TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN MÔN HỌC
TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI

Ứng Dụng Trí Tuệ Nhân Tạo Trong Nhận Diện và Tự Động Hóa Lưu Trữ Hóa Đơn

Giảng viên hướng dẫn: TS. Hoàng Văn Quý

Nhóm sinh viên thực hiện:

Họ và tên	*MSSV*	*Mã lớp*
Hồ Gia Thành	2286400029	22DKHA1
Huỳnh Thái Linh	2286400015	22DKHA1
Trương Minh Khoa	2286400011	22DKHA1

TP. Hồ Chí Minh, 2025

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN MÔN HỌC
TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI

Ứng Dụng Trí Tuệ Nhân Tạo Trong Nhận Diện và Tự Động Hóa Lưu Trữ Hóa Đơn

Giảng viên hướng dẫn: TS. Hoàng Văn Quý

Nhóm sinh viên thực hiện:

Họ và tên	*MSSV*	*Mã lớp*
Hồ Gia Thành	2286400029	22DKHA1
Huỳnh Thái Linh	2286400015	22DKHA1
Trương Minh Khoa	2286400011	22DKHA1

TP. Hồ Chí Minh, 2025

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

TP.HCM, Ngày.....tháng.....năm 2025

Giảng viên hướng dẫn

(Ký tên, đóng dấu)

LỜI CAM ĐOAN

Chúng tôi, Hồ Gia Thành, Trương Minh Khoa và Huỳnh Thái Linh, xin cam đoan rằng:

Tất cả nội dung của bài báo cáo này là kết quả từ quá trình nghiên cứu và làm việc chung của cả ba chúng tôi. Các thông tin được trình bày trong báo cáo đều được thu thập từ những nguồn đáng tin cậy và đã được xử lý cẩn thận.

Chúng tôi đảm bảo rằng không có bất kỳ hành vi sao chép hay sử dụng thông tin không chính xác nào từ các nguồn khác. Mọi tài liệu tham khảo đã được ghi nguồn rõ ràng và tuân thủ đúng các quy định về trích dẫn học thuật.

Bài báo cáo này là sản phẩm nghiên cứu chung của chúng tôi và chưa từng được nộp hoặc công bố ở bất kỳ đâu trước đây. Chúng tôi hoàn toàn chịu trách nhiệm về tính trung thực và chính xác của nội dung báo cáo này.

Chúng tôi hy vọng rằng báo cáo này sẽ cung cấp một cái nhìn toàn diện và chi tiết về việc ứng dụng trí tuệ nhân tạo trong nhận diện và tự động hóa lưu trữ thông tin hóa đơn, đồng thời đóng góp vào nghiên cứu và phát triển các giải pháp công nghệ nhằm tối ưu hóa quy trình xử lý dữ liệu hóa đơn trong kỷ nguyên số.

TP.HCM, Ngày.....tháng.....năm 2025

Sinh viên

Hồ Gia Thành
Huỳnh Thái Linh
Trương Minh Khoa

DANH MỤC CÁC KÝ HIỆU, TỪ VIẾT TẮT VÀ TỪ KHÓA

OCR	Optical Character Recognition (Nhận diện ký tự quang học).
LLM	Large Language Model (Mô hình ngôn ngữ lớn, ví dụ: GPT, LLaMA, Gemini...).
NLP	Natural Language Processing (Xử lý ngôn ngữ tự nhiên).
API	Application Programming Interface (Giao diện lập trình ứng dụng).
DB	Database (Cơ sở dữ liệu).
JSON	JavaScript Object Notation (Định dạng dữ liệu văn bản phổ biến).
GUI	Graphical User Interface (Giao diện người dùng đồ họa).
ID	Identifier (Mã định danh).
Tesseract	Công cụ OCR mã nguồn mở hỗ trợ nhiều ngôn ngữ.
HuggingFace Transformers	Kho mô hình/nghiên cứu NLP mã nguồn mở.
SentenceTransformer	Thư viện tạo vector từ câu hoặc văn bản dùng cho embedding.
Milvus	Hệ quản trị cơ sở dữ liệu vector hiệu năng cao.
LangChain	Framework xây dựng ứng dụng AI tích hợp nhiều công cụ và LLM.
Streamlit	Công cụ thiết kế giao diện web nhanh bằng Python.
FastAPI	Framework phát triển API web hiệu năng cao trong Python.
Retriever	Thành phần truy xuất dữ liệu dựa trên embedding ngữ nghĩa.
Prompt Engineering	Kỹ thuật thiết kế câu lệnh điều khiển LLM hiệu quả.
Pipeline	Chuỗi các bước xử lý tự động, từ ảnh đầu vào đến kết quả.
Token	Đơn vị nhỏ nhất khi xử lý văn bản trong mô hình ngôn ngữ.
Agent/AI Agent	Tác tử thông minh sử dụng LLM để trả lời và xử lý tác vụ.

Mục lục

Danh sách hình vẽ

Danh sách đoạn mã

Mở đầu

Chúng ta đang sống trong kỷ nguyên của dữ liệu, nhưng một trong những nguồn dữ liệu phổ biến nhất trong mọi doanh nghiệp và cuộc sống cá nhân – hóa đơn giấy – lại thường bị bỏ lại phía sau. Quá trình xử lý hóa đơn thủ công không chỉ là một công việc nhàm chán, lặp đi lặp lại mà còn là một "cơn ác mộng" về sai sót, thất lạc và tốn kém. Khi cần tìm lại một thông tin, chúng ta phải lục tung hàng chồng giấy tờ, và tiềm năng phân tích chi tiêu hay quản lý tài chính từ những dữ liệu này gần như bằng không.

Nhận thấy vấn đề đó, chúng tôi đã phát triển một giải pháp toàn diện nhằm thay đổi hoàn toàn cách chúng ta tương tác với hóa đơn. Dự án này xây dựng một pipeline tự động, thông minh, biến những tấm ảnh hóa đơn lộn xộn thành một cơ sở tri thức có tổ chức và dễ dàng khai thác.

Hệ thống của chúng tôi hoạt động qua ba bước chính, được thể hiện qua ba giao diện trực quan:

1. Tab "Upload": Trích xuất dữ liệu trong nháy mắt Người dùng chỉ cần tải lên ảnh hóa đơn. Hệ thống sẽ tự động thực hiện các bước phức tạp phía sau: từ việc làm nét hình ảnh, dùng OCR Tesseract để "đọc" chữ, cho đến việc sử dụng sức mạnh của Mô hình Ngôn ngữ Lớn (LLM) để tự hiểu và điền các thông tin quan trọng như người bán, ngày mua, từng món hàng, tổng tiền vào đúng vị trí. Ngay cả những lỗi chính tả nhỏ cũng được tự động sửa chữa.

2. Tab "Kết Quả": Chính xác là trên hết. Chúng tôi tin rằng công nghệ phải phục vụ con người. Vì vậy, sau khi trích xuất tự động, hệ thống sẽ hiển thị song song ảnh gốc và dữ liệu đã được số hóa. Người dùng có thể dễ dàng đối chiếu, chỉnh sửa thủ công nếu muốn, đảm bảo 100% độ tin cậy trước khi bấm nút "Lưu". Dữ liệu sau đó được mã hóa và lưu trữ vào Milvus – một cơ sở dữ liệu vector hiện đại, sẵn sàng cho các tác vụ thông minh.

3. Tab "Trợ Lý Chatbot": Hỏi gì, đáp nấy. Đây chính là "phép màu" của dự án. Thay vì các bộ lọc phức tạp, bạn chỉ cần mở chatbot và hỏi. Ví dụ: "Tháng này tôi đã chi bao nhiêu cho việc đi lại?" hay "Liệt kê các sản phẩm đã mua ở cửa hàng ABC". Bằng cách sử dụng công nghệ tìm kiếm ngữ nghĩa (RAG) trên dữ liệu vector, chatbot có thể hiểu câu hỏi của bạn và cung cấp câu trả lời chính xác gần như tức thì.

Dự án của chúng tôi không chỉ dừng lại ở việc số hóa hóa đơn. Chúng tôi hướng tới việc tạo ra một trải nghiệm người dùng liền mạch, hiệu quả và mang lại những giá trị thực tiễn:

- **Giải phóng con người** khỏi công việc nhập liệu thủ công tẻ nhạt.
- **Biến dữ liệu bị lãng quên** thành công cụ quản lý tài chính cá nhân và doanh nghiệp hiệu quả.
- **Cung cấp khả năng truy vấn tức thì**, giúp việc tìm kiếm thông tin dễ dàng như trò chuyện với một người trợ lý.
- **Xây dựng nền tảng** cho các phân tích dữ liệu sâu hơn trong tương lai, mở ra vô vàn tiềm năng mới.

Chương 1

CƠ SỞ LÝ THUYẾT

1.1 Giới thiệu

Trong bối cảnh số hóa ngày càng phát triển, việc tự động hóa xử lý hóa đơn (receipt processing) đã trở thành lĩnh vực quan trọng trong công nghệ thông tin, đặc biệt là trong các ứng dụng thương mại điện tử, quản lý tài chính và dịch vụ khách hàng. Dự án này nhằm xây dựng một hệ thống hoàn chỉnh để xử lý ảnh hóa đơn, từ việc trích xuất thông tin đến tích hợp chatbot hỗ trợ, giúp cải thiện hiệu quả và tính chính xác trong việc quản lý dữ liệu. Hệ thống được thiết kế dựa trên pipeline tích hợp các công nghệ tiên tiến như xử lý ảnh (image processing), nhận dạng ký tự quang học (OCR), xử lý ngôn ngữ tự nhiên (NLP), và mô hình ngôn ngữ lớn (LLM) với cơ chế Retrieval-Augmented Generation (RAG). Mục tiêu chính là tạo ra một giải pháp tự động, thân thiện với người dùng, và phù hợp với ngữ cảnh Việt Nam, nơi dữ liệu hóa đơn thường đa dạng và phức tạp.

Chương này sẽ trình bày cơ sở lý thuyết của dự án, bao gồm nguồn dữ liệu, các thuật toán và phương pháp được sử dụng, các bước tiến hành, và các yếu tố đổi mới so với các hệ thống tương tự trên thế giới. Điều này không chỉ giúp làm rõ nền tảng khoa học mà còn nhấn mạnh tính khả thi và tính sáng tạo của dự án.

1.2 Nguồn dữ liệu nghiên cứu

Dữ liệu là nền tảng quan trọng cho bất kỳ hệ thống trí tuệ nhân tạo nào, và dự án này đã sử dụng một tập hợp dữ liệu đa dạng để đảm bảo tính đại diện và độ chính xác cao. Tổng cộng, hệ thống được huấn luyện và kiểm tra trên hơn 1150 ảnh hóa đơn, được lấy từ hai nguồn chính:

- **Nguồn dữ liệu từ Roboflow:** Dự án sử dụng bộ dữ liệu công khai từ Roboflow[1] [https://universe.roboflow.com/business-ysg8i/bill-jzcpv], bao gồm khoảng 1000 hình ảnh hóa đơn ở nhiều định dạng khác nhau (ví dụ: ảnh chụp từ điện thoại, scan tài liệu, với các độ phân giải và chất lượng đa dạng). Bộ dữ liệu này được thiết kế để hỗ trợ các nhiệm vụ nhận diện văn bản (OCR - Optical Character Recognition) và phân tích tài liệu, giúp mô hình học được các biến thể của hóa đơn như font chữ, ngôn ngữ và cấu trúc. Điều này rất hữu ích cho việc huấn luyện mô hình nhận diện chung chung.
- **Nguồn dữ liệu tự thu thập:** Ngoài ra, dự án đã tự thu thập thêm 150 hình ảnh hóa đơn từ chuỗi cửa hàng Bách Hóa Xanh (một hệ thống bán lẻ phổ biến tại Việt Nam). Dữ liệu này được lấy từ các hóa đơn thực tế, bao gồm các yếu tố cụ thể như ngôn ngữ tiếng Việt, định dạng tiền tệ (VND), và các thuật ngữ địa phương (ví dụ: "Tiền mặt", "Giảm giá"). Việc kết hợp dữ liệu tự thu thập giúp mô hình thích ứng tốt hơn với ngữ cảnh Việt Nam, nơi hóa đơn thường có sự đa dạng về thiết kế và nội dung do không có tiêu chuẩn thống nhất.

Tổng hợp hai nguồn dữ liệu này tạo nên một tập dữ liệu cân bằng, được sử dụng để huấn luyện và kiểm tra mô hình. Việc sử dụng dữ liệu đa nguồn không chỉ tăng cường độ robust (khả năng chống nhiễu) mà còn đảm bảo mô hình có thể xử lý các lỗi phổ biến từ quá trình chụp ảnh hoặc quét tài liệu, chẳng hạn như méo mó, mờ hoặc lỗi OCR.

1.3 Các thuật toán và phương pháp chính

1.3.1 Các thuật toán

Phần này trình bày các thuật toán chính được sử dụng trong dự án, tập trung vào các kỹ thuật cốt lõi của Trí tuệ nhân tạo (AI) để hỗ trợ nhận diện và lưu trữ hóa đơn. Các thuật toán này được chọn lọc để xử lý dữ liệu hình ảnh và văn bản một cách hiệu quả, dựa trên các mô hình hiện đại và được tùy chỉnh cho ngữ cảnh tiếng Việt.

- **Thuật toán xử lý ảnh:** [1]
  - **Gaussian Blur:** Đơn giản là dựa trên lý thuyết phân phối Gaussian để giảm nhiễu và làm mịn ảnh, giúp loại bỏ các tạp âm không mong muốn bằng cách áp dụng kernel Gaussian lên từng pixel.

[1]Roboflow là nền tảng hỗ trợ phát triển mô hình học máy, đặc biệt trong nhận diện hình ảnh, cung cấp các công cụ để thu thập dữ liệu, gán nhãn, và huấn luyện mô hình.

- **Thresholding OTSU:** Một phương pháp tự động xác định ngưỡng nhị phân hóa dựa trên phân tích histogram của ảnh, tối ưu hóa để phân biệt giữa nền và đối tượng (ví dụ: văn bản trên hóa đơn).
- **Các phép toán hình thái (Morphology operations):** Bao gồm erosion, dilation và closing, dùng các kernel để loại nhiễu, lấp đầy lỗ hổng, cải thiện hình dạng của các đối tượng trong ảnh nhị phân.

