 ProcessingService (Truy xuất thông tin CSDL, đánh giá độ tương đồng)

- **Công nghệ:** Python, Trình kết nối PostgreSQL, Google AI Python SDK (cho Gemini Embeddings), có thể dùng NumPy/SciPy cho các phép toán vector.
- **Chức năng:**
  - **Tạo Embedding (cho email đến):** Nhận nội dung email, gọi Gemini Embedding API để lấy biểu diễn vector.
  - **Truy vấn CSDL:**
  - Kết nối đến CSDL PostgreSQL.
  - Thực thi các truy vấn SQL để tìm thông tin liên quan. Bao gồm:
  - Tính toán độ tương đồng cosine (hoặc thước đo khoảng cách khác) giữa embedding của email đến và các embedding đã được tính toán trước lưu trong CSDL (yêu cầu hỗ trợ vector trong CSDL hoặc truy xuất và tính toán trong bộ nhớ).
  - Chọn N mục FAQ hoặc mô tả sản phẩm liên quan nhất dựa trên điểm tương đồng.
  - Có thể dự phòng bằng tìm kiếm từ khóa nếu tìm kiếm embedding không mang lại kết quả tốt.
  - Trả về các thông tin, ngữ cảnh được truy xuất cho LLMService tạo phản hồi email hoặc gọi TelegramService để forward email đến CSKH nếu không thể tự xử lý
- **Giao diện Tương tác:** Gọi/Nhận LLMService, Gọi TelegramService, Gọi Gemini Embedding API, Tương tác với CSDL.

5.5 Telegram Service

- **Công nghệ:** Python, Thư viện Telegram Bot API.
- **Kích hoạt:** Được gọi bởi ProcessingService khi một email cần sự xử lý của con người.
- **Chức năng:**
  - Nhận thông tin cần thiết (người gửi gốc, tiêu đề, đoạn nội dung/liên kết...) từ ProcessingService.
  - Định dạng thông tin này thành một tin nhắn rõ ràng cho nhân viên hỗ trợ.
  - Sử dụng Telegram Bot Token và Chat ID đã cấu hình để gửi tin nhắn đến nhóm hỗ trợ được chỉ định thông qua Telegram Bot API.
  - Xử lý các lỗi API tiềm ẩn (token không hợp lệ, không tìm thấy chat, giới hạn tần suất).
- **Giao diện Tương tác:** Được gọi bởi ProcessingService, Gọi Telegram Bot API.

6 Thiết kế Dữ liệu

- **Bảng:** `knowledge_base`
  - `id` (SERIAL, Khóa chính)
  - `source_type` (VARCHAR(50), Not Null, Phân loại nguồn: 'faq', 'product')
  - `source_id` (VARCHAR(100), Tùy chọn, ID của nguồn, ví dụ: FAQ ID, product SKU)
  - `content` (TEXT, Not Null, Nội dung text gốc dùng để nhúng vector)
  - `price` (NUMERIC(10, 2), Tùy chọn, Giá sản phẩm nếu `source_type` là 'product')
  - `embedding` (VECTOR(768), Not Null, Vector nhúng của trường `content`)
  - `last_updated` (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP, Thời gian cập nhật lần cuối)

7 Công nghệ Sử dụng

- **Ngôn ngữ Lập trình:** Python (Phiên bản 3.8+)
- **Thư viện Cốt lõi:** Google API Client cho Python, Google AI Python SDK, python-telegram-bot, psycopg2, requests, Flask/FastAPI (Tùy chọn, cho endpoint của listener).
- **Dịch vụ AI:** Google Gemini API (LLM cho phân loại/tạo phản hồi, mô hình Embedding)
- **Dịch vụ Email:** Gmail API (bao gồm Push Notifications qua Google Cloud Pub/Sub)
- **Dịch vụ Nhắn tin:** Telegram Bot API
- **Cơ sở Dữ liệu:** PostgreSQL với extension pgvector.

8 Bảo mật

- **Khóa API/Thông tin Xác thực:** Tất cả API credential (Gmail, Google AI, Telegram) và thông tin kết nối CSDL phải được lưu trữ an toàn (ví dụ: biến môi trường, dịch vụ quản lý bí mật) và không được hardcode trong mã nguồn.
- **Làm sạch Đầu vào (Input Sanitization):** Mặc dù đầu vào là từ email, cần cẩn thận nếu phân tích nội dung HTML; chủ yếu sử dụng nội dung dạng plain text.
- **Xử lý Lỗi:** Đảm bảo lỗi không tiết lộ chi tiết hệ thống nhạy cảm trong log hoặc phản hồi.

9 Triển khai

- **Khả năng mở rộng:** Dễ dàng scale (có thể cấu hình load balancing ở google cloud service, tự động load balance email đến các service khác nhau có cùng subscription khi thêm service Python)
- **Cấu hình:** Đưa cấu hình ra bên ngoài (API credential, kết nối CSDL, chat ID Telegram, ngưỡng phân loại) thông qua biến môi trường hoặc tệp cấu hình.
- **Phụ thuộc:** Sử dụng 'requirements.txt' để quản lý các phụ thuộc Python.
- **Môi trường triển khai:** Với phạm vi xây dựng prototype phục vụ môn học, service và db được triển khai trên máy tính cá nhân.

10 Xử lý Lỗi và Giảm thiểu Rủi ro

- **AI phân loại sai email (Rủi ro):**
  - **Biện pháp Giảm thiểu:** Xây dựng prompt cẩn thận cho việc phân loại. Triển khai ngưỡng tin cậy; nếu độ tin cậy phân loại từ LLM (nếu có) thấp, mặc định chuyển tiếp đến Telegram. Ghi log các phân loại để xem xét và tinh chỉnh prompt sau này. (Tối ưu prompt).
- **AI chọn sai thông tin để phản hồi khách hàng (Rủi ro):**
  - **Biện pháp Giảm thiểu:** Tối ưu chiến lược embedding (chia nhỏ văn bản, chọn mô hình phù hợp). Truy xuất nhiều đoạn ngữ cảnh và hướng dẫn LLM trong prompt tổng hợp thông tin cẩn thận và nêu rõ khi không thể trả lời chắc chắn dựa trên ngữ cảnh được cung cấp.
- **Hệ thống email không ổn định (Rủi ro - Gmail API / Push Notifications):**
  - **Biện pháp Giảm thiểu:** Triển khai xử lý lỗi mạnh mẽ và cơ chế thử lại (retry) với backoff cho tất cả các lệnh gọi Gmail API. Thêm giám sát để phát hiện lỗi trong việc nhận thông báo đẩy hoặc lỗi gọi API. Cân nhắc cơ chế kiểm tra định kỳ (polling) làm phương án dự phòng nếu thông báo đẩy không đáng tin cậy (mặc dù kém hiệu quả hơn). (Xây dựng cơ chế dự phòng).
- **Giới hạn Tần suất/Lỗi API (Chung):**

- **Biện pháp Giảm thiểu:** Triển khai xử lý lỗi thích hợp cho các mã trạng thái HTTP (ví dụ: 429 Too Many Requests, 5xx Server Errors) từ tất cả các API bên ngoài. Bao gồm logic thử lại với exponential backoff. Ghi log lỗi API rõ ràng.

- **CSDL không khả dụng:**

  - **Biện pháp Giảm thiểu:** Triển khai connection pooling và logic thử lại cho kết nối/truy vấn CSDL. Ghi log lỗi CSDL. Tùy thuộc vào mức độ quan trọng, có thể mặc định chuyển tiếp đến Telegram nếu CSDL gặp sự cố.