

# CartMate: Chatbot Tư Vấn Thời Trang Tích Hợp Công Nghệ RAG

**Đinh Thiên Ân**

University of Information Technology  
22520010@gm.uit.edu.vn

**Huỳnh Trọng Nghĩa**

University of Information Technology  
22520003@gm.uit.edu.vn

**Lê Trần Gia Bảo**

University of Information Technology  
22520105@gm.uit.edu.vn

## Abstract

Đây là báo cáo đồ án môn CS311.P11, track NLP do thầy Đặng Văn Thìn hướng dẫn. Chủ đề của nhóm chúng em là làm về chatbot. Hệ thống chatbot được xây dựng dựa trên mô hình RAG (Retrieval-Augmented Generation), kết hợp với các phương pháp tiên tiến như Semantic Router và Reflection để tối ưu hóa hiệu suất và độ chính xác. Hệ thống sử dụng Hybrid Search (kết hợp Semantic Search và Keyword Search) để truy xuất thông tin từ cơ sở dữ liệu MongoDB, sau đó tạo câu trả lời tự nhiên bằng mô hình ngôn ngữ lớn Gemini-1.5-flash.

## 1 Introduction

Trong bối cảnh thương mại điện tử (TMĐT) phát triển mạnh mẽ, việc tìm kiếm và mua sắm sản phẩm trực tuyến trở nên phổ biến. Tuy nhiên, người dùng thường gặp khó khăn trong việc tìm kiếm sản phẩm phù hợp do lượng thông tin lớn và đa dạng. Để giải quyết vấn đề này, chúng em xây dựng một chatbot (Vakayil et al., 2024) thông minh dựa trên mô hình **RAG (Retrieval-Augmented Generation)** (Lewis et al., 2020).

### 1.1 Các chức năng chính của Chatbot

#### • Tìm kiếm và gợi ý sản phẩm:

- Chatbot có khả năng tìm kiếm sản phẩm dựa trên yêu cầu của người dùng (ví dụ: tên sản phẩm, màu sắc, kích thước) và đưa ra các gợi ý phù hợp.
- Ví dụ: "Tôi cần tìm một đôi giày trắng, có mẫu nào không?", "Có đồ cho trẻ em không?", "Áo sơ mi nữ bên shop giá như thế nào?"

#### • Tư vấn sản phẩm dựa trên nhu cầu cá nhân:

- Chatbot trong vai trò là tư vấn viên đưa ra các sản phẩm và tư vấn dựa theo nhu cầu của người dùng. Kể cả các câu hỏi phức tạp, nhiều thành phần: size, màu sắc, thời điểm mặc, lý do mua hàng...

- Ví dụ: "Tôi muốn mua một chiếc áo khoác dáng dài nhưng lại lo vì chiều cao của mình chỉ có 1m50. Có mẫu nào khiến tôi trông cao hơn không? Gợi ý giúp tôi cả cách phối đồ với chiếc áo đó nhé."
- "Tôi là một người có dáng người hơi mập và thấp, vai rộng, vòng 2 lớn. Tôi muốn mặc một bộ đồ khiến tôi trông thon gọn hơn để dự tiệc, nhưng vẫn phải thoải mái. Shop có thể gợi ý một set đồ và phụ kiện kèm theo không?"

#### • Trả lời câu hỏi thường gặp (FAQ):

- Chatbot tự động trả lời các câu hỏi thường gặp về chính sách bán hàng, vận chuyển, đổi trả, và các thông tin khác.
- Ví dụ: "Sản phẩm này có freeship không?"

#### • Phân tích và phản hồi phản hồi của người dùng:

- Chatbot có thể thu thập và phân tích phản hồi từ người dùng để cải thiện chất lượng dịch vụ và đưa ra các đề xuất phù hợp hơn.

### 1.2 Các phương pháp dùng trong đồ án

#### • RAG (Retrieval-Augmented Generation):

- Định nghĩa:** RAG là mô hình kết hợp hai thành phần chính: (1) Truy xuất thông tin từ cơ sở dữ liệu (Retrieval) và (2) Tạo văn bản tự nhiên (Generation). Mô hình này giúp chatbot trả lời câu hỏi một cách chính xác và tự nhiên bằng cách kết hợp thông tin từ nguồn dữ liệu bên ngoài với khả năng tạo văn bản của mô hình ngôn ngữ lớn (LLMs) (Mo et al., 2024).
- Ứng dụng trong đồ án:** RAG được sử dụng để tìm kiếm thông tin sản phẩm từ cơ sở dữ liệu và tạo câu trả lời tự nhiên dựa trên thông tin đó.