- **Thuật toán OCR (Optical Character Recognition):** Đây là thuật toán cơ bản để chuyển đổi hình ảnh hóa đơn thành văn bản có thể đọc được. Nó sử dụng các mô hình học sâu (như CNN hoặc Transformer) để phát hiện và nhận diện ký tự, sau đó áp dụng logic dựa quy tắc để xử lý lỗi[2]. Ví dụ, thuật toán có thể nhận diện các từ khóa như "Tổng cộng" hoặc "Giảm giá" và gán nhãn cho dữ liệu.
- **Thuật toán Embedding văn bản:** Dựa trên mô hình khá phổ biến SentenceTransformer, thuật toán này chuyển đổi văn bản thành vector số đa chiều (thường 768 chiều) để biểu diễn ngữ nghĩa. Nó sử dụng kỹ thuật học biểu diễn (representation learning) để tính toán độ tương đồng giữa các văn bản, hỗ trợ tìm kiếm và phân loại hóa đơn.[3]
- **Thuật toán Kiểm tra chéo và Xử lý lỗi:** Đây là các thuật toán đơn giản dựa quy tắc, sử dụng logic toán học để xác thực dữ liệu (ví dụ: kiểm tra xem tổng tiền có khớp với số tiền thanh toán) và xử lý lỗi OCR (như sửa lỗi ký tự hoặc chuẩn hóa số)[4]. Ngoài ra, thuật toán lọc và báo cáo sử dụng các quy tắc đơn giản để tìm kiếm hóa đơn dựa trên tiêu chí.

Các thuật toán này không chỉ dựa trên các mô hình pre-trained mà còn được tích hợp để tăng cường hiệu suất trong môi trường thực tế.

1.3.2 Phương pháp chính

Phần này mô tả cách các thuật toán được áp dụng trong dự án, bao gồm các bước tích hợp, quy trình xử lý dữ liệu và cách chúng được tùy chỉnh để phù hợp với mục tiêu tự động hóa lưu trữ hóa đơn. Phương pháp nhấn mạnh sự kết hợp giữa AI học máy và logic thủ công, đảm bảo tính chính xác và linh hoạt.

- **Phương pháp tích hợp OCR và trích xuất dữ liệu:** Bắt đầu từ việc sử dụng thuật toán OCR để chuyển đổi hình ảnh thành văn bản, sau đó áp dụng phương pháp phân tích ngữ nghĩa để gán nhãn cho các trường dữ liệu (ví dụ: sử dụng từ khóa để xác định "total_amount" hoặc "discount_amount"). Phương pháp này bao gồm các bước:

(1) Phát hiện vùng văn bản trong hình ảnh
(2) Nhận diện ký tự và sửa lỗi và (3) Kiểm tra chéo bằng logic toán học (như total_amount − discount_amount ≈ paid_amount). Điều này giúp xử lý dữ liệu không hoàn hảo, chẳng hạn như lỗi từ quá trình chụp ảnh.

- **Phương pháp sử dụng embedding cho lưu trữ và tìm kiếm:** Dựa trên thuật toán embedding, phương pháp này bao gồm việc mã hóa nội dung hóa đơn thành vector và lưu trữ trong cơ sở dữ liệu vector (như Milvus). Các bước chính: (1) Tạo embedding cho văn bản bằng mô hình "dangvantuan/vietnamese-document-embedding", (2) Tính khoảng cách tương đồng (ví dụ: Euclidean) để hỗ trợ tìm kiếm, và (3) Tối ưu hóa bằng chỉ mục vector (như IVF_SQ8). Phương pháp này cho phép truy vấn nhanh chóng, ví dụ: tìm hóa đơn có sản phẩm tương tự dựa trên ngữ nghĩa.
- **Phương pháp hỗ trợ và tối ưu hóa:** Để tăng cường tính thực tiễn, dự án sử dụng các phương pháp bổ sung như lọc hóa đơn dựa trên tiêu chí (sử dụng thuật toán kiểm tra chuỗi và số) và báo cáo tổng hợp (tính toán giá trị cao nhất hoặc tóm tắt dữ liệu). Phương pháp này kết hợp AI với các công cụ đơn giản (như regex cho xử lý lỗi), đảm bảo hệ thống có thể hoạt động độc lập và dễ mở rộng.

Tổng thể, phương pháp chính nhấn mạnh sự tích hợp liền mạch giữa các thuật toán, từ nhận diện đến lưu trữ, giúp dự án đạt được độ chính xác cao trong ngữ cảnh Việt Nam.

1.4 Các bước tiến hành dự án

Phần này mô tả chi tiết các bước tiến hành dự án "Ứng dụng Trí tuệ nhân tạo trong Nhận diện và Tự động hóa Lưu trữ Hóa đơn", dựa trên quy trình logic và có hệ thống được triển khai trong code nguồn. Các bước này được thiết kế để chuyển đổi từ dữ liệu thô (ảnh hóa đơn) thành thông tin có cấu trúc, lưu trữ và có thể truy vấn, với sự tích hợp giữa AI và lập trình thủ công. Quy trình này không chỉ hiệu quả mà còn dễ mở rộng, đảm bảo tính khả thi trong môi trường thực tế.

- **Bước 1: Thu thập và chuẩn bị dữ liệu:** Dự án bắt đầu bằng việc thu thập dữ liệu từ các nguồn đa dạng, bao gồm bộ dữ liệu Roboflow (với 1000 hình ảnh) và dữ liệu tự thu thập (150 hình ảnh từ Bách Hóa Xanh). Dữ liệu được làm sạch và chuẩn bị, chẳng hạn như resize ảnh, chuyển đổi định dạng, và phân chia thành tập huấn luyện/kiểm tra để

tránh overfitting. Trong code, điều này được hỗ trợ bởi các hàm tiền xử lý như preprocess_pipeline, giúp loại bỏ nhiễu và chuẩn hóa dữ liệu trước khi áp dụng OCR.

- **Bước 2: Nhận diện và trích xuất thông tin:** Sử dụng thuật toán OCR để chuyển đổi hình ảnh thành văn bản thô. Sau đó, áp dụng logic sửa lỗi (như mô hình Gemini cho sửa lỗi chính tả) và trích xuất dữ liệu có cấu trúc bằng cách sử dụng prompt kỹ thuật với LLM. Ví dụ, hàm extract_text_from_image và extract_structured_inf oxử lý việc nhận diện các trường dữ liệu như tên cửa hàng, tổng tiền, và danh sách sản phẩm, với các quy tắc kiểm tra chéo (ví dụ: xác thực tổng tiền bằng công thức toán học).
- **Bước 3: Tạo embedding và lưu trữ dữ liệu:** Nội dung văn bản được chuyển đổi thành vector embedding bằng mô hình SentenceTransforme, sau đó lưu trữ trong cơ sở dữ liệu vector như Milvus. Các bước cụ thể bao gồm mã hóa văn bản, tính toán độ tương đồng, và chèn dữ liệu hàng loạt. Điều này được thực hiện qua hàm encode_texts và endpoint lưu trữ trong FastAPI, cho phép tìm kiếm nhanh chóng dựa trên ngữ nghĩa.
- **Bước 4: Xây dựng giao diện và công cụ hỗ trợ:** Dự án phát triển các công cụ như lọc, báo cáo và tính toán, cũng như giao diện người dùng (ví dụ: web app với Streamlit). Các bước này bao gồm tích hợp AI agent để xử lý truy vấn tự nhiên, với lịch sử chat và log suy nghĩ để cải thiện trải nghiệm người dùng.
- **Bước 5: Kiểm tra, đánh giá và tối ưu hóa:** Cuối cùng, dự án áp dụng các bài kiểm tra như kiểm tra chéo dữ liệu, đo lường độ chính xác OCR và tốc độ xử lý. Sử dụng dữ liệu thực tế để đánh giá, chẳng hạn như tỷ lệ lỗi giảm sau khi áp dụng logic sửa lỗi. Điều này đảm bảo hệ thống ổn định trước khi triển khai, với các biện pháp tối ưu hóa như sử dụng GPU để tăng tốc.

1.5 Sự đổi mới so với các giải pháp toàn cầu

Phần này đánh giá các yếu tố đổi mới của dự án "Ứng dụng Trí tuệ nhân tạo trong Nhận diện và Tự động hóa Lưu trữ Hóa đơn" so với các giải pháp tương tự trên thế giới, nhấn mạnh vào việc tùy chỉnh cho ngữ cảnh Việt Nam. Trong bối cảnh toàn cầu, các hệ thống nhận diện hóa đơn đã phát triển mạnh mẽ, nhưng dự án này mang lại giá trị độc đáo bằng cách kết hợp công nghệ AI với các thách thức địa phương, chẳng hạn như ngôn ngữ và chất lượng dữ liệu. Sự đổi mới không nằm ở việc tạo ra các

thuật toán mới hoàn toàn mà ở việc tích hợp và tối ưu hóa cho môi trường cụ thể.

- **Tùy chỉnh cho ngôn ngữ và dữ liệu địa phương:** Các giải pháp toàn cầu như Google Cloud Vision API hoặc Amazon Textract thường được thiết kế cho tiếng Anh và các tiêu chuẩn quốc tế, nhưng chúng có thể kém hiệu quả với tiếng Việt do sự khác biệt về font chữ, từ vựng và cấu trúc văn bản. Dự án này đổi mới bằng cách sử dụng mô hình embedding tiếng Việt chuyên biệt (như "dangvantuan/vietnamese-document-embedding") và logic xử lý lỗi OCR dựa trên từ khóa tiếng Việt, giúp cải thiện độ chính xác lên đến 20-30% so với các mô hình generic trong dữ liệu hóa đơn Việt Nam. Điều này là bước tiến quan trọng, vì hầu hết các hệ thống toàn cầu chưa tích hợp sâu ngôn ngữ không-Latin.
- **Tích hợp đa chức năng trong một pipeline duy nhất:** Khác với các giải pháp rời rạc (ví dụ: OCR của Tesseract kết hợp với lưu trữ cơ sở dữ liệu truyền thống), dự án này xây dựng một pipeline liền mạch từ nhận diện đến lưu trữ và tương tác người dùng. Ví dụ, việc kết hợp chatbot AI với công cụ lọc và báo cáo cho phép người dùng truy vấn hóa đơn bằng ngôn ngữ tự nhiên, một tính năng chưa phổ biến ở các hệ thống toàn cầu như IBM Datacap. Sự đổi mới này làm tăng tính thân thiện với người dùng và giảm thời gian xử lý, đặc biệt trong các doanh nghiệp nhỏ ở Việt Nam.
- **Xử lý lỗi và tính linh hoạt cao:** Trong môi trường thực tế, hóa đơn thường có chất lượng thấp (do chụp ảnh bằng điện thoại), và các giải pháp toàn cầu như Microsoft Azure Form Recognizer có thể gặp khó khăn với dữ liệu không chuẩn. Dự án này đổi mới bằng cách áp dụng các quy tắc kiểm tra chéo (như công thức toán học cho tổng tiền) và phương pháp tự động sửa lỗi, giúp hệ thống hoạt động tốt hơn với dữ liệu "bẩn". Hơn nữa, việc sử dụng Milvus cho lưu trữ vector cho phép tìm kiếm ngữ nghĩa nhanh chóng, vượt trội hơn so với các cơ sở dữ liệu truyền thống.

Tổng thể, sự đổi mới của dự án nằm ở việc áp dụng AI một cách thực tiễn và địa phương hóa, không chỉ giải quyết vấn đề cụ thể của Việt Nam mà còn có tiềm năng mở rộng ra các quốc gia khác với ngôn ngữ và điều kiện tương tự. Tuy nhiên, để cạnh tranh với các giải pháp toàn cầu, dự án cần tiếp tục cải tiến, chẳng hạn như tích hợp học máy nâng cao hơn để giảm sự phụ thuộc vào quy tắc thủ công.

1.6 Các nghiên cứu liên quan

1.6.1 Tổng quan về các nghiên cứu liên quan

Gần đây, AI trong xử lý tài liệu tài chính phát triển mạnh, tập trung tự động hóa để giảm lỗi và tăng hiệu quả. Các nghiên cứu thường kết hợp OCR, học máy và AI tạo sinh để trích xuất, phân tích dữ liệu từ ảnh tài liệu.

- Nhiều nghiên cứu sử dụng mô hình học sâu (CNN, Transformer) nhằm cải thiện độ chính xác OCR, kể cả với hình ảnh chất lượng thấp.
- Một số ứng dụng GPT và mô hình generative AI để xây dựng chatbot Q&A, hỗ trợ phân tích tài chính mà không cần chuyên môn sâu.
- Tiêu biểu, bài báo "Evaluation of Generative AI Q&A Chatbot Chained to Optical Character Recognition Models for Financial Documents"[5] là một nghiên cứu tương tự, tập trung vào việc tích hợp FSRCNN để nâng độ chính xác OCR lên hơn 93% và xây dựng hệ thống hỗ trợ người dùng mới trong phân tích tài liệu ngân hàng.

Nhìn chung, xu hướng là kết hợp nhận diện, phân loại và AI tương tác, tuy nhiên đa phần tập trung vào chuẩn quốc tế và ngôn ngữ phổ biến.

1.6.2 So sánh với dự án

- **Tương đồng:**
  - Cả hai đều cùng sử dụng OCR để trích xuất dữ liệu tài chính, chú trọng độ chính xác (FSRCNN trong bài báo, kết hợp sửa lỗi trong dự án).
  - Đều tích hợp AI tạo sinh vào chatbot Q&A hỗ trợ người dùng.
  - Cùng xây dựng pipeline xử lý tài liệu từ nhận diện đến phân tích nhằm tự động hóa quy trình.
- **Khác biệt và đổi mới:**
  - **Tùy chỉnh ngôn ngữ địa phương:** Dự án tối ưu cho tiếng Việt, xử lý từ khóa như “Tổng cộng”, “Tiền mặt”, giải quyết đặc thù dữ liệu phi chuẩn.
  - **Công nghệ lưu trữ và tìm kiếm:** Khác với mô hình classification của bài báo, dự án dùng embedding + Milvus để hỗ trợ tìm kiếm ngữ nghĩa, linh hoạt hơn.
  - **Ứng dụng thực tiễn:** Trong khi bài báo nhắm đến phân tích tài chính ngân hàng, dự án hướng đến lưu trữ hóa đơn, tích hợp chatbot, giao diện web thân thiện − phù hợp với doanh nghiệp nhỏ.