#### • Router:

076	– <i>Định nghĩa:</i> Router là cơ chế định tuyến giúp phân loại câu hỏi của người dùng và chuyển hướng đến module xử lý phù hợp. Router sử dụng mô hình embedding để tính toán điểm tương đồng giữa câu hỏi và các mẫu đã được định nghĩa trước.	121
077		122
078		123
079		
080		
081		
082	– <i>Ứng dụng trong đồ án:</i> Router được sử dụng để phân loại câu hỏi thành 2 loại: hỏi đáp về sản phẩm và hỏi đáp cá câu hỏi thông thường (không liên quan đến sản phẩm) từ đó chuyển hướng đến module tương ứng.	
083		
084		
085		
086		
087	• <b>Reflection:</b>	
088	– <i>Định nghĩa:</i> Reflection là cơ chế cho phép chatbot tự đánh giá và điều chỉnh câu trả lời dựa trên phản hồi từ người dùng hoặc dữ liệu lịch sử. Reflection giúp cải thiện độ chính xác và tự nhiên của câu trả lời.	
089		
090		
091		
092		
093	– <i>Ứng dụng trong đồ án:</i> Reflection được sử dụng để chatbot tự động điều chỉnh câu trả lời khi người dùng không hài lòng, đồng thời học hỏi từ các phản hồi để cải thiện chất lượng tương tác.	
094		
095		
096		
097		
098	• <b>History Truncated:</b>	
099	– <i>Định nghĩa:</i> History Truncated (Li et al., 2021) là kỹ thuật giảm thiểu số lượng token trong lịch sử hội thoại bằng cách loại bỏ các thông tin không cần thiết. Kỹ thuật này giúp tối ưu hóa hiệu suất và giảm chi phí tính toán khi gửi dữ liệu đến mô hình ngôn ngữ lớn (LLMs).	
100		
101		
102		
103		
104		
105		
106	– <i>Ứng dụng trong đồ án:</i> History Truncated được sử dụng để giới hạn độ dài lịch sử hội thoại, đảm bảo chatbot chỉ sử dụng thông tin cần thiết để tạo câu trả lời, từ đó giảm thiểu thời gian xử lý và tài nguyên hệ thống.	
107		
108		
109		
110		
111	<b>2 Lý do chọn đề tài</b>	
112	Chúng em quyết định chọn đề tài xây dựng chatbot sử dụng mô hình <b>RAG (Retrieval-Augmented Generation)</b> kết hợp với <b>Router</b> và <b>Reflection</b> vì những lý do sau:	
113		
114		
115		
116	<b>Nhu cầu thực tế</b>	
117	• Thương mại điện tử, đặc biệt là lĩnh vực thời trang, đang phát triển mạnh mẽ. Người dùng cần một công cụ hỗ trợ tìm kiếm và tư vấn sản phẩm nhanh chóng, chính xác.	
118		
119		
120		
	• Các cửa hàng trực tuyến cần giải pháp tự động hóa để xử lý yêu cầu khách hàng một cách hiệu quả.	124
	<b>Tính ưu việt của mô hình RAG</b>	125
	• RAG kết hợp khả năng tìm kiếm thông tin và tạo văn bản tự nhiên, giúp chatbot phản hồi chính xác và gần gũi với người dùng.	126
		127
	• Việc tích hợp <b>Router</b> và <b>Reflection</b> giúp hệ thống linh hoạt hơn, có khả năng tự điều chỉnh và cải thiện chất lượng phản hồi.	128
		129
		130
	<b>Tối ưu hóa hiệu suất</b>	131
	• Sử dụng <b>history truncated</b> để giảm token, tiết kiệm chi phí và tăng tốc độ phản hồi.	132
		133
	• Phương pháp <b>hybrid search</b> kết hợp tìm kiếm từ khóa và ngữ nghĩa, đảm bảo độ chính xác cao.	134
		135
		136
	<b>Tính khả thi và ứng dụng</b>	137
	• Dữ liệu hơn 13.000 sản phẩm thời trang được lưu trữ trên <b>MongoDB</b> (Banker et al., 2016), dễ dàng mở rộng và quản lý.	138
		139
		140
	• Sử dụng mô hình <b>embedding tiếng Việt 768</b> chiều, phù hợp với thị trường trong nước.	141
		142
	<b>Mục tiêu</b>	143
	Chúng em hướng đến xây dựng một chatbot thông minh, không chỉ hỗ trợ tìm kiếm sản phẩm mà còn có khả năng học hỏi và cải thiện theo thời gian, mang lại trải nghiệm mua sắm tốt nhất cho người dùng.	144
		145
		146
		147
		148
	<b>3 Dữ liệu</b>	149
	<b>3.1 Dữ liệu sử dụng</b>	150
	Dữ liệu của hệ thống bao gồm hơn 13.000 sản phẩm thuộc các danh mục thời trang như quần áo, giày dép, túi xách, đồng hồ, váy đầm... Mỗi sản phẩm được mô tả chi tiết với các thông tin như:	151
		152
		153
		154
	• Tên sản phẩm, giá, khuyến mãi, số lượng tồn kho.	155
		156
	• Thuộc tính như màu sắc, kích thước.	157
	• Thông tin cửa hàng (tên cửa hàng, đánh giá, khu vực kho hàng).	158
		159
	Dữ liệu được lưu trữ trên <b>MongoDB</b> , một hệ quản trị cơ sở dữ liệu NoSQL linh hoạt và hiệu suất cao.	160
		161

### 3.2 Crawl dữ liệu từ Sendo và Tiki

Dùng **Beautiful Soup (BS4)** (Richardson, 2007) và **Scrapy** (Kouzis-Loukas, 2016) để crawl dữ liệu từ 2 sàn thương mại điện tử lớn ở Việt Nam là Sendo và Tiki. Quy trình crawl dữ liệu:

- Xác định các trang web cần crawl (ví dụ: danh mục sản phẩm thời trang trên Sendo và Tiki).
- Sử dụng BS4 để phân tích cấu trúc HTML và trích xuất thông tin sản phẩm như tên, giá, mô tả, hình ảnh, và thuộc tính.
- Sử dụng Scrapy để tự động hóa quá trình crawl dữ liệu từ nhiều trang và lưu trữ dữ liệu vào file JSON hoặc CSV.
- Làm sạch dữ liệu sau khi crawl để loại bỏ các thông tin không cần thiết hoặc trùng lặp.

### 3.3 Tiền xử lý dữ liệu

Các phương pháp tiền xử lý dữ liệu sau khi crawl từ internet:

- **Chuẩn hóa dữ liệu:** Đảm bảo các trường dữ liệu có định dạng thống nhất (ví dụ: chuyển đổi giá thành số, chuẩn hóa đơn vị đo lường).
- **Xử lý dữ liệu thiếu:** Điền giá trị mặc định hoặc loại bỏ các bản ghi thiếu thông tin quan trọng.
- **Tách thuộc tính:** Tách các thuộc tính như màu sắc, kích thước thành các trường riêng biệt để dễ dàng truy vấn.
- **Loại bỏ dữ liệu trùng lặp:** Sử dụng các thuật toán so sánh để loại bỏ các sản phẩm trùng lặp.
- **Chuyển thành dạng json:** Để dễ truy vấn, găm tài nguyên lưu trữ

### 3.4 Đưa dữ liệu vào MongoDB

- **Embedding các cột cần truy vấn:** Tên sản phẩm và mô tả sản phẩm được chuyển đổi thành vector sử dụng mô hình embedding (ví dụ: sentence-transformers/all-MiniLM-L6-v2<sup>1</sup>).
- **Đưa dữ liệu vào MongoDB Atlas:** Sử dụng tính năng **Data Import** của MongoDB Atlas để tải dữ liệu từ file JSON hoặc CSV vào cơ sở dữ liệu.

<sup>1</sup><https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>

- **Tạo vector search:** Sử dụng câu lệnh sau để tạo chỉ mục vector search với độ đo cosine và chiều dữ liệu là 768:

```
db.products.createIndex(  
    { embedding: "cosineSimilarity"  
    },  
    { dimension: 768 } );
```

- **Tạo keyword search:** Sử dụng câu lệnh sau để tạo chỉ mục full-text search cho các trường như tên sản phẩm và mô tả:

```
db.products.createIndex(  
    { name: "text", description: "  
    text" } );
```

- **Kết nối MongoDB với driver Python:** Sử dụng thư viện **pymongo** để kết nối và tương tác với MongoDB từ Python: *pip install pymongo*  
Ví dụ kết nối:

```
from pymongo import MongoClient  
client = MongoClient("mongodb+srv  
://<username>:<password>@cluster0  
.mongodb.net/myFirstDatabase?  
retryWrites=true&w=majority")  
db = client["mydatabase"]  
collection = db["products"]
```

## 4 Công nghệ sử dụng

### 4.1 Database

- MongoDB:
  - **Định nghĩa:** MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL, lưu trữ dữ liệu dưới dạng các tài liệu JSON (BSON). MongoDB nổi bật với tính linh hoạt, khả năng mở rộng và hiệu suất cao, phù hợp với các ứng dụng cần xử lý dữ liệu lớn và đa dạng.
  - **Ứng dụng trong đồ án:**
    - \* Lưu trữ dữ liệu sản phẩm: Thông tin sản phẩm như tên, giá, mô tả, thuộc tính (màu sắc, kích thước) và thông tin cửa hàng được lưu trữ trong các collection của MongoDB.
    - \* Hỗ trợ tìm kiếm thông minh: MongoDB được tích hợp với các chỉ mục (index) để hỗ trợ tìm kiếm văn bản (text search) và tìm kiếm vector (vector search), giúp chatbot truy xuất thông tin nhanh chóng và chính xác.
    - \* Lưu trữ lịch sử hội thoại: Các cuộc trò chuyện giữa người dùng và chatbot được lưu trữ để phục vụ cho việc phân tích và cải thiện chất lượng tương tác.

257 \* Kết nối với hệ thống: Sử dụng thư viện  
258 pymongo để kết nối và tương tác với  
259 MongoDB từ ứng dụng Python.

## 260 4.2 Frontend và Backend

### 261 Backend

- 262 • FastAPI <sup>2</sup>:
  - 263 – FastAPI là một framework web hiện đại,  
264 nhanh chóng và dễ sử dụng, được xây dựng  
265 dựa trên Python, hỗ trợ tự động tạo tài liệu  
266 API và xử lý các yêu cầu đồng thời một cách  
267 hiệu quả.
  - 268 – *Ứng dụng*: FastAPI được sử dụng để xây  
269 dựng các API kết nối giữa front-end và back-  
270 end của hệ thống, đảm bảo truyền tải dữ liệu  
271 nhanh chóng và bảo mật.
- 272 • Docker <sup>3</sup>:
  - 273 – Docker là một nền tảng mã nguồn mở giúp  
274 đóng gói ứng dụng và các thành phần phụ  
275 thuộc vào các container, cho phép triển khai  
276 ứng dụng một cách nhất quán trên các môi  
277 trường khác nhau.
  - 278 – *Ứng dụng*: Docker được sử dụng để đóng  
279 gói và triển khai hệ thống một cách dễ dàng,  
280 đảm bảo tính nhất quán và khả năng mở  
281 rộng.

### 282 Frontend

- 283 • Node.js:
  - 284 – Node.js là một nền tảng chạy mã JavaScript  
285 phía server, cho phép xây dựng các ứng dụng  
286 web nhanh chóng và hiệu quả.
  - 287 – *Ứng dụng*: Node.js được sử dụng để xây  
288 dựng giao diện người dùng (UI) đơn giản và  
289 tương tác với hệ thống thông qua các API  
290 được cung cấp bởi FastAPI.