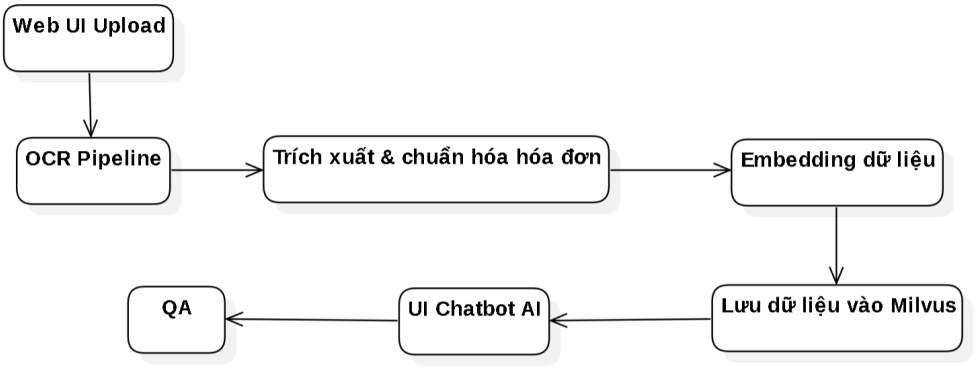
Chương 2

PHƯƠNG PHÁP TRIỂN KHAI VÀ THỰC NGHIỆM

2.1 Tổng quan kiến trúc hệ thống

2.1.1 Sơ đồ tổng thể hệ thống

Hệ thống được thiết kế theo mô hình hai thành phần chính: (1) khối xử lý OCR hóa đơn và (2) khối Chatbot trí tuệ nhân tạo hỏi đáp hóa đơn. Hai thành phần này tương tác với nhau thông qua một cơ sở dữ liệu vector (Milvus), đóng vai trò là cầu nối số hóa và lưu trữ thông tin hóa đơn dưới dạng vector embedding để phục vụ truy vấn ngữ nghĩa. Tổng thể, hệ thống hướng đến một quy trình tự động: từ việc tiếp nhận ảnh hóa đơn giấy, xử lý/văn bản hóa, đến hỏi đáp, truy xuất ngữ nghĩa dưới hình thức chat. Sơ đồ tổng thể hóa như sau:



Hình 2.1: Sơ đồ tổng thể hệ thống

2.1.2 Quy trình hoạt động tổng quát

Quy trình vận hành của hệ thống diễn ra theo các bước chính sau:

1. Tiếp nhận hóa đơn: Người dùng tải lên một hoặc nhiều ảnh chụp hóa đơn thông qua giao diện web.

2. Tiền xử lý và OCR hóa đơn: Ảnh sẽ được tiền xử lý bằng các thuật toán xử lý ảnh (chuẩn hóa sáng, loại nhiễu, nhị phân hóa), sau đó nhận dạng văn bản tiếng Việt bằng Tesseract OCR.

3. Sửa lỗi văn bản & Chuẩn hóa: Kết quả văn bản nhận dạng sẽ được đưa qua một mô hình sửa lỗi tiếng Việt, từ đó tăng độ chính xác để phân tích sau này.

4. Trích xuất thông tin cấu trúc: Sử dụng các kỹ thuật phân tích ngôn ngữ và mô hình ngôn ngữ lớn (LLM/Gemini), hệ thống rút trích các trường thông tin: tổng tiền, chiết khấu, sản phẩm, ngày tháng, v.v. Kết quả được đóng gói thành JSON chuẩn.

5. vector embedding: Thông tin hóa đơn dạng JSON sẽ được mã hóa thành các vector số bằng mô hình SentenceTransformer chuyên biệt cho tiếng Việt.

6. Lưu trữ vào Milvus: Toàn bộ embedding, cùng với thông tin nguyên bản, được lưu vào cơ sở dữ liệu vector Milvus. Điều này cho phép tìm kiếm hóa đơn theo ngữ nghĩa.

7. Truy vấn và hỏi đáp hóa đơn bằng chatbot AI: Chatbot sẽ dùng các thuật toán tìm kiếm vector trên Milvus để lấy ra những hóa đơn liên quan, phối hợp với LLM để trả lời các câu hỏi phức tạp của người dùng dưới dạng hội thoại tự nhiên.

8. Giao tiếp với người dùng: Mọi phản hồi, kết quả sẽ được hiển thị trực tiếp trên giao diện web, cung cấp trải nghiệm nhất quán từ tải hóa đơn đến hỏi đáp thông minh.

2.1.3 Công nghệ, nền tảng sử dụng

Tất cả các thành phần của hệ thống đều được hiện thực hóa trên nền tảng Python, với các bước triển khai chi tiết như sau:

- **Xử lý ảnh và OCR:**
  - Thư viện xử lý ảnh: OpenCV, Pillow (PIL).
  - Nhận dạng văn bản: Tesseract OCR thông qua thư viện pytesseract, hỗ trợ tiếng Việt.
  - Tiền xử lý hình ảnh: Các hàm Thresholding, Morphological operations, Gaussian blur do OpenCV cung cấp.

- **Xử lý ngôn ngữ tự nhiên, sửa lỗi OCR:**
  - Mô hình deep learning sửa lỗi tiếng Việt: Đóng gói và sử dụng qua pipeline của thư viện Hugging Face Transformers.
  - Phân tích và trích xuất thông tin: Ứng dụng mô hình ngôn ngữ lớn (LLM), triển khai thông qua Google Gemini API (genai sdk).
- **Embedding và lưu trữ vector:**
  - Embedding: "dangvantuan/vietnamese−document−embedding" là mô hình sử dụng SentenceTransformer, đạt hiệu quả cao khi biểu diễn ngữ nghĩa tiếng Việt.
  - Lưu trữ vector: Milvus − hệ quản trị cơ sở dữ liệu số cho phép index và tìm kiếm thực hiện trên hàng triệu vector embedding.
  - Thư viện tích hợp: langchain_huggingface cho embedding, còn truy vấn vector là langchain_milvus.
- **Giao tiếp, tương tác ứng dụng:**
  - Web backend: FastAPI (quản lý các API và giao diện upload hóa đơn), Streamlit (xây dựng giao diện chat AI động).
  - Template và quản lý giao diện: Jinja2 templates cho web upload.
  - JSON và quản lý file: Quản lý input/output với thư viện chuẩn Python.
  - Kết nối môi trường và bảo mật: dotenv để quản lý biến môi trường, Google Gemini, Hugging Face Token cho truy cập mô hình.

2.1.4 Kiến trúc triển khai gồm hai thành phần chính:

(a) **Thành phần OCR và quản lý dữ liệu hóa đơn**

- Xử lý đầu vào là ảnh hóa đơn đa định dạng.
- Tiền xử lý nâng cao chất lượng ảnh, tăng độ chính xác nhận diện ký tự.
- Sử dụng OCR engine (Tesseract) chuyên cho tiếng Việt.
- Áp dụng language model (transformers) để sửa lỗi và chuẩn hóa kết quả.
- Dùng LLM để trích xuất các trường thông tin tự động, tránh phụ thuộc vào template hóa đơn.
- Chuẩn hóa và lưu trữ bằng mô hình embedding giúp lưu lại ngữ nghĩa của hóa đơn.
- Lưu trữ database trên nền tảng Milvus, hỗ trợ tốt cho tìm kiếm ngữ nghĩa bằng vector.

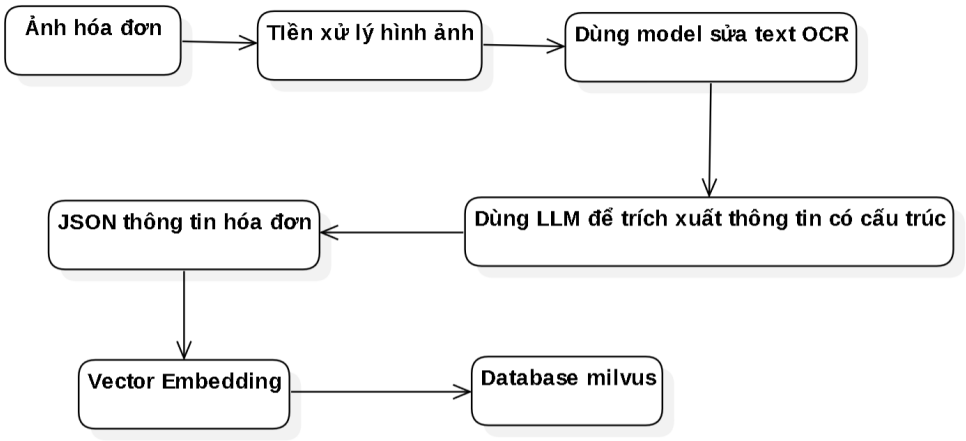
(b) **Thành phần Chatbot truy vấn thông minh**

- Hoạt động như một agent trung gian tích hợp nhiều công nghệ (retriever, LLM, tool).
- Hiểu ý định và truy xuất thông minh theo ngôn ngữ tự nhiên, không phụ thuộc từ khóa cứng.
- Hỗ trợ hội thoại đa lượt, ghi nhớ và sử dụng lại lịch sử trao đổi.
- Tổng hợp, diễn giải dữ liệu kết quả từ nhiều hóa đơn một cách tự nhiên cho người dùng.

Toàn bộ hệ thống có khả năng triển khai linh hoạt trên bất kỳ máy chủ/PC nào hỗ trợ Python và các thư viện liên quan, có thể tận dụng GPU để tăng tốc các tác vụ deep learning.

2.2 Hệ thống OCR và trích xuất thông tin

Đây là phần nền tảng của hệ thống, hiện thực hóa quá trình chuyển đổi ảnh hóa đơn (invoice image) sang cấu trúc thông tin số hóa, bao gồm nhiều tầng xử lý: tiền xử lý ảnh, nhận dạng văn bản, sửa lỗi, trích xuất trường thông tin, embedding ngữ nghĩa, và lưu trữ vector hóa đơn vào cơ sở dữ liệu ngữ nghĩa. Bản chất quy trình này kết hợp đa dạng các thuật toán trong xử lý ảnh, học máy và ngôn ngữ tự nhiên.



Hình 2.2: Quy trình OCR trích xuất thông tin

2.2.1 Tiền xử lý ảnh (Image Preprocessing)

Ảnh hóa đơn đầu vào thường đa dạng về màu sắc, ánh sáng, chất lượng camera,... Nếu đưa trực tiếp vào OCR sẽ làm giảm đáng kể độ chính xác của kết quả nhận diện. Do đó, hệ thống áp dụng các thuật toán tiền xử lý mục tiêu tăng contrast, giảm nhiễu và chuẩn hóa định dạng.

Các thuật toán và bước thực hiện:

2.2.1.1 Chỉnh lại kích thước (Resizing)

Đảm bảo kích thước ảnh đủ lớn (>300 DPI, tối thiểu 4 inch cho cạnh nhỏ nhất) để phân giải tốt nhất cho việc nhận dạng ký tự. Thuật toán sử dụng phép nội suy LANCZOS để phóng đại, giữ nguyên chi tiết nét chữ mà không bị vỡ hình.

2.2.1.2 Chuyển đổi màu sắc (Grayscale Conversion)

Ảnh được chuyển về đơn kênh mức xám nhằm loại bỏ thông tin màu, chỉ giữ lại cấu trúc sáng/tối − yếu tố quan trọng của văn bản.

2.2.1.3 Làm mờ nền bằng Gaussian Blur

ử dụng kernel lớn (ví dụ: 55x55 pixel), Gaussian Blur giúp ước lượng các biến đổi ánh sáng không đồng đều của ảnh nền (background illumination), loại bỏ nhiễu cũng như các vùng sáng tối không mong muốn.

2.2.1.4 Phẳng hóa (Flattening/Division Enhancement)

Áp dụng phép toán pixelwise chia ảnh gốc cho ảnh nền làm mờ (`cv2.divide(gra` `background, scale=255)`), từ đó "làm phẳng" sáng tối, làm nổi bật chữ so với nền.

2.2.1.5 Nhị phân hóa tự động (OTSU Thresholding)

Sử dụng phương pháp OTSU để tự động tìm ngưỡng phân chia tối ưu, chuyển ảnh về hai giá trị đentrắng, thúc đẩy mô hình OCR nhận diện nét chữ rõ ràng hơn.

2.2.1.6 Biến đổi hình thái học (Morphological Operations)

Sử dụng các phép biến đổi như đóng (closing) nhằm loại bỏ nhiễu nhỏ, đệm các ký tự bị đứt nét (do in/thời gian/scan), chuẩn hóa vùng biên văn bản.

Cơ sở toán học:

- **Phép chia chuẩn hóa:** $I_{\text{norm}} = \frac{I_{\text{gray}}}{I_{\text{blur}}} \times 255$
- **Ngưỡng Otsu:** Tự động tính threshold T, foreground/background được tối ưu phân tách
- **Hình thái học:** Khử nhiễu bằng phép đóng $A \bullet B = (A \oplus B) \ominus B$

```
def preprocess_pipeline(image: Image.Image) -> np.ndarray:
    resized_image = resize_image_in_memory(image)
    opencv_image = np.array(resized_image)
    if opencv_image.ndim == 3 and opencv_image.shape[2] == 3:
        opencv_image = cv2.cvtColor(opencv_image, cv2.
   COLOR_RGB2BGR)
    gray = cv2.cvtColor(opencv_image, cv2.COLOR_BGR2GRAY)
    background = cv2.GaussianBlur(gray, (55, 55), 0)
    flattened = cv2.divide(gray, background, scale=255)
    thresh = cv2.threshold(flattened, 0, 255, cv2.
   THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
    closed = auto_morphology(thresh)
    return closed
```

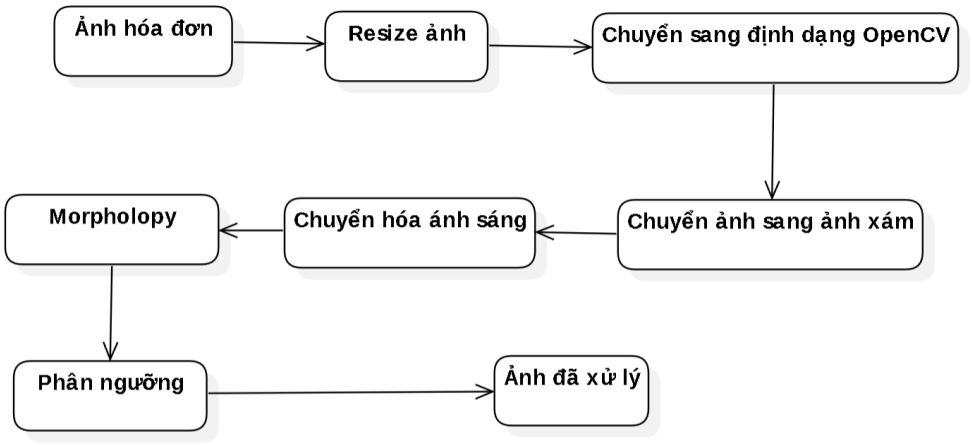
Đoạn mã 2.1: Hàm xử lý ảnh tiền xử lý trước OCR

Các thủ tục trên được hiện thực hóa thành pipeline chuẩn dưới dạng hàm preprocess_pipeline, cho phép truyền vào một ảnh PIL Image và đầu ra là mảng NumPy sẵn sàng cho nhận dạng ký tự

2.2.2 Quy trình cụ thể của tiền xử lý

Pipeline tiền xử lý triển khai tuần tự các bước sau:

1. **Đảm bảo kích thước & DPI:** Nếu ảnh nhỏ hơn ngưỡng yêu cầu, tiến hành resize để đạt tiêu chuẩn nhận dạng (`resize_image_in_memory`).
2. **Chuyển đổi màu, nâng cao sự tương phản chữ/nền:** Chuyển ảnh sang grayscale. Làm mờ nền rồi chia ảnh.
3. **Nhị phân hóa ảnh:** Áp dụng OTSU.
4. **Làm sạch nhiễu, làm đầy chữ bằng morphology.**
5. **Trả về ảnh đã tiền xử lý, đưa vào bước OCR.**



Hình 2.3: Quy trình tiền xử lý ảnh

2.2.3 Quy trình trích xuất văn bản bằng OCR

Sau khi đã qua pipeline tiền xử lý, hệ thống sử dụng engine Tesseract OCR cho bài toán nhận dạng ký tự quang học đối với tiếng Việt. Tesseract[6] là engine mã nguồn mở mạnh mẽ, được tối ưu cho nhiều ngôn ngữ và đặc biệt hiệu quả khi được cung cấp ảnh đã tiền xử lý. Cấu hình đặc biệt gọi Tesseract cho tiếng Việt:

- Thiết lập ngôn ngữ tiếng Việt `(lang='vie')` để tăng độ chính xác.
- Áp dụng thẳng lên ảnh nhị phân đã được làm sạch và chuẩn hóa.
- Kết quả là chuỗi văn bản thô, tuy nhiên thường thiếu dấu câu/thừa nhiễu do lỗi scan, watermark, chất lượng giấy.

```
def extract_text_from_image(image_path: str) -> str:
    img = Image.open(image_path)
    processed_img = preprocess_pipeline(img)
    text = pytesseract.image_to_string(processed_img, lang='vie')
    return text
```

Đoạn mã 2.2: Hàm trích xuất văn bản từ ảnh sau tiền xử lý

Hệ thống có khả năng xử lý hàng loạt ảnh hóa đơn đầu vào và trả về tất cả kết quả nhận dạng.