## 291 5 Kiến trúc hệ thống

### 292 5.1 Mô hình RAG (Retrieval-Augmented 293 Generation)

294 **Figure 1** mô tả pipeline cơ bản của một mô hình  
295 RAG. Mô hình RAG kết hợp hai thành phần chính:

<sup>2</sup><https://fastapi.tiangolo.com/>

<sup>3</sup><https://docs.docker.com/>

### Retrieval (Truy xuất thông tin)

- 297 • **Cơ chế tìm kiếm**: Sử dụng cơ chế tìm kiếm  
298 để truy xuất thông tin từ cơ sở dữ liệu. Trong  
299 đồ án này, nhóm chúng em sử dụng kết hợp  
300 **Semantic Search** và **Keyword Search** (cả hai  
301 đều có sẵn trên MongoDB Atlas) để tạo thành  
302 **Hybrid Search**.  
303
- 304 • **Điểm mạnh của Hybrid Search**:
  - 305 – Kết hợp ưu điểm của cả hai phương pháp:  
306 Semantic Search hiểu được ngữ nghĩa của  
307 câu hỏi, trong khi Keyword Search đảm bảo  
308 độ chính xác cao với các từ khóa cụ thể.
  - 309 – Giúp hệ thống linh hoạt hơn trong việc tìm  
310 kiếm thông tin, đặc biệt khi người dùng đưa  
311 ra các câu hỏi phức tạp hoặc không rõ ràng.
- 312 • **Quy trình tìm kiếm**:
  - 313 – Sau khi có  $n$  kết quả từ Semantic Search,  
314 nhóm chúng em kết hợp chúng với kết quả  
315 từ Keyword Search.
  - 316 – Sử dụng phương pháp **RRF (Reciprocal  
317 Rank Fusion)** để rerank và lấy ra  $k$  kết quả  
318 tốt nhất ( $k < n$ ,  $k$  có thể điều chỉnh).
- 319 • **Chú thích về RRF**:
  - 320 – RRF là một phương pháp rerank dựa trên  
321 công thức:

$$\text{RRF}(d) = \sum_{i=1}^m \frac{1}{k + \text{rank}_i(d)}$$

322 Trong đó:

- 323 \*  $d$ : Tài liệu cần xếp hạng.
- 324 \*  $\text{rank}_i(d)$ : Thứ hạng của tài liệu  $d$  trong  
325 danh sách kết quả thứ  $i$ .
- 326 \*  $k$ : Hằng số điều chỉnh (thường được đặt  
327 là 60).
- 328 – Phương pháp này được ứng dụng rộng rãi  
329 trong các hệ thống tìm kiếm kết hợp nhiều  
330 nguồn dữ liệu khác nhau.

### 331 Generation (Tạo văn bản)

- 332 • **Mô hình ngôn ngữ lớn (LLM)**: Sử dụng mô  
333 hình ngôn ngữ lớn để tạo câu trả lời dựa trên  
334 thông tin đã truy xuất.
- 335 • **Model sử dụng**: Trong đồ án này, nhóm chúng  
336 em sử dụng mô hình gemini-1.5-flash, một mô  
337 hình ngôn ngữ hiệu suất cao và nhẹ, phù hợp  
338 với các ứng dụng thời gian thực.





Figure 1: Basic RAG pipeline

- **API Key từ Gemini (Team et al., 2024):** Nhóm chúng em sử dụng API Key từ Gemini để tích hợp mô hình vào hệ thống, giúp giảm tải nguyên tính toán và tối ưu hóa thời gian phản hồi.
- **Lợi ích:**
  - Giảm tải cho hệ thống: Mô hình nhẹ và hiệu quả giúp giảm thời gian loading và tải nguyên tính toán.
  - Tạo câu trả lời tự nhiên và chính xác: Mô hình Gemini được huấn luyện để hiểu ngữ cảnh và tạo văn bản gần gũi với người dùng.

## 5.2 RAG kết hợp Router

**Figure 2** minh họa pipeline RAG có router. Router là một cơ chế định tuyến, giúp phân loại câu hỏi của người dùng và chuyển hướng đến module xử lý phù hợp. Điều này không chỉ tối ưu hóa quy trình xử lý mà còn giảm tải cho hệ thống, đảm bảo hoạt động hiệu quả.

- **ChitChat:** Xử lý các câu hỏi không liên quan đến sản phẩm, chẳng hạn như: "Hôm nay ăn gì?", "Hãy kể cho tôi một câu chuyện cười"...
- **Product:** Xử lý các câu hỏi liên quan đến sản phẩm, chẳng hạn như: "Tư vấn cho tôi đồ đi dự tiệc", "Sản phẩm đầm đen đẹp"...

Hệ thống sử dụng Semantic Router, một cơ chế định tuyến thông minh dựa trên mô hình ngôn ngữ để phân loại và chuyển hướng câu hỏi của người dùng đến các module xử lý phù hợp. Cụ thể, quy trình hoạt động của Router được thực hiện như sau:

### 5.2.1 Chuyển đổi câu hỏi thành vector

- **Mô hình embedding:** Hệ thống sử dụng mô hình sentence-transformers/all-MiniLM-L6-v2 từ Hugging Face để chuyển đổi câu hỏi của người dùng thành các vector. Mô hình này được huấn luyện để hiểu ngữ nghĩa của câu và biểu diễn chúng dưới dạng vector 384 chiều.
- **Ưu điểm:** Việc sử dụng mô hình embedding giúp hệ thống hiểu được ý nghĩa của câu hỏi một cách sâu sắc, thay vì chỉ dựa trên từ khóa đơn thuần.

### 5.2.2 Tính toán điểm tương đồng

- **Thư viện semantic\_router:** Hệ thống sử dụng thư viện mã nguồn mở semantic\_router để tính toán điểm tương đồng giữa vector của câu hỏi và các vector mẫu đã được định nghĩa trước.
- **Các mẫu định nghĩa:** Mỗi module xử lý (ví dụ: tìm kiếm sản phẩm, hỏi đáp thông tin) được liên kết với một tập hợp các câu hỏi mẫu. Các câu hỏi mẫu này được chuyển đổi thành vector và lưu trữ trong hệ thống.
- **Phương pháp tính toán:** Điểm tương đồng được tính toán dựa trên khoảng cách cosine giữa các vector. Khoảng cách càng nhỏ, câu hỏi càng có khả năng thuộc về module đó.

### 5.2.3 Xếp hạng và chuyển hướng

- **Xếp hạng module:** Dựa trên điểm tương đồng, hệ thống xếp hạng các module và chọn module có điểm số cao nhất.
- **Chuyển hướng câu hỏi:** Câu hỏi của người dùng được chuyển đến module phù hợp để xử lý. Ví dụ:
  - Nếu câu hỏi liên quan đến tìm kiếm sản phẩm, hệ thống sẽ chuyển hướng đến module tìm kiếm.
  - Nếu câu hỏi mang tính chất hỏi đáp thông tin, hệ thống sẽ chuyển hướng đến module hỏi đáp.

### 5.2.4 Tối ưu hóa hiệu suất

- **Giảm độ phức tạp tính toán:** Bằng cách sử dụng mô hình embedding nhỏ gọn như all-MiniLM-L6-v2, hệ thống đảm bảo hiệu suất cao và thời gian phản hồi nhanh chóng.
- **Linh hoạt và mở rộng:** Semantic Router cho phép dễ dàng thêm hoặc bớt các module xử lý mà không cần thay đổi cấu trúc hệ thống.

### 5.2.5 Ví dụ minh họa

- **Câu hỏi:** "Tôi muốn mua một chiếc váy màu đen, size M."
- **Quy trình xử lý:**
  - Câu hỏi được chuyển đổi thành vector bằng mô hình all-MiniLM-L6-v2.
  - Hệ thống tính toán điểm tương đồng với các vector mẫu của các module.
  - Nếu điểm tương đồng cao nhất thuộc về module tìm kiếm sản phẩm, câu hỏi sẽ được chuyển hướng đến module này để xử lý.

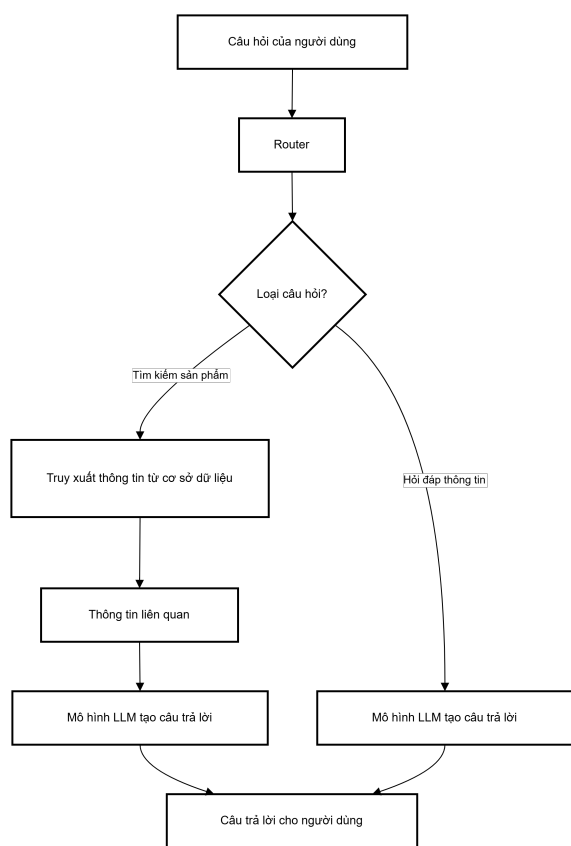


Figure 2: RAG with router

## Tăng tính tự nhiên trong giao tiếp:

- Reflection giúp chatbot duy trì tính liên tục trong cuộc trò chuyện, tạo cảm giác như đang nói chuyện với một người thật sự thay vì một cỗ máy.

## Ví dụ minh họa

Giả sử người dùng hỏi:

- Người dùng:** "Tôi muốn mua một chiếc váy màu đen, size M."
- Chatbot:** "Dưới đây là một số sản phẩm phù hợp: [Danh sách sản phẩm]."
- Người dùng:** "Tôi không thích những sản phẩm này, có cái nào khác không?"

Với reflection, chatbot sẽ:

- Đánh giá lại câu trả lời trước đó.
- Hiểu rằng người dùng không hài lòng với các sản phẩm được đề xuất.
- Điều chỉnh và đưa ra danh sách sản phẩm khác phù hợp hơn.

## 5.3 Reflection trong RAG

Reflection là một cơ chế tiên tiến cho phép chatbot tự đánh giá và cải thiện câu trả lời dựa trên phản hồi từ người dùng hoặc dữ liệu lịch sử. Phương pháp này không chỉ giúp chatbot hiểu rõ hơn về ngữ cảnh của cuộc trò chuyện mà còn nâng cao khả năng tương tác tự nhiên và chính xác. **Figure 4** minh họa reflection trong RAG.

### Cơ chế hoạt động của Reflection

#### Tích hợp lịch sử hội thoại:

- Thay vì chỉ gửi truy vấn hiện tại của người dùng, reflection yêu cầu gửi toàn bộ lịch sử cuộc trò chuyện mỗi khi gọi API.
- Lịch sử hội thoại bao gồm các câu hỏi và câu trả lời trước đó, giúp chatbot hiểu được ngữ cảnh và mục đích của người dùng một cách đầy đủ hơn.

### Lợi ích của Reflection

#### Cải thiện độ chính xác:

- Nhờ việc hiểu ngữ cảnh đầy đủ, chatbot có thể đưa ra câu trả lời chính xác hơn, tránh các hiểu lầm hoặc sai sót do thiếu thông tin.