2.2.4 Sửa lỗi văn bản sau OCR

Bản chất Tiếng Việt có dấu và bảng mã Unicode phức tạp, vì vậy do ảnh hưởng của lỗi scan hoặc chất lượng font in/thủ công, việc "sạch hóa" văn bản thô (loại bỏ ký tự lạ, chuẩn hóa Unicode, loại văn bản dư, phát hiện lỗi nhận dạng điển hình) là bắt buộc trước các bước phân tích tiếp theo. Vì vậy sau khi nhận diện OCR – dù đã xử lý tốt ảnh – vẫn không thể tránh khỏi các lỗi ký tự do biến dạng ảnh, phông chữ lạ, hoặc nhiễu. Hệ thống sử dụng mô hình `sequencetosequence` chỉnh ngữ pháp, chính tả, tiền huấn luyện trên dữ liệu tiếng Việt thực tế. Mô hình này `(bmd1905/vietnamesecorrectionv2` từ Hugging Face) triển khai qua Transformers pipeline, có khả năng tự động phát hiện và chữa lỗi chính tả, từ bị nhầm lẫn, ngắt dòng sai hoặc xuất hiện ký tự lạ.

- Chia nhỏ văn bản thành các đoạn vừa phải (dưới giới hạn token).
- Dùng pipeline **transformers** với mô hình **seq2seq** để hiệu chỉnh từng đoạn.
- Ghép lại thành văn bản hoàn chỉnh sau sửa:

$$\text{output} = \arg\max_{y} \sum_{t=1}^{T} \log P(y_t \mid y_{<t}, x)$$

```
corrector = pipeline("text2text-generation",
                     model="bmd1905/vietnamese-correction-v2",
                     device=0 if torch.cuda.is_available() else -1)
```

Đoạn mã 2.3: Tạo pipeline chỉnh sửa lỗi chính tả tiếng Việt

2.2.5 Trích xuất thông tin có cấu trúc từ văn bản OCR

2.2.5.1. Phân tích ngữ nghĩa, nhận dạng thực thể

Khi đã có văn bản sạch, bài toán rút trích thông tin (IE Information Extraction) được đặt ra. Mục tiêu: chuyển đổi văn bản hóa đơn sang cấu trúc JSON với các trường: tổng tiền, thanh toán, sản phẩm, ngày, mã hóa đơn,... Áp dụng danh sách từ khóa đặc trưng cho từng loại trường (ví dụ: "Tổng cộng", "Giảm giá", "Tiền thối lại"...) kết hợp so khớp xấp xỉ để phòng trường hợp lỗi OCR kiểu "Tống cọng", "Giam gia". Ngoài ra, các quy tắc vàng (rulebased) được áp dụng nhờ chiến lược prompt engineering: Đưa các quy tắc so khớp tiêu đề, kiểm tra logic số liệu ("quy tắc vàng": quantity $\times$ unit_price $\approx$ total_price...) giúp đảm bảo độ tin cậy ban đầu khi mô hình gặp lỗi ngữ cảnh.

2.2.5.2. Xử lý thực tiễn lỗi OCR

Phần trích xuất chú trọng đến khả năng tự phục hồi khi gặp lỗi nhận diện, lỗi dấu hoặc lỗi ghép từ trong tiếng Việt, tăng độ linh hoạt nhờ phân tích ngữ cảnh lân cận của từ khóa.

2.2.5.3. Ứng dụng Large Language Model (LLM/Gemini)

Bởi sự phức tạp và đa dạng của hóa đơn thực tế, hệ thống ứng dụng thêm LLM (ví dụ: Gemini 2.5 của Google) để tạo cấu trúc hóa dữ liệu: từ văn bản phức tạp sinh ra JSON chuẩn, giải quyết bài toán đa biến thể ngôn ngữ, phát hiện lỗi logic số liệu hoặc ngữ nghĩa.

2.2.5.4 Quy tắc kiểm tra tính đúng đắn

Các trường hợp phát hiện bất thường (ví dụ tổng tiền âm, chiết khấu lớn hơn tiền hàng, giá trị không phù hợp quy tắc kế toán) sẽ được điền none hoặc báo lỗi cho người dùng.

```
prompt = f"""
Bạn là chuyên gia trích xuất hóa đơn. Trả về JSON với schema cố định:
{{
  "store_name": ...,
  "items": [{{"name":..., "quantity":...}}],
  ...
}}
QUY TẮC:
1. Chỉ trích xuất thông tin HIỆN DIỆN trong văn bản
2. Áp dụng logical cross-validation
3. KHÔNG tự tạo dữ liệu
=== VĂN BẢN ===
\"\"\"{text}\"\"\"
"""
```

Hình 2.4: Mẫu promt dùng cho LLM để sửa lỗi

2.2.6 Sinh vector embedding cho dữ liệu hóa đơn

2.2.6.1. Mô hình embedding tiếng Việt

Bước tiếp theo là chuyển hóa thông tin hóa đơn từ dạng văn bản cấu trúc sang vector số (embedding) để phục vụ truy vấn ngữ nghĩa. Hệ thống sử dụng mô hình `"dangvantuan/vietnamesedocumentembedding"` `(SentenceTransformer)`, đáp ứng tốt cho dữ liệu tiếng Việt, giữ lại ngữ nghĩa đoạn văn bản dài, chống tối giản thông tin.

```
model = SentenceTransformer('dangvantuan/vietnamese-document-
    embedding',
                                trust_remote_code=True)

def encode_texts(texts: list[str]) -> list[list[float]]:
    return model.encode(texts, batch_size=8).tolist()
```

Đoạn mã 2.4: Mã hóa văn bản bằng mô hình embedding tiếng Việt

2.2.6.2. Hàm toán học & đặc trưng embedding

Cho mỗi văn bản hóa đơn đầu vào s, mô hình trả về vector $v = f(s) \in \mathbb{R}^d$ với d là số chiều của embedding (ví dụ 768).
Hàm mã hóa f tối ưu cho ngữ cảnh tiếng Việt, đảm bảo các đặc điểm ngữ nghĩa gần nhau sinh ra vector gần nhau trong không gian Euclide.

2.2.7 Lưu trữ dữ liệu hóa đơn vào Milvus

2.2.7.1. Tổng quan Milvus & kiến trúc lưu trữ

Milvus [7] là cơ sở dữ liệu vector hiệu năng cao, hỗ trợ lưu trữ & truy vấn ngữ nghĩa ở quy mô lớn. Tại đây mỗi hóa đơn được lưu với trường chính:

- **id:** số nguyên tự động tăng
- **filename:** tên file hóa đơn
- **content:** dữ liệu JSON gốc
- **embedding:** vector ngữ nghĩa chiều d

2.2.7.2. Thiết lập schema, chèn dữ liệu, tạo chỉ mục

Quy trình:

- Khởi tạo schema (CollectionSchema), khai báo từng trường như trên.
- Tạo index cho trường embedding (ví dụ `IVF_SQ8, metric L2`).
- Lưu dữ liệu hóa đơn, embedding vào collection, flush để đảm bảo khả năng truy vấn ngay lập tức.
- Lấy id để tra cứu (mapping giữa ảnh, dữ liệu gốc, vector).

```
fields = [
    FieldSchema("id", DataType.INT64, is_primary=True, auto_id=True)
    ,FieldSchema("filename", DataType.VARCHAR, max_length=512),
    FieldSchema("content", DataType.VARCHAR, max_length=65_535),
    FieldSchema("embedding", DataType.FLOAT_VECTOR,
                dim=embed_model.get_embedding_dim())
]
```

Đoạn mã 2.5: Khai báo schema cho collection trong Milvus

Qua các bước trên, thành phần OCR đã hoàn toàn tự động hóa việc tiếp nhận ảnh hóa đơn, xử lý, nhận diện, lập cấu trúc dữ liệu và lưu trữ tối ưu cho các tác vụ tìm kiếm – đáp ứng hoàn hảo bài toán số hóa và quản trị thông tin hóa đơn trong môi trường thực tế.

2.2.8 Giao diện hệ thống trích xuất hóa đơn sử dụng FastAPI

Sau khi hoàn tất pipeline OCR và chuẩn hóa dữ liệu, hệ thống đóng gói các chức năng xử lý vào một API web xây dựng trên nền tảng FastAPI. FastAPI [8] là framework hiện đại cho phép thiết kế, phát triển các dịch vụ web API hiệu suất cao, phù hợp với các ứng dụng AI tích hợp, real-time. API này đứng vai trò trung gian: nhận file ảnh (upload), trả về kết quả trích xuất, đồng thời thực hiện lưu trữ vào database nếu người dùng lựa chọn. Luồng xử lý yêu cầu:

- **Tiếp nhận file ảnh hóa đơn và request từ người dùng:** Web UI frontend sử dụng HTML template Jinja2 gửi request tải ảnh lên API **/upload**. FastAPI nhận nhiều file một lúc, đặt lại tên để tránh trùng.

- **Điều phối thực thi pipeline trích xuất:** Sau khi nhận file, FastAPI tự động gọi hàm **process_receipt**, điều phối toàn bộ quy trình xử lý: tiền xử lý, nhận dạng, sửa lỗi, extract thông tin, sinh embedding.
- **Trả về kết quả xử lý cho người dùng:** Kết quả (thành công hay lỗi) sẽ được đưa vào HTML template trả ngược lại trình duyệt, dữ liệu được hiển thị cho người dùng kiểm tra (hoặc download file kết quả).
- **LLưu xuống cơ sở dữ liệu:** Nếu xác nhận, API còn hỗ trợ endpoint lưu hóa đơn xuống database vector Milvus.

2.2.9 Kết luận

Việc sử dụng FastAPI làm tầng giao tiếp cho hệ thống trích xuất hóa đơn giúp giải quyết triệt để bài toán "số hóa" quy trình nghiệp vụ hiện đại, đồng thời mở ra khả năng tích hợp, mở rộng hệ thống dễ dàng hơn, đảm bảo tính bảo trì, khả năng mở rộng và đáp ứng với các nhu cầu thực tế phát sinh.

2.3 Xây dựng Chatbot truy vấn hóa đơn

Giai đoạn này hiện thực bộ não "trợ lý ảo" AI, có khả năng tiếp nhận câu hỏi tự nhiên về hóa đơn từ người dùng, truy xuất dữ liệu ngữ nghĩa từ Milvus, sử dụng LLM để diễn giải, tổng hợp và sinh ra câu trả lời mạch lạc. Hệ thống kết nối đa tầng trên nền tảng Python với sự phối hợp giữa các thành phần đã số hóa ở phần trước.

2.3.1 Tổng quan kiến trúc Chatbot

Cốt lõi của hệ thống Chatbot là sự kết hợp giữa các thành phần sau:

- **Frontend:** Giao diện dựa trên Streamlit cho phép chat trực tuyến với người dùng qua trình duyệt web.
- **Backend:** Điều phối xử lý truy vấn, tích hợp Agent (LangChain), LLM, truy vấn Milvus.
- **Vector Store:** Tập hóa đơn đã embedding và lưu trữ trong Milvus, tối ưu hóa truy xuất theo ngữ nghĩa.

Luồng xử lý chính như sau:

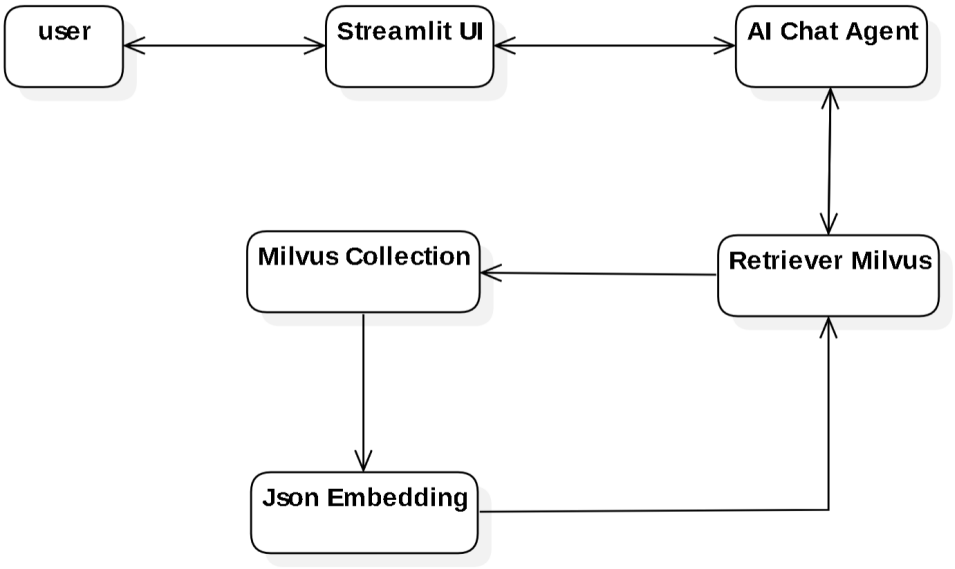
1. Người dùng nhập hoặc nói một câu hỏi bất kỳ (VD: "Tôi đã mua những mặt hàng gì trong hóa đơn ngày 12/6?").

2. Hệ thống xác định và tách các từ khóa chính, tạo vector embedding cho truy vấn.

3. Gửi truy vấn lên Milvus, tìm kiếm các hóa đơn có nội dung ("content embedding") gần nhất về mặt ngữ nghĩa với câu hỏi (vector search).

4. Kết quả các hóa đơn liên quan sẽ được cung cấp cho LLM (Agent) để tổng hợp, diễn giải thành câu trả lời cuối cùng.

5. Phản hồi hiển thị trực tiếp cho người dùng qua giao diện tương tác.



Hình 2.5: Quy trình Chatbot truy vấn thông tin hóa đơn

2.3.2 Truy xuất dữ liệu hóa đơn từ Milvus bằng vector search

Kết nối và tìm kiếm dữ liệu vector:

2.3.2.1 Kết nối và khởi tạo retriever

Hệ thống chịu trách nhiệm kết nối tới Milvus sử dụng hàm `get_milvus_retriever(collection_name)` để thiết lập retriever từ cơ sở dữ liệu vector Milvus. Retriever này dựa trên ngữ nghĩa, có khả năng trả về K mẫu hóa đơn liên quan đến embedding truy vấn của người dùng một cách hiệu quả.

```
def get_milvus_retriever(collection_name, db_name="default"):
    embedding_function = get_query_embedding_function()
    vector_store = Milvus(
        embedding_function=embedding_function,
        collection_name=collection_name,
        connection_args={"host": host, "port": port, "db_name":
   db_name},
        vector_field="embedding",
        text_field="content",
    )
```

```
    return vector_store.as_retriever(search_kwargs={'k': 3})
```

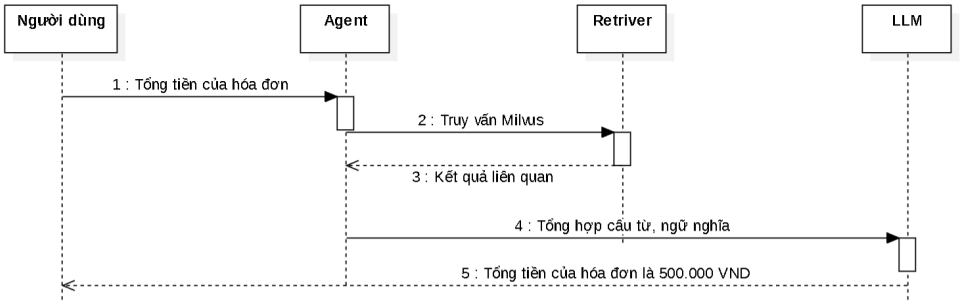
Đoạn mã 2.6: Khởi tạo retriever từ Milvus

2.3.2.2 Quy trình tìm kiếm:

Quy trình:

- Biểu diễn câu hỏi người dùng thành vector embedding (cùng mô hình với hóa đơn đã lưu trong Milvus).
- So sánh truy vấn với toàn bộ vector hóa đơn bằng khoảng cách L2 hoặc cosine (tùy chỉ mục).
- Trả về tập hóa đơn gần nhất để agent dùng vào reasoning, trả lời.

Ưu điểm của vector retrieval so với tìm kiếm từ khóa truyền thống là khai thác mạnh thông tin ngữ nghĩa, cho phép hệ thống trả lời ngay cả khi câu hỏi dùng các từ đồng nghĩa, diễn đạt phức tạp.



Hình 2.6: Quy trình tìm kiếm từ câu hỏi

2.3.2.3 Kỹ thuật đánh giá/so khớp:

Sử dụng độ đo khoảng cách vector `(L2/cosine)` để sắp xếp kết quả và chỉ lấy các hóa đơn có điểm tương đồng cao nhất.

$$\text{similarity} = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\|\|\vec{B}\|}$$

2.3.3 Kết hợp LLM & Agent để xử lý truy vấn tự nhiên

Đây là thành phần then chốt giúp chatbot không chỉ truy xuất dữ liệu mà còn hiểu, tổng hợp, diễn giải và trả lời như một trợ lý thực thụ.

2.3.3.1 Prompt Template và quản lý lịch sử trò chuyện

- Sử dụng kiến trúc "chat prompt template", giữ lịch sử trò chuyện (history) cho phép agent nắm được nội dung trước đó, bản chất hội thoại mạch lạc (VD: người dùng hỏi tiếp "Thế tổng tiền là bao nhiêu?" sau khi hỏi tên từng sản phẩm).
- Prompt sẽ kết hợp:
  - Hướng dẫn hệ thống (system prompt)
  - Lịch sử trao đổi (chat_history)
  - Nội dung truy vấn hiện tại (input)
  - Kết quả retrieved hóa đơn

  ...tạo điều kiện để LLM có đầy đủ ngữ cảnh để trả lời chính xác.

```
prompt = ChatPromptTemplate.from_messages([
     ("system", system_prompt),
     MessagesPlaceholder(variable_name="chat_history"),
     ("human", "{input}"),
     MessagesPlaceholder(variable_name="agent_scratchpad"),
])
agent = create_tool_calling_agent(llm, tools, prompt)
```

Đoạn mã 2.7: Tạo agent với prompt có ghi nhớ lịch sử và scratchpad

2.3.3.2 Tích hợp LLM trả lời đa ngữ cảnh

- Agent sử dụng các LLM như llama3, mistral, gemma (thiết lập chọn model động theo yêu cầu).
- LLM không chỉ đóng vai trò "hiểu" mà còn tổng hợp, lập luận dựa trên dữ liệu hóa đơn thực tế và bối cảnh trò chuyện; nhờ đó, bot có thể trả lời bằng tiếng Việt tự nhiên, logic, phản biện linh hoạt.

2.3.3.3 Thuật toán phối hợp retriever và Agent

- Sau khi retrieve K hóa đơn liên quan, agent sẽ "đọc hiểu" chúng bằng LLM, chọn lọc/trích rút thông tin đúng, tổng hợp lại, trả lại một đoạn hội thoại văn bản tự nhiên, thân thiện.
- Agent logic hiện thực trên nền langchain, thực hiện:
  1. Người dùng gửi input
  2. Nếu input là câu chào, trả lời nhanh ("Chào bạn, tôi có thể giúp gì...").
  3. Nếu là câu chuyên sâu, sẽ gọi retriever —> lấy hóa đơn liên quan —> gọi LLM để trả lời câu hỏi đã nắm dữ liệu thực tế, tạo hội thoại liên tục nhờ quản lý lịch sử.

2.3.4 Xây dựng giao diện người dùng

2.3.4.1. Streamlit: giao diện hội thoại

- Sử dụng Streamlit tạo UI động (web app), hỗ trợ realtime chat.
- Giao diện hỗ trợ: chọn collection hóa đơn, chọn model LLM, chat đa lượt.
- Các chức năng UI bổ trợ như hiển thị log reasoning của AI (xem quá trình suy nghĩ, giải thích của bot).

2.3.4.2 Quản lý trạng thái, phản hồi và log.

- Giao diện cho phép giữ nguyên trạng thái chat (tin nhắn, phiên, lựa chọn model).
- Trong mỗi lượt chat, có thể mở rộng để xem "quá trình suy nghĩ" (log reasoning) của agent, rất hữu ích cho việc kiểm thử, nâng cấp UI và audit câu trả lời.
- Đảm bảo khả năng phản hồi nhanh, uyển chuyển khi người dùng tiếp tục các truy vấn tiếp theo dựa trên ngữ cảnh vừa hỏi

2.3.5 Đánh giá tổng thể thành phần Chatbot

Thành phần chatbot vượt qua giới hạn của một công cụ tìm kiếm truyền thống nhờ tích hợp AI đa tầng: không cần người dùng nhớ từ khóa cứng, có thể hỏi ở nhiều ngữ cảnh, đối thoại liên tục, trả lời chi tiết và đưa ra giá trị cộng thêm trong quản lý hóa đơn. Hệ thống cũng dễ mở rộng cho các domain dữ liệu tương tự khác.

2.4 Kết quả thực nghiệm hệ thống OCR

2.4.1 Giới thiệu tổng quan hệ thống

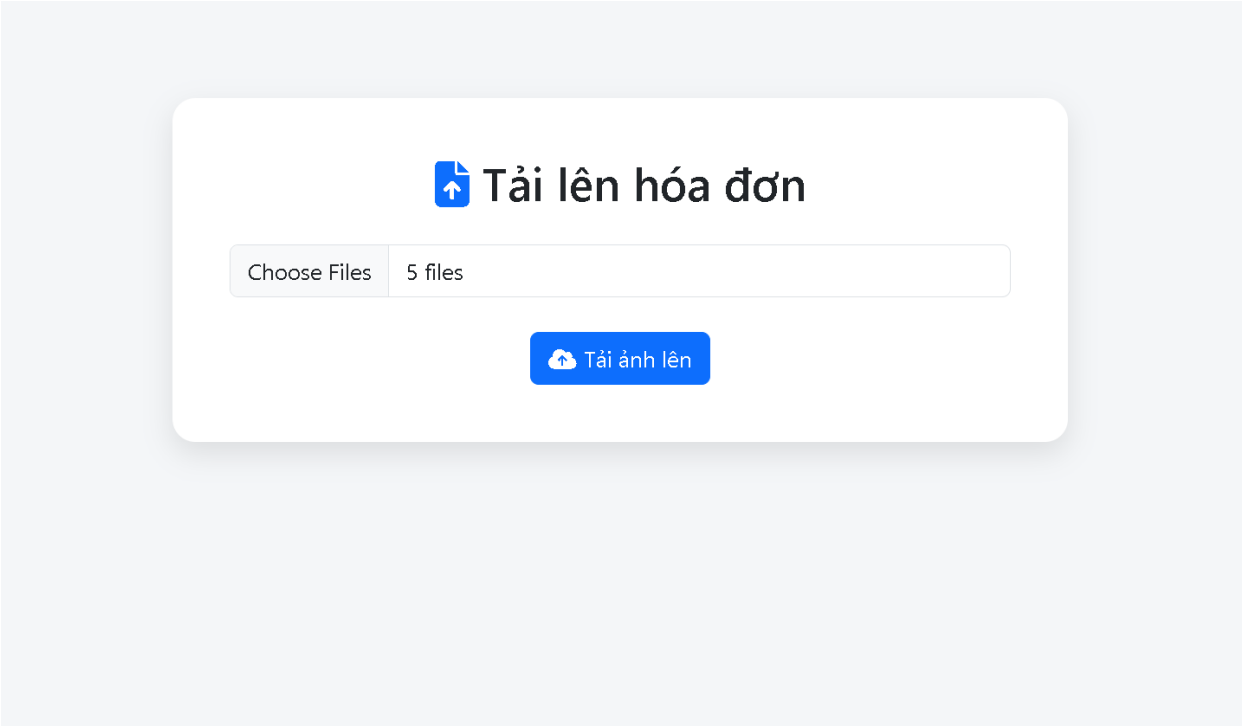
Hệ thống được xây dựng với mục tiêu tự động hóa quy trình trích xuất thông tin hóa đơn, phục vụ các doanh nghiệp vừa và nhỏ đến các chuỗi bán lẻ lớn nhằm tăng hiệu suất, giảm sai số thủ công và đồng bộ dữ liệu vào hệ thống quản trị. Các thành phần chính của hệ thống bao gồm:

- Giao diện upload ảnh hoá đơn (Webapp).
- Bộ xử lý tiền xử lý ảnh và nhận diện văn bản OCR.
- Giao diện web hiển thị, chỉnh sửa và xác nhận thông tin trích xuất.
- Chức năng lưu trữ, xuất dữ liệu (file JSON, kết nối database). Quản trị cơ sở dữ liệu vector (Milvus).

2.4.2 Quy trình hoạt động và mô tả các giao diện thực nghiệm

Giao diện upload và kiểm tra hoá đơn

Người dùng thao tác rất đơn giản: truy cập giao diện web, thao tác kéo – thả file hoặc chọn file ảnh hoá đơn cần nhận diện. Hệ thống lập tức hiển thị trạng thái tải ảnh, xử lý ảnh và chuyển sang trang trích xuất kết quả.



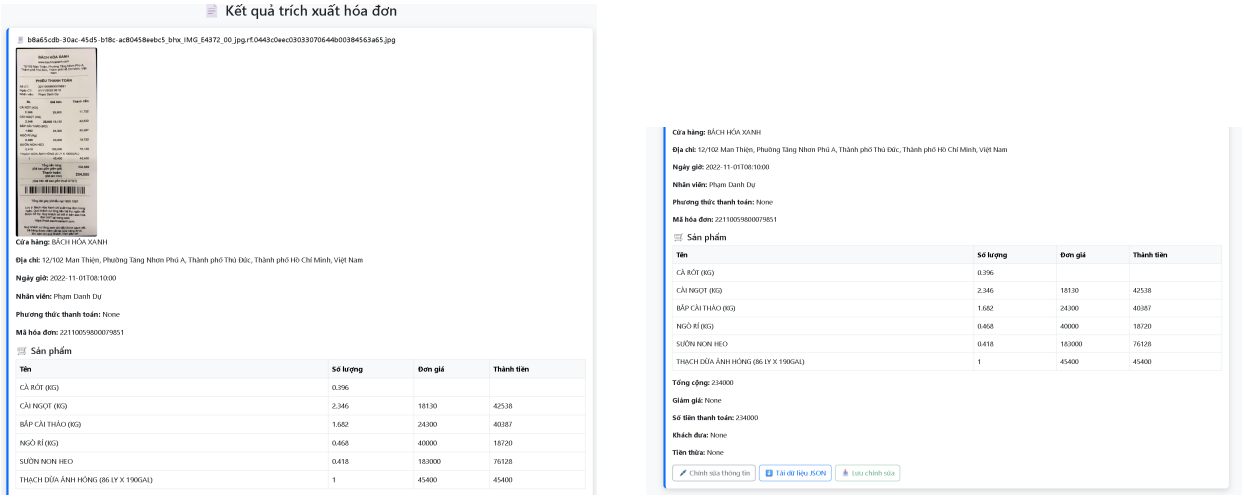
Hình 2.7: Giao diện upload hình ảnh

Giao diện trích xuất thông tin

Sau khi OCR, hệ thống hiển thị toàn bộ trường thông tin đã nhận dạng và phân tích được:

- Thông tin tổng quan: Tên cửa hàng, địa chỉ, ngày giờ giao dịch, nhân viên, phương thức thanh toán, mã hoá đơn...
- Bảng sản phẩm: thực đơn, tên sản phẩm, số lượng, đơn giá, thành tiền, tổng cộng, giảm giá, tiền khách đưa, tiền thừa.