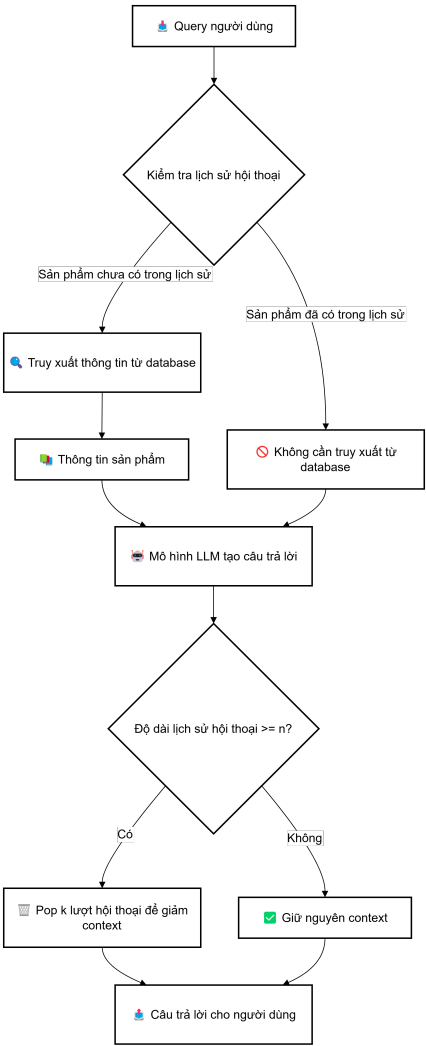


Figure 3: RAG with reflection

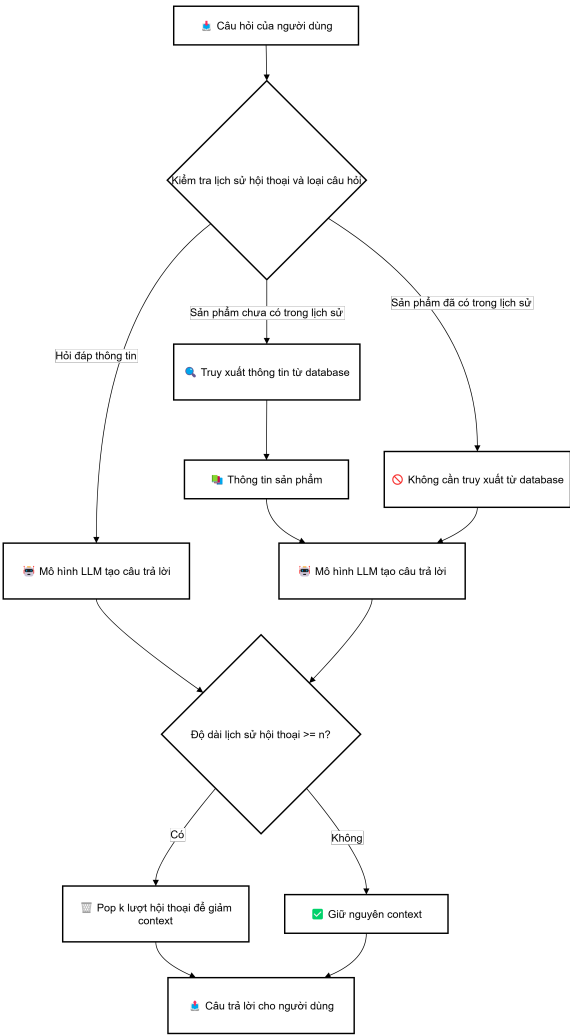


Figure 4: RAG pipeline

6 Đánh giá

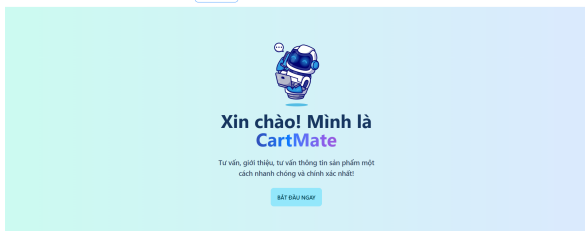
6.1 Phương pháp đánh giá

Retrieval Performance

- **Độ chính xác (Precision):** Tỷ lệ các tài liệu truy xuất được là phù hợp so với tổng số tài liệu truy xuất.
- **Thời gian phản hồi (Response Time):** Thời gian trung bình để hệ thống truy xuất thông tin và trả về kết quả.
- **Phương pháp đo lường:** Sử dụng bộ dữ liệu kiểm tra (test dataset) và so sánh kết quả truy xuất với ground truth.

Generation Quality

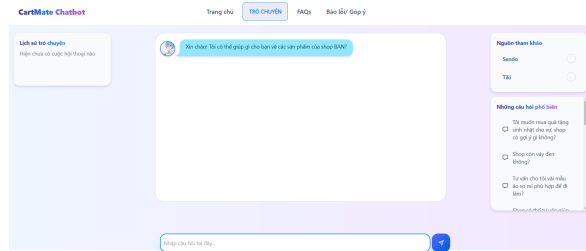
- **Độ chính xác của thông tin:** Đánh giá xem câu trả lời có chứa thông tin chính xác so với dữ liệu

482	tham chiếu hay không.	7 Hạn chế	521
483	• <b>Tính tự nhiên của câu trả lời:</b> Đánh giá mức	7.1 Khả năng xử lý câu hỏi phức tạp	522
484	độ tự nhiên và trôi chảy của câu trả lời thông	• <b>Mô tả:</b> Hệ thống gặp khó khăn trong việc xử lý	523
485	qua các tiêu chí như ngữ pháp, từ vựng, và cấu	các câu hỏi phức tạp hoặc đa nghĩa, dẫn đến câu	524
486	trúc câu.	trả lời không chính xác hoặc không đầy đủ.	525
487	• <b>Khả năng phản hồi phù hợp với ngữ cảnh:</b>	• <b>Hướng giải quyết:</b> Cải thiện mô hình ngôn	526
488	Đánh giá xem câu trả lời có phù hợp với ngữ	ngữ lớn (LLM) bằng cách huấn luyện thêm trên	527
489	cảnh của câu hỏi hay không.	các bộ dữ liệu chuyên sâu và tích hợp các kỹ	528
490	• <b>Phương pháp đo lường:</b> Sử dụng đánh giá	thuật xử lý ngữ cảnh phức tạp như Contextual	529
491	của con người (human evaluation) và các chỉ số	Embedding.	530
492	tự động như BLEU, ROUGE để đo lường chất	7.2 Thời gian phản hồi chậm với lượng dữ liệu	531
493	lượng câu trả lời.	lớn	532
494	6.2 Kết quả đánh giá	• Khi lượng dữ liệu truy xuất lớn, thời gian phản	533
495	Retrieval Performance	hồi của hệ thống có thể bị chậm, ảnh hưởng đến	534
496	• <b>Độ chính xác (Precision): 90%.</b>	trải nghiệm người dùng.	535
497	• <b>Thời gian phản hồi (Response Time):</b> trung	• <b>Hướng giải quyết:</b> Tối ưu hóa cơ sở dữ liệu	536
498	bình 10 giây.	bằng cách sử dụng các chỉ mục (index) hiệu quả	537
499	Generation Quality	hơn và triển khai hệ thống trên các nền tảng	538
500	• <b>Độ chính xác của thông tin: 90%</b>	đám mây có khả năng mở rộng (scalable cloud	539
501	• <b>Tính tự nhiên của câu trả lời: 80%.</b>	platforms).	540
502	• <b>Khả năng phản hồi phù hợp với ngữ cảnh:</b>	7.3 Hạn chế 3: Hạn chế về ngôn ngữ	541
503	<b>90%</b>	• Hệ thống hiện tại chủ yếu hỗ trợ tiếng Việt, gặp	542
504	• <b>So sánh giữa các mô hình LLM:</b>	khó khăn khi xử lý các ngôn ngữ khác.	543
505	– Mô hình gemini-1.5-flash cho kết quả tốt	• <b>Hướng giải quyết:</b> Tích hợp thêm các mô hình	544
506	hơn so với các mô hình khác trong việc tạo	đa ngôn ngữ (multilingual models) như mBERT	545
507	câu trả lời tự nhiên và chính xác.	hoặc XLM-R để hỗ trợ nhiều ngôn ngữ hơn.	546
508	6.3 Nhận xét	8 CartMate - Chatbot tư vấn sản phẩm	547
509	Retrieval Performance	thời trang	548
510	• Hệ thống truy xuất thông tin đạt hiệu suất cao	8.1 Minh họa hệ thống	549
511	với độ chính xác và độ phủ tốt, đặc biệt khi sử	Trang chủ	550
512	dụng Hybrid Search.		
513	• Thời gian phản hồi nhanh chóng, đáp ứng được	Giao diện có các chức năng chính	551
514	yêu cầu của ứng dụng thời gian thực.	• Trang chủ: Quay lại giao diện chính của chatbot.	552
515	Generation Quality	• Trò chuyện: Truy cập vào phần trò chuyện với	553
516	• Mô hình gemini-1.5-flash tạo ra các câu trả lời	chatbot.	554
517	tự nhiên và chính xác, phù hợp với ngữ cảnh của		
518	câu hỏi.		
519	• Tuy nhiên, vẫn còn một số hạn chế trong việc		
520	xử lý các câu hỏi phức tạp hoặc đa nghĩa.		



- FAQs: Giải đáp thắc mắc và hướng dẫn người dùng cách sử dụng chatbot.
- FAQs: Hiển thị các câu hỏi thường gặp để người dùng tham khảo.
- Bắt đầu ngay: Bắt đầu tương tác với chatbot CartMate để nhận tư vấn sản phẩm.

## Trò chuyện



- **Lịch sử trò chuyện (bên trái):**
  - Hiển thị danh sách các cuộc hội thoại đã diễn ra.
- **Hộp thoại chính (ở giữa):**
  - Hiển thị tin nhắn chào hỏi từ chatbot: "Xin chào! Tôi có thể giúp gì cho bạn về các sản phẩm của shop BAN?".
- **Nguồn tham khảo (bên phải):**
  - Người dùng có thể chọn một trong hai nguồn thông tin: **Sendo** hoặc **Tiki**.
- **Những câu hỏi phổ biến (bên phải):**
  - Hiển thị danh sách các câu hỏi gợi ý, ví dụ:
    - \* "Tôi muốn mua quà tặng sinh nhật cho vợ, shop có gợi ý gì không?"
    - \* "Shop còn váy đen không?"
    - \* "Tư vấn cho tôi vài mẫu áo sơ mi phù hợp để đi làm?"
  - Người dùng có thể nhập câu hỏi vào thanh nhập liệu và nhấn biểu tượng gửi để tương tác với chatbot.
  - Hiển thị thông báo: "Mô hình có thể đưa ra câu trả lời không chính xác ở một số trường hợp, vì vậy hãy luôn kiểm chứng thông tin bạn nhé!".