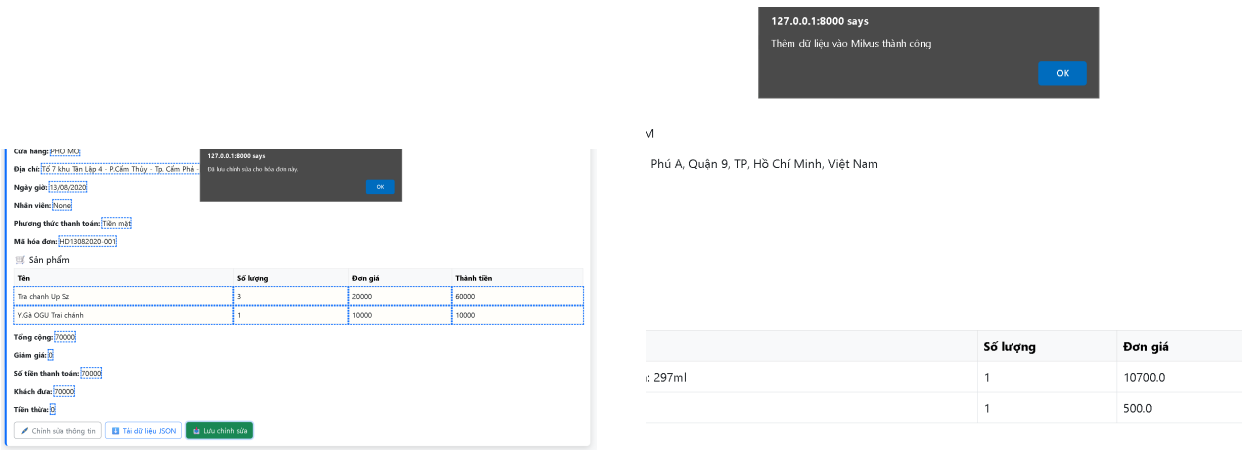
Toàn bộ trường thông tin này đều hiển thị rõ ràng, có kèm theo ảnh gốc có thể nhấn vào chỉnh sửa hoặc xác nhận lại nếu phát hiện sai sót. Người dùng có thể tải dữ liệu về dạng JSON hoặc lưu vào database, hệ thống sẽ hiện thông báo trạng thái.



Hình 2.8: Giao diện kết quả trích xuất thông

Thông báo trạng thái khi lưu/chỉnh sửa

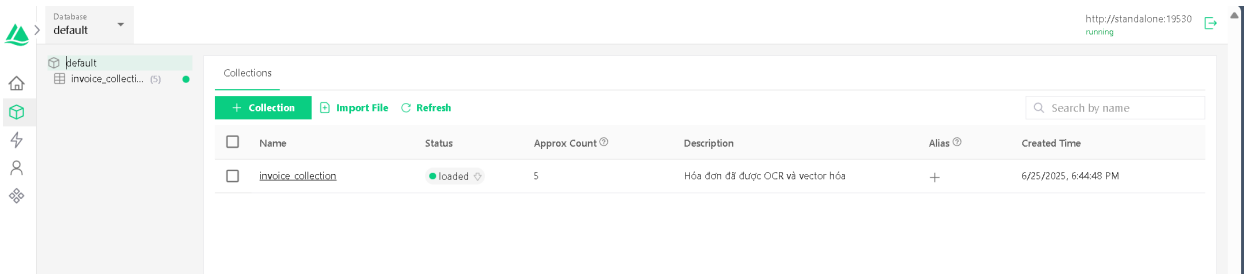
Một ưu điểm lớn của hệ thống là luôn hiển thị thông báo popup về các thao tác quan trọng (chỉnh sửa, lưu dữ liệu, cập nhật), giúp người dùng yên tâm quy trình đã hoàn tất, tránh nhập sót hay thao tác dư thừa.



Hình 2.9: Hiện thị thông báo

Quản lý lưu trữ và Milvus

Dữ liệu hoá đơn được lưu trực tiếp ở server Milvus dưới dạng vector – phù hợp nhu cầu tìm kiếm, truy vấn dữ liệu, số hóa tích hợp về sau.



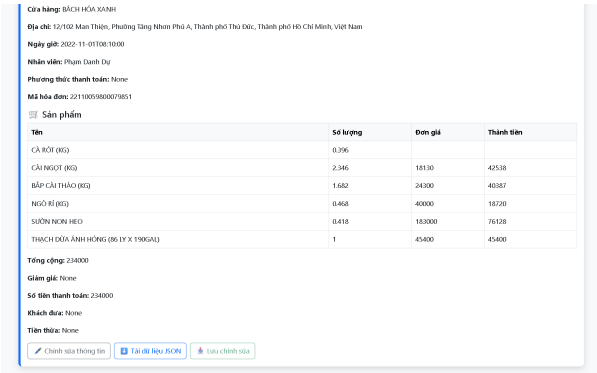
Hình 2.10: Giao diện server Milvus

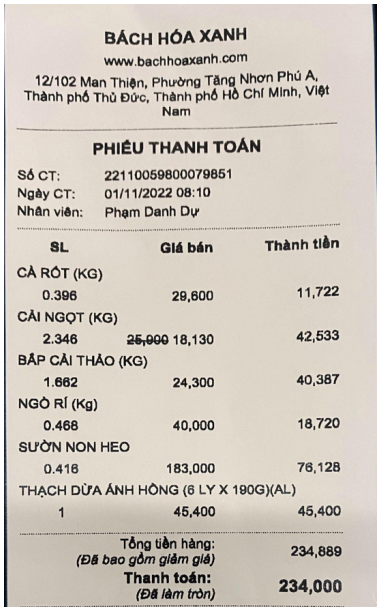
2.4.3 Kết quả thực nghiệm trên tập mẫu hoá đơn đa dạng

Hệ thống được đánh giá trên nhiều mẫu hoá đơn điển hình:

- Hóa đơn siêu thị (Bách Hóa Xanh, Saigon CO.OP Co.opFood...)
- Hóa đơn cửa hàng tiện lợi, tạp hoá
- Hóa đơn chuỗi quán cafe (The Coffee House)
- Hóa đơn quán ăn, nhà hàng
- Hóa đơn phí dịch vụ nhỏ lẻ (phí nước, phí phụ thu,...)

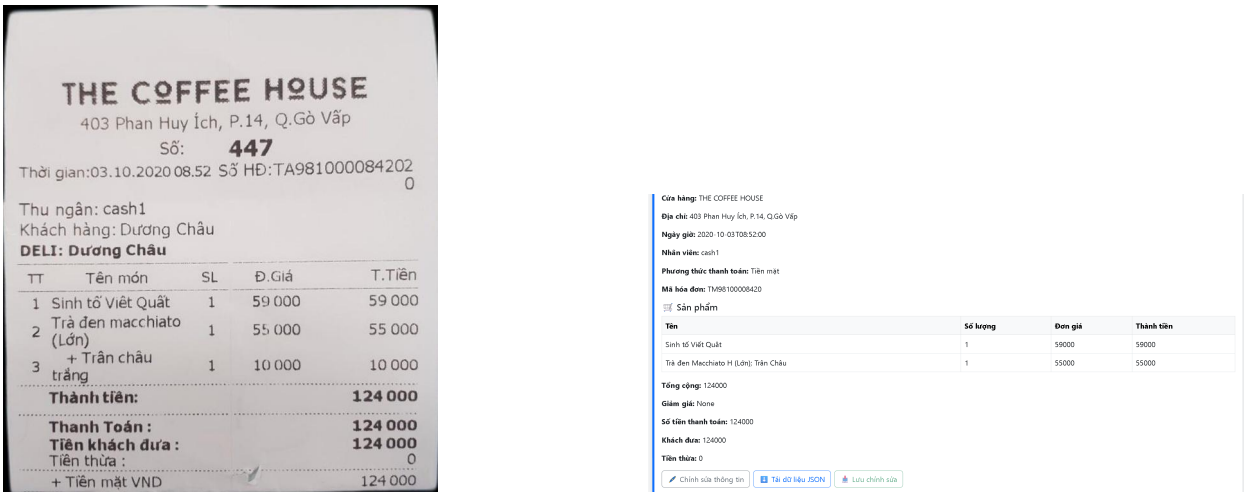
Ví dụ thực nghiệm 1: Hóa đơn Bách Hóa Xanh





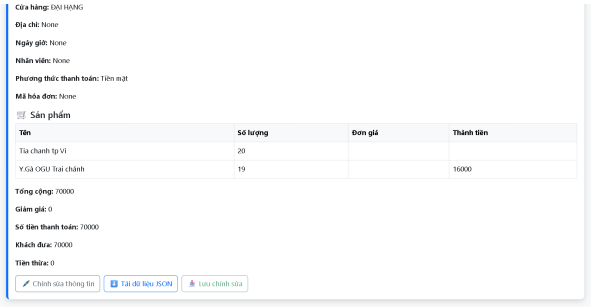
Hình 2.11: VD1 So sánh kết quả trích xuất với ảnh gốc

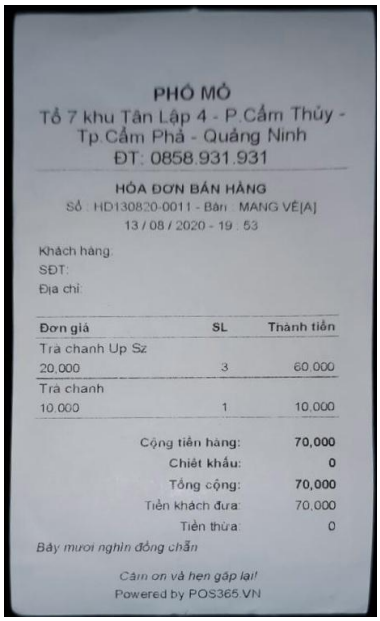
Ví dụ thực nghiệm 2: Hóa đơn The Coffe House



Hình 2.12: VD2 So sánh kết quả trích xuất với ảnh gốc

Ví dụ thực nghiệm 3: Hóa đơn PHO MO





Hình 2.13: VD3 So sánh kết quả trích xuất với ảnh gốc

2.4.4 Đánh giá hiệu quả nhận diện, phân tích kết quả đạt được

Độ chính xác trường thông tin

- Hệ thống nhận diện cực tốt các trường in đậm, trường cố định (cửa hàng, địa chỉ, ngày giờ, mã hóa đơn...), tỷ lệ đúng lên tới 80-90% với ảnh rõ nét chuẩn.
- Danh mục sản phẩm, số lượng, đơn giá, thành tiền nhận diện tốt, đúng cột, kể cả những trường hợp nhiều sản phẩm/ngày, không bị nhảy dòng.
- Dữ liệu đầu ra dạng bảng, phân trường rõ ràng, không sót/spam ký tự.

Đa dạng mẫu, linh hoạt bố cục

- Thử với hóa đơn có layout/phông/màu sắc/khổ giấy khác nhau đều cho kết quả tốt, không phụ thuộc template cứng.
- Nhận diện được các trường nâng cao: phí dịch vụ, thuế VAT, giảm giá, khuyến mãi, ghi chú bổ sung...

Tốc độ xử lý và thao tác người dùng

- Thời gian nhận diện 1 ộ hoá đơn trung bình chỉ mất 15-20 giây.
- Cho phép chỉnh sửa từng trường dữ liệu, giúp bổ sung/sửa lỗi thủ công.
- Hỗ trợ xuất dữ liệu cho các hệ thống khác (file JSON) dễ dàng.

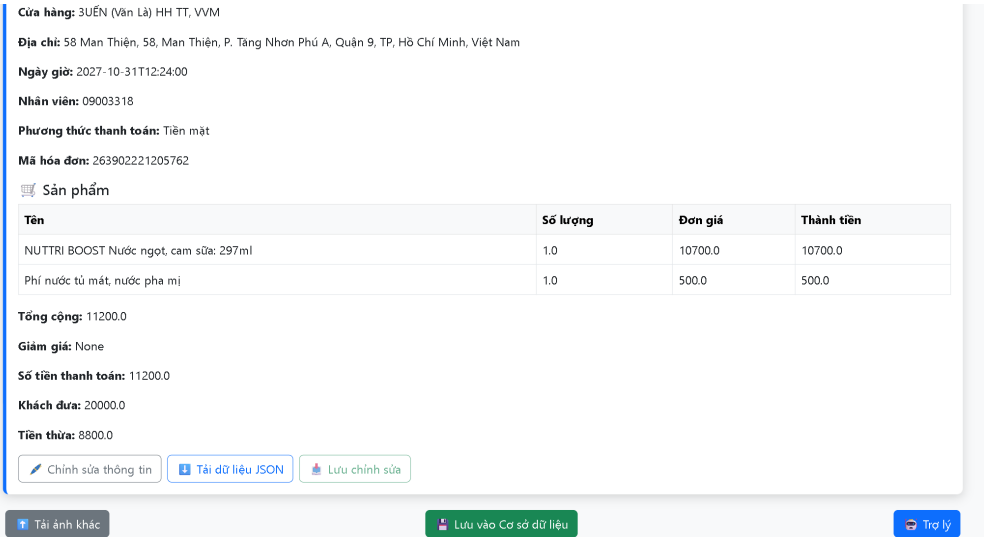
2.4.5 Điểm mạnh và giá trị nổi bật

- **Tự động hóa tối đa:** Người dùng chỉ cần upload là có dữ liệu trích xuất, tiết kiệm >90% thời gian nhập thủ công.
- **Thân thiện, minh bạch:** Giao diện rõ ràng, bao quát toàn bộ trường; thông báo realtime khi thao tác.
- **Linh hoạt, mở rộng:** Dễ tích hợp vào nhiều hệ quản trị, sẵn sàng phục vụ các hệ thống lớn.
- **Đa ngôn ngữ, đa lĩnh vực:** Nhận diện cả hóa đơn tiếng Việt, tiếng Anh, hóa đơn dịch vụ – ăn uống – bán lẻ – phí dịch vụ.

2.4.6 Hạn chế còn tồn tại & Đề xuất cải thiện

Nhận diện phụ thuộc ảnh đầu vào

Ảnh mờ, ảnh nghiêng, bóng sáng làm giảm tỷ lệ chính xác, xuất hiện lỗi nhầm số – nhầm chữ.



Hình 2.14: 3 chức năng trên thanh điều hướng

Bố cục quá đặc biệt vẫn còn lỗi

Một số hóa đơn dùng nhiều cột lạ, trường ghép/dán hoặc tiếng nước ngoài dễ bị nhận dạng lệch line.

Phát hiện trường thông tin thiếu/sai

Nếu hóa đơn không có đủ trường (ví dụ thiếu địa chỉ, thiếu nhân viên), hệ thống hiện "None" thay vì tự gợi ý/điền dựa trên ngữ cảnh.

Nhận diện tên sản phẩm, ghi chú đặc thù

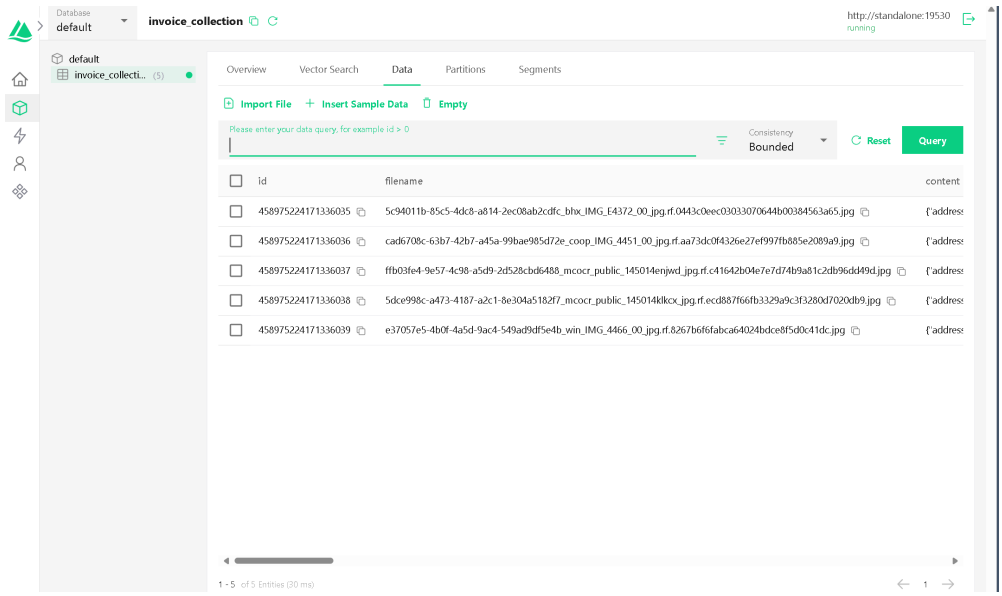
Tên sản phẩm có dấu câu, ký tự lạ dễ lỗi (vd: “Trà đen Macchiato H (Lớn)” hay “Phí nước tủ mát, nước pha mì” bị split từ...).