## FAQs

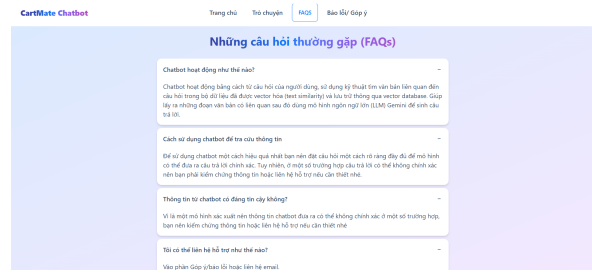


Figure 5: Enter Caption

## Báo lỗi & Góp ý

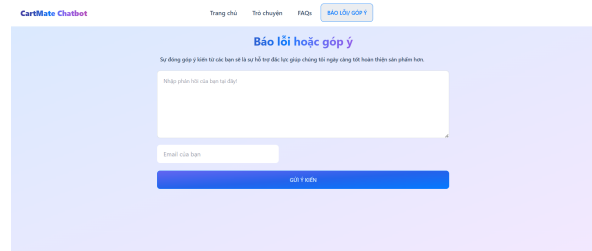


Figure 6: Enter Caption

- ### 8.2 Các điểm hạn chế của hệ thống
- ### 8.3 Các hướng đi cải thiện & mở rộng hệ thống
- **Tích hợp thêm các mô hình embedding tiên tiến:**
    - Sử dụng các mô hình embedding lớn hơn và hiệu quả hơn như sentence-transformers/all-mpnet-base-v2 hoặc paraphrase-multilingual-MiniLM-L12-v2 để cải thiện độ chính xác của việc chuyển đổi câu hỏi thành vector.
    - Thử nghiệm với các mô hình đa ngôn ngữ để hỗ trợ nhiều ngôn ngữ khác nhau, phù hợp với thị trường toàn cầu.
  - **Tối ưu hóa cơ chế Reflection:**
    - Cải thiện khả năng tự đánh giá và điều chỉnh câu trả lời bằng cách tích hợp thêm các thuật toán học máy như Reinforcement Learning (RL) để chatbot học hỏi từ phản hồi của người dùng.
    - Xây dựng một cơ sở dữ liệu phản hồi để lưu trữ và phân tích các phản hồi từ người dùng, từ đó cải thiện chất lượng câu trả lời theo thời gian.
  - **Mở rộng dữ liệu và danh mục sản phẩm:**
    - Bổ sung thêm dữ liệu từ các danh mục sản phẩm khác như đồ gia dụng, thiết bị điện tử,

615 hoặc thực phẩm để mở rộng phạm vi ứng  
616 dụng của hệ thống.

- 617 – Tích hợp dữ liệu từ nhiều nguồn khác nhau  
618 (ví dụ: các sàn thương mại điện tử khác) để  
619 làm phong phú thông tin sản phẩm.
- 620 • **Cải thiện hiệu suất và tốc độ xử lý:**
  - 621 – Sử dụng các kỹ thuật nén và tối ưu hóa mô  
622 hình để giảm thiểu thời gian xử lý và tài  
623 nguyên tính toán.
  - 624 – Triển khai hệ thống trên các nền tảng đám  
625 mây (Cloud) để đảm bảo khả năng mở rộng  
626 và xử lý đồng thời nhiều yêu cầu từ người  
627 dùng.

## 628 9 Kết luận

629 Hệ thống chatbot sử dụng RAG kết hợp Router và  
630 Reflection đã mang lại một giải pháp thông minh  
631 và hiệu quả cho việc tìm kiếm và tư vấn sản phẩm.  
632 Với khả năng hiểu ngữ cảnh và tự điều chỉnh câu  
633 trả lời, hệ thống không chỉ cải thiện độ chính xác  
634 mà còn tối ưu hóa trải nghiệm người dùng.  
635 Trong tương lai, việc tích hợp thêm các mô hình  
636 tiên tiến và mở rộng dữ liệu sẽ giúp hệ thống trở  
637 nên linh hoạt hơn, đáp ứng được nhiều nhu cầu đa  
638 dạng trong thương mại điện tử và các lĩnh vực khác.

## 639 References

640 Kyle Banker, Douglas Garrett, Peter Bakkum, and  
641 Shaun Verch. 2016. *MongoDB in action: covers*  
642 *MongoDB version 3.0*. Simon and Schuster.

643 Dimitrios Kouzis-Loukas. 2016. *Learning scrapy*.  
644 Packt Publishing Livery Place.

645 Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio  
646 Petroni, Vladimir Karpukhin, Naman Goyal, Hein-  
647 rich Küttler, Mike Lewis, Wen-tau Yih, Tim  
648 Rocktäschel, et al. 2020. Retrieval-augmented gen-  
649 eration for knowledge-intensive nlp tasks. *Advances*  
650 *in Neural Information Processing Systems*, 33:9459–  
651 9474.

652 Juntao Li, Chang Liu, Chongyang Tao, Zhangming  
653 Chan, Dongyan Zhao, Min Zhang, and Rui Yan.  
654 2021. Dialogue history matters! personalized re-  
655 sponse selection in multi-turn retrieval-based chat-  
656 bots. *ACM Transactions on Information Systems*  
657 *(TOIS)*, 39(4):1–25.

658 Yuhong Mo, Hao Qin, Yushan Dong, Ziyi Zhu,  
659 and Zhenglin Li. 2024. Large language model  
660 (llm) ai text generation detection based on trans-  
661 former deep learning algorithm. *arXiv preprint*  
662 *arXiv:2405.06652*.

Leonard Richardson. 2007. Beautiful soup documenta-  
663 tion. 664

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan  
665 Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer,  
666 Damien Vincent, Zhufeng Pan, Shibo Wang, et al.  
667 2024. Gemini 1.5: Unlocking multimodal under-  
668 standing across millions of tokens of context. *arXiv*  
669 *preprint arXiv:2403.05530*. 670

Sonia Vakayil, D Sujitha Juliet, Sunil Vakayil, et al.  
671 2024. Rag-based llm chatbot using llama-2. In *2024*  
672 *7th International Conference on Devices, Circuits*  
673 *and Systems (ICDCS)*, pages 1–5. IEEE. 674