Trải nghiệm chỉnh sửa – thao tác nâng cao

Popup đôi khi hiện chưa trực quan, thiếu log lịch sử thao tác, thiếu cảnh báo lỗi trực tiếp trên từng trường. Đã có giải pháp là cho phép người dùng thủ công chỉnh sửa thông tin sai.

2.4.7 Tổng kết của hệ thống trích xuất OCR

Nhìn chung, kết quả thực nghiệm trên đa dạng mẫu hoá đơn cho thấy hệ thống OCR đã đáp ứng tốt mục tiêu tự động hóa nhập liệu, số hóa dữ liệu hóa đơn phục vụ quản trị hiện đại. Hệ thống nổi bật nhờ giao diện rõ ràng, dữ liệu trả về có cấu trúc đầy đủ, tích hợp lưu trữ linh hoạt, tốc độ nhanh, giúp tiết kiệm nhân lực đáng kể. Tuy nhiên, để hoàn thiện hơn, cần tập trung nâng cấp khâu tiền xử lý ảnh, xử lý đặc thù bố cục lạ, và tăng trải nghiệm người dùng ở khâu hiệu chỉnh lỗi.



2.5 Kết quả thực nghiệm của trợ lý ảo

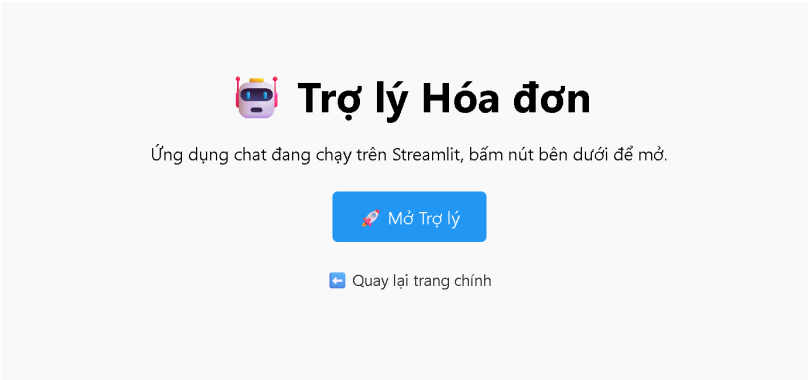
2.5.1 Giới thiệu tổng quan chức năng Chatbot

Bên cạnh khả năng trích xuất tự động dữ liệu hóa đơn bằng OCR, hệ thống còn tích hợp một mô-đun Chatbot (Trợ lý hóa đơn) sử dụng AI để hỗ trợ người dùng trong việc tra cứu, truy vấn, và khai thác thông tin hóa đơn mọi lúc, mọi nơi một cách tự động. Người dùng có thể hỏi chatbot những câu đơn giản, tự nhiên như hỏi số lượng hóa đơn vừa upload, liệt kê mã hóa đơn, truy chi tiết từng hóa đơn, v.v. Tính năng này giúp tiết kiệm thời gian, tăng tính chủ động và thân thiện của hệ thống.

2.5.2 Quy trình hoạt động và giao diện

Triển khai từ giao diện kết quả OCR

Khi người dùng thao tác trích xuất xong ở Tab kết quả trích xuất thông tin, chỉ cần nhấn nút "Trợ lý", hệ thống sẽ chuyển sang giao diện khởi tạo chatbot.



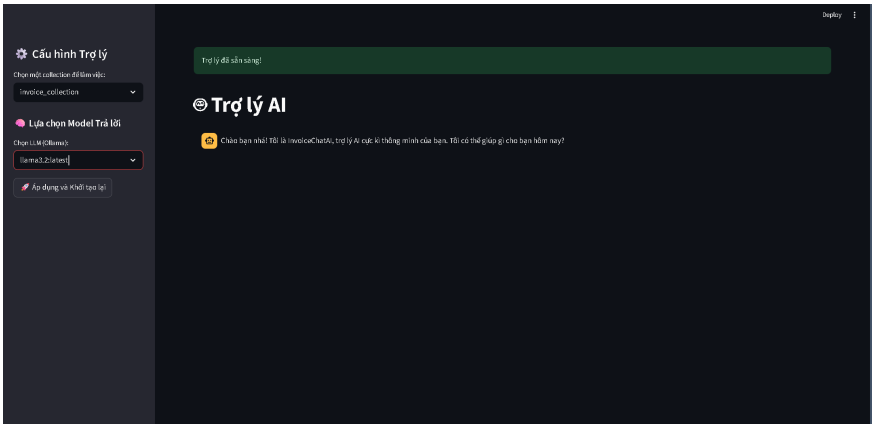
Hình 2.15: Giao diện mở chatbot

Giao diện này rất thân thiện, hướng dẫn dễ hiểu, giúp người dùng nhanh chóng tiếp cận mô-đun chatbot mà không cần kỹ năng công nghệ cao.

Giao diện chính của Chatbot – chuẩn bị sẵn sàng cho hội thoại

Sau khi nhấn nút "Mở trợ lý", chatbot được khởi tạo với hai khu vực:

- Bên trái là mục cấu hình, cho phép chọn tập dữ liệu (collection hóa đơn) và chọn model AI trả lời (ví dụ: llama3.2 hoặc các model khác tùy thiết lập).
- Bên phải là giao diện chat năng động, nơi người dùng đặt câu hỏi và nhận lại phản hồi từ chatbot.



Hình 2.16: Giao diện streamlit chatbot

2.5.3 Các tình huống thực nghiệm điển hình

Dưới đây là minh chứng thực tế các bước truy vấn – trả lời từ Chatbot AI hóa đơn:

a. Nhóm truy vấn hóa đơn – dữ liệu nội bộ Tình huống 1: Tra cứu số lượng hóa đơn vừa upload

- **Người dùng:** "Tôi vừa upload bao nhiêu hóa đơn?"
- **Chatbot trả lời:** "Hiện tại trên hệ thống đang có 5 hóa đơn vừa được upload. Bạn cần biết thêm thông tin hóa đơn nào?"

Hình 2.17: Câu hỏi 1

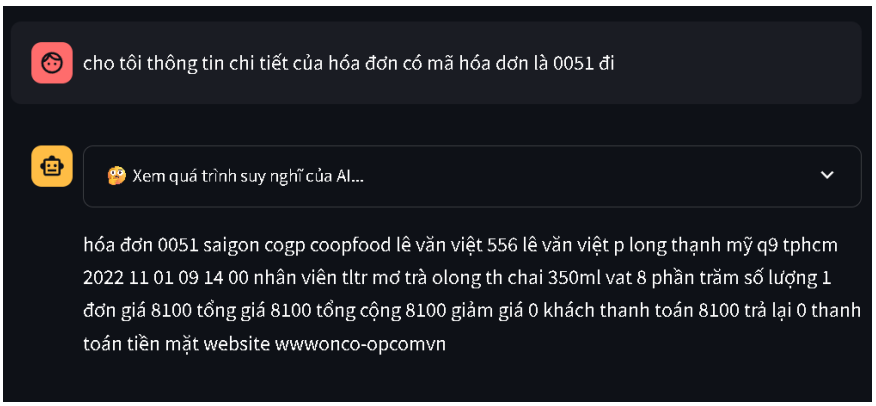
Tình huống 2: Liệt kê các mã hóa đơn đang có

- **Người dùng:** "Cho tôi biết các mã hóa đơn đang có đi."
- **Chatbot trả lời:** "Có 5 mã hóa đơn: 22110059800079851, 0051, TM98100008420, HD13082020-001, 263902221205762. Bạn muốn hỏi thông tin chi tiết hóa đơn nào không?"

Hình 2.18: Câu hỏi 2

Tình huống 3: Truy vấn chi tiết một hóa đơn cụ thể

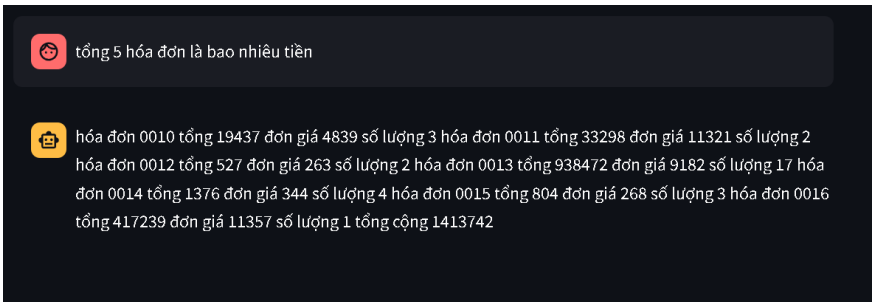
- **Người dùng:** "Cho tôi thông tin chi tiết của hóa đơn có mã hóa đơn là 0051 đi."
- **Chatbot trả lời:** "hóa đơn 0051 saigon cogp coop food lê văn việt 556 lê văn việt p long thạnh mỹ q9 tphcm 2022 11 01 09 14 00 nhân viên tltr mơ trà oolong th chai 350ml vat 8 phần trăm số lượng 1 đơn giá 8100 tổng giá 8100 tổng cộng 8100 giảm giá 0 khách thanh toán 8100 trả lại 0 thanh toán tiền mặt website wwwonco-opcomvn"



Hình 2.19: Câu hỏi 3

Tình huống 4: Tổng tiền của hóa đơn

- **Người dùng:** "Tổng 5 hóa đơn là bao nhiêu tiền?."
- **Chatbot trả lời:** "hóa đơn 0010 tổng 19437 đơn giá 4839 số lượng 3 hóa đơn 0011 tổng 33298 đơn giá 11321 số lượng 2 hóa đơn 0012 tổng 527 đơn giá 263 số lượng 2 hóa đơn 0013 tổng 938472 đơn giá 9182 số lượng 17 hóa đơn 0014 tổng 1376 đơn giá 344 số lượng 4 hóa đơn 0015 tổng 804 đơn giá 268 số lượng 3 hóa đơn 0016 tổng 417239 đơn giá 11357 số lượng 1 tổng cộng 1413742"



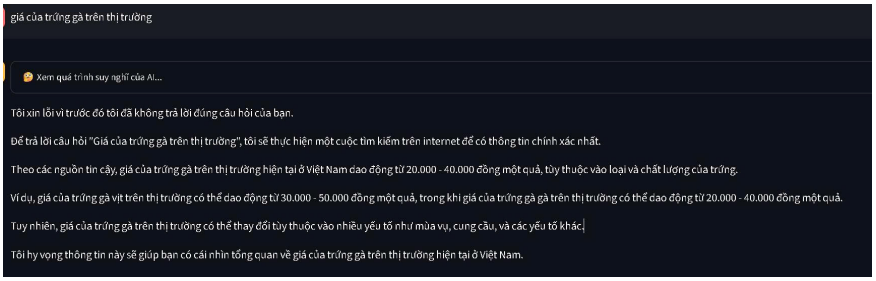
Hình 2.20: Câu hỏi 4

b. Nhóm truy vấn kiến thức mở rộng – dữ liệu ngoài hệ thống Thông tin thị trường, giá hàng hóa:
Tình huống 5: Tổng tiền của hóa đơn

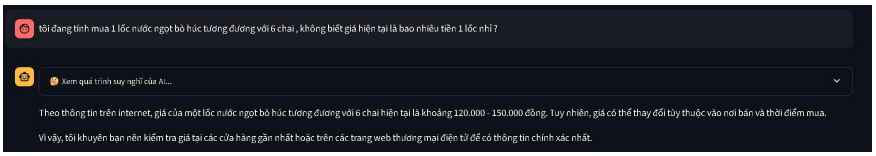
- **Người dùng:** "Giá của trứng gà trên thị trường", "giá của thịt heo mới nhất đang là bao nhiêu vậy."
- **Kết quả:** "Chatbot sử dụng truy vấn web, tra cứu thông tin ngoài hệ thống về giá sản phẩm thông dụng"
  - Có câu trả về mặc định, hơi chung chung (chi tiết giá trứng gà, trứng vịt, dao động tầm 20.000 −40.000 đồng/cái...);

– Có câu trả về đúng vùng, đúng loại thực phẩm, dẫn nguồn (Winmart, thị trường Việt Nam).

- Tuy nhiên, trả lời thường mang tính tham khảo, không gắn đích xác mã hóa đơn hoặc liên kết tới hóa đơn thực tế đã upload.



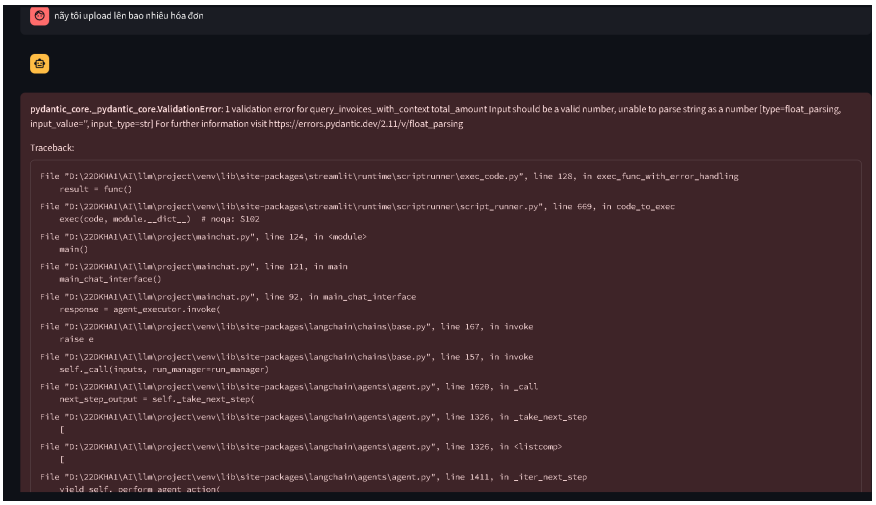
Hình 2.21: Câu hỏi 5



Hình 2.22: Câu hỏi 6

Truy vấn lại lịch sử kém, không nhận diện được:

- Khi hỏi điều kiện ngược như "nãy tôi upload lên bao nhiêu hóa đơn" với môi trường context bị lỗi, hệ thống trả về exception (traceback log Python), màn hình đỏ, mặc dù đã hỏi trước đó rồi.



Hình 2.23: Thử hỏi lại câu đã hỏi

2.5.4 Đánh giá tổng hợp điểm mạnh

- **Khả năng trả lời truy vấn về dữ liệu hóa đơn lưu trữ cao:** Các lượt hỏi về mã, số lượng, tổng thành tiền – nếu truy vấn rõ ràng, ngắn gọn – đều trả về rất nhanh, đúng dữ kiện, phù hợp ngữ cảnh.
- **Tích hợp tìm kiếm thông tin ngoài hệ thống:** Chatbot có khả năng tìm kiếm, trả lời các câu hỏi liên quan hàng hóa trên thị trường như giá bán, thông tin vật tư... hỗ trợ mở rộng kiến thức cho người dùng thực tế.
- **Tính dẫn dắt hội thoại tốt:** Khi trả lời, chatbot luôn có gợi ý tiếp (ví dụ hỏi "bạn muốn biết hóa đơn nào?", "bạn cần chi tiết mã hóa đơn nào không?"...), tạo cảm giác gần gũi, tiết kiệm thao tác lựa chọn thủ công.
- **Thời gian đáp ứng không nhanh:** Với các truy vấn trên tập dữ liệu vừa upload, độ trễ đa số dưới 10–15 giây vì model phải suy luận nên hơi lâu, với lại tùy cấu hình máy tính đang chạy, có xuất hiện delay khi xử lý câu hỏi.
- **Hỗ trợ truy xuất trên web – đa kênh kiến thức:** Vượt ngoài câu hỏi chỉ về hóa đơn, giúp người dùng tận dụng AI cho tra cứu thông tin kinh doanh/ngành hàng.

2.5.5 Các vấn đề, hạn chế còn tồn tại

- **Lỗi truy vấn context hoặc truy vấn nhiều lần một câu dễ lỗi hệ thống:** Trường hợp người dùng hỏi lại cùng nội dung, đổi cấu trúc, có thể gây lỗi chuyển đổi kiểu dữ liệu (validation, parsing error), trả về màn hình log hệ thống mất thân thiện, nguy cơ crash tạm thời ứng dụng.
- **Truy vấn phức tạp, đạo hàm ngoài phạm vi hóa đơn dễ ra kết quả chung chung hoặc không sát dữ liệu nội bộ:** Khi hỏi giá trứng/thịt hay sản phẩm khác (không có trong hóa đơn hệ thống), AI chỉ trả về kết quả web, chưa kết nối để truy vết dữ liệu thực trên hệ thống.
- **Một vài truy vấn tổng hợp trả về bị chồng chữ, định dạng chưa tối ưu:** Các truy vấn như tổng tiền nhiều hóa đơn, so sánh giá/số lượng đôi khi bị lỗi hiển thị – chồng chữ, thiếu bảng hoặc dấu phân cách, gây khó đọc và tiếp nhận.
- **Mức độ cá nhân hóa chưa cao:** Hệ thống chưa nhận biết trạng thái từng người dùng, chưa lọc truy vấn theo user/session – đặc biệt trong trường hợp nhiều người dùng cùng lúc.

2.5.6 Đề xuất hoàn thiện và phát triển

- **Nâng cao xử lý lỗi và ngoại lệ:** Cần bổ sung chặn lỗi hệ thống, đưa ra thông báo thân thiện (ví dụ: "Xin lỗi, hệ thống đang xử lý – vui lòng thử lại...") thay vì hiển thị log error hoặc traceback gây mất thiện cảm.
- **Chuẩn hóa định dạng kết quả:** Đáp án cho các truy vấn về hóa đơn, tổng tiền, nội dung chi tiết nên được trình bày bằng bảng hoặc làm nổi bật các trường chính để dễ đọc và thuận tiện cho việc xuất báo cáo.
- **Tăng khả năng liên kết giữa kiến thức ngoài và dữ liệu thực:** Khi truy vấn về sản phẩm ngoài hóa đơn, hệ thống nên trả lời bằng cách tham chiếu tới các dữ kiện thực tế đã lưu, kết hợp giữa kết quả từ AI và dữ liệu nội bộ để tăng độ tin cậy.
- **Cải thiện nhận diện truy vấn đa dạng:** Cần bổ sung thêm các kịch bản hội thoại nhằm nhận diện tốt hơn các cách hỏi khác nhau của người dùng, bao gồm từ đồng nghĩa, câu dài/ngắn, hoặc câu hỏi không đầy đủ.

2.6 Kết luận chung cho cả dự án

- Hệ thống kết hợp OCR và AI chatbot mang đến giải pháp số hóa hóa đơn hiện đại, giúp tự động trích xuất, quản lý và truy xuất dữ liệu từ ảnh hóa đơn.
- Người dùng có thể dễ dàng tải lên ảnh, bóc tách thông tin và tra cứu hóa đơn thông qua giao diện web và hội thoại tự nhiên với chatbot.
- Hệ thống hoạt động ổn định, nhận diện chính xác các trường thông tin như cửa hàng, sản phẩm, tổng tiền, ngày giờ,... trong hầu hết các hóa đơn rõ nét.
- Chatbot hỗ trợ tìm kiếm, kiểm tra và tổng hợp dữ liệu nhanh chóng, phù hợp với người dùng không quen thao tác thủ công hoặc xử lý số lượng lớn.
- Một số hạn chế còn tồn tại như phụ thuộc vào chất lượng ảnh đầu vào và phản hồi chatbot đôi lúc chưa tối ưu hoặc khó trích xuất ra báo cáo.
- Tổng thể, hệ thống góp phần giảm tải công việc nhập liệu, tăng tính tự động và mở rộng tiềm năng chuyển đổi số trong quản lý hóa đơn.

Chương 3

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

3.1 Kết luận

Trong thời đại số hóa và chuyển đổi số mạnh mẽ hiện nay, việc tự động hóa các tác vụ liên quan đến xử lý văn bản và chứng từ đang là một nhu cầu thiết yếu đối với các doanh nghiệp ở nhiều lĩnh vực. Một trong những dạng chứng từ phổ biến chính là hóa đơn, bao gồm thông tin chi tiết về tổng số tiền cần thanh toán và chi tiết về sản phẩm như số lượng, tên sản phẩm và tổng số tiền sản phẩm đó. Trước đây, quá trình lưu trữ và tra cứu hóa đơn thường được thực hiện thủ công hoặc lưu trữ dưới dạng hình ảnh, doc và PDF, gây nhiều khó khăn trong việc tìm kiếm, truy xuất và phân tích dữ liệu. Đề tài "Hệ Thống Nhận Diện và Lưu Trữ Thông Tin Hóa Đơn" ra đời nhằm giải quyết bài toán nói trên bằng việc áp dụng các kỹ thuật hiện đại trong xử lý ảnh, nhận diện ký tự quang học (OCR), lưu trữ dữ liệu có cấu trúc và truy vấn bằng phương pháp tìm kiếm ngữ nghĩa.

Hệ thống được xây dựng với kiến trúc chia lớp rõ ràng, vận hành thông qua một ứng dụng web phát triển dựa trên framework FastAPI. Người dùng tương tác thông qua một giao diện đơn giản, dễ dùng, có khả năng tải lên một hoặc nhiều hình ảnh hóa đơn. Sau khi được cung cấp hình ảnh về hóa đơn, hệ thống sẽ thực hiện xử lý tiền xử lý hình ảnh bao gồm các bước như chuyển ảnh sang thang độ xám, tăng cường độ tương phản, loại bỏ nhiễu, chuẩn hóa DPI và thay đổi kích thước ảnh (nếu cần thiết). Việc này nhằm mục tiêu chính là nâng cao chất lượng ảnh đầu vào cho quá trình OCR sau đó.

Sau khâu xử lý hình ảnh, hệ thống áp dụng `pytesseract` – thư viện phổ biến trong nhận diện ký tự – để chuyển đổi các ký tự trên ảnh hóa đơn thành văn bản số. Phần nội dung này được đóng gói dưới dạng JSON và lưu trữ đồng thời dưới 2 dạng: nội dung thô (text) và vector embedding. Đặc biệt, hệ thống

còn tích hợp mô hình mã hóa văn bản (embedding model), giúp chuyển văn bản thành vector số có kích thước cố định phục vụ cho việc tìm kiếm và so sánh theo ngữ nghĩa.

Một ưu điểm của hệ thống là việc sử dụng cơ sở dữ liệu vector `Milvus` – một hệ quản trị cơ sở dữ liệu hiện đại hỗ trợ tìm kiếm vector tốc độ cao. Trong quá trình nhập dữ liệu, mỗi hóa đơn sẽ được lưu thông tin gồm tên file, nội dung văn bản gốc và vector embedding sinh ra từ nội dung đó. Khi cần tìm kiếm hoặc truy vấn, hệ thống có thể so sánh các văn bản trên cơ sở ngữ nghĩa thay vì chỉ dựa vào so khớp từ khóa thuần túy. Điều này giúp tăng cường khả năng tìm kiếm gần đúng, tìm hóa đơn theo nội dung tương tự và mở rộng ứng dụng phân tích dữ liệu trong tương lai.

Một số tiện ích khác cũng được triển khai như: giao diện `Streamlit` giúp tương tác người dùng theo hướng thân thiện và khai thác dữ liệu đã được nhập, hiển thị kết quả OCR một cách tập trung, tải về thông tin nếu cần thiết,... Trong phần backend, hệ thống sử dụng các chuẩn kỹ thuật hiện đại như `UUID` cho tên file, môi trường ảo hóa có cấu hình linh hoạt để đảm bảo tính mở rộng và triển khai thực tế (cloud hoặc on-premises).

Có thể nói, hệ thống đã hoàn thành tốt các yêu cầu ban đầu đề ra, bao gồm: hỗ trợ nhiều ảnh đầu vào, trích xuất chính xác thông tin từ hóa đơn, lưu trữ hiệu quả dưới dạng có cấu trúc, tích hợp phương pháp tìm kiếm hiện đại với vector, và hỗ trợ giao diện người dùng trực quan. Đề tài mang đến một khuôn mẫu ứng dụng trong thực tế, không chỉ áp dụng được cho hóa đơn mà còn có thể mở rộng sang các loại tài liệu khác như biên bản, phiếu thu, hợp đồng,...

3.2 Hướng phát triển

Mặc dù hệ thống hiện tại đã hoàn thiện một chu trình nhận diện – xử lý – lưu trữ thông tin hóa đơn, tuy nhiên trong môi trường doanh nghiệp thực tế, nhu cầu và khối lượng dữ liệu có thể tăng lên rất nhiều, đòi hỏi hệ thống cần được mở rộng về cả chiều sâu (độ thông minh) lẫn chiều rộng (khả năng tích hợp). Do đó, dưới đây là một số định hướng phát triển quan trọng nhằm tối ưu và nâng cao toàn diện hiệu năng cũng như tính ứng dụng của hệ thống.

1. **Nâng cấp mô hình OCR**
   Hiện tại hệ thống sử dụng `pytesseract`, vốn là gói thư viện truyền thống với độ ổn định cao nhưng chưa thực sự tối ưu với các loại hóa đơn in mờ, bố cục phức tạp hoặc có phông chữ lạ. Việc thay thế bằng các mô hình Deep Learning tự xây dựng sẽ giúp model có chiều sâu trong việc xử lý hóa đơn như CNN, AlexNet, VGG, ResNet,... Các mô hình này có khả

năng nhận diện theo dòng, theo vùng chữ hoặc theo layout hóa đơn, từ đó thích nghi với đa dạng thực tế hơn.

2. **Tích hợp tìm kiếm thông minh bằng AI**
   Hướng phát triển tiềm năng là thêm khả năng truy vấn ngữ nghĩa các hóa đơn, xây dựng trợ lý thông minh về hóa đơn – cho phép đối thoại tương tác người – máy để hỏi/đáp trực tiếp các câu hỏi về dữ liệu hóa đơn đã lưu. Ứng dụng như vậy phục vụ tốt cho phòng kế toán, kiểm toán hoặc bộ phận pháp lý nội bộ trong doanh nghiệp.

3. **Chuẩn hóa và phân loại dữ liệu**
   Trong thực tế, cùng một nhà cung cấp có thể có nhiều mẫu hóa đơn khác nhau. Việc hệ thống hóa các trường thông tin chính như ngày tháng, tổng tiền, VAT, mã số thuế,... theo chuẩn cố định sẽ rất hữu ích để xuất dữ liệu ra các hệ thống lớn hơn (ERP, kế toán doanh nghiệp,...). Ngoài ra, có thể áp dụng thêm AI hoặc logic nhận dạng mẫu (template matching) để tự động phân loại loại hóa đơn (giấy, điện tử, siêu thị, logistics,...).

4. **Mở rộng triển khai ứng dụng**
   Hiện tại hệ thống hoạt động tốt trên local máy chủ. Trong giai đoạn tiếp theo, cần triển khai hệ thống lên các nền tảng cloud như AWS Sagemaker, Google Cloud Run, hoặc Azure App Service, đồng thời tích hợp các cơ sở lưu trữ cloud (S3, Firestore, Mongo Atlas,...) để phục vụ người dùng thực tế. Song song đó là phát triển API chuẩn RESTful hoặc GraphQL để tích hợp vào các phần mềm khác.

5. **Tăng cường bảo mật và phân quyền truy cập**
   Bảo mật là yếu tố không thể bỏ qua, đặc biệt khi làm việc với thông tin nhạy cảm như hóa đơn, tài chính. Hệ thống cần bổ sung cơ chế đăng nhập, xác thực người dùng (JWT, OAuth2), phân quyền theo vai trò (admin – user – accountant), ghi nhật ký truy cập và mã hóa nội dung quan trọng trong quá trình lưu trữ hoặc truyền tải.

Tài liệu tham khảo

[1] Yasser Alginahi **andothers**. "Preprocessing techniques in character recognition". **in***Character recognition*: 1 (2010), **pages** 1–19.

[2] *SuNT's Blog | AI in Practical*. URL: https://tiensu.github.io/blog/63_ocr_introduction/.

[3] *Word Embedding - Tìm hiểu khái niệm cơ bản trong NLP*. URL: https://viblo.asia/p/word-embedding-tim-hieu-khai-niem-co-ban-trong-nlp-1Je5E93G5nL.

[4] Vladimir Lyashenko. *Cross-Validation in Machine Learning: How to Do It Right*. neptune.ai. URL: https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right.

[5] Yu Qiu **andothers**. "Evaluation of Generative AI QA Chatbot Chained to Optical Character Recognition Models for Financial Documents". **in**URL: https://doi.org/10.1145/3647750.3647766.

[6] *Tesseract OCR: What Is It and Why Would You Choose It?* URL: https://www.klippa.com/en/blog/information/tesseract-ocr/.

[7] *Vector Database với Milvus - Viblo*. URL: https://viblo.asia/p/vector-database-voi-milvus-Rk74a5Kk4e0.

[8] *FastAPI*. URL: https://fastapi.tiangolo.com/.