Semantic Spotter Project - Fashion Search AI

(LangChain-based Generative Semantic Search System)

Prepared by: **Huy Vo Duc**

Course 6 - GenAI Elective – Module 15

upGrad and IIITB Machine Learning & AI Program – November 2024

Contents

1 Background and Motivation 2
2 Objectives 3
3 Why LangChain? 3
4 Dataset Summary 4
5 Overall System Design 5
6 Conceptual System Layers 6
6.1 Data and Embedding Layer 6
6.2 Semantic Retrieval and Ranking Layer 6
6.3 Generative Recommendation Layer 6
7 Implementation Details 7
8 Challenges Faced 8
9 Lessons Learned 8
10 Conclusion 9

1 Background and Motivation

In previous assignments, semantic search systems were implemented using a custom, manual pipeline, where individual components such as embedding generation, vector similarity search, reranking logic, and response formatting were coded explicitly. While this approach helped build a strong conceptual understanding of retrieval augmented generation (RAG) systems, it introduced several challenges related to scalability, maintainability, and extensibility.

As modern LLM-based applications grow in complexity, managing these components manually can become error-prone and difficult to evolve. This motivates the adoption of higher-level frameworks such as LangChain, which provide abstractions for building robust, modular, and LLM-agnostic generative systems.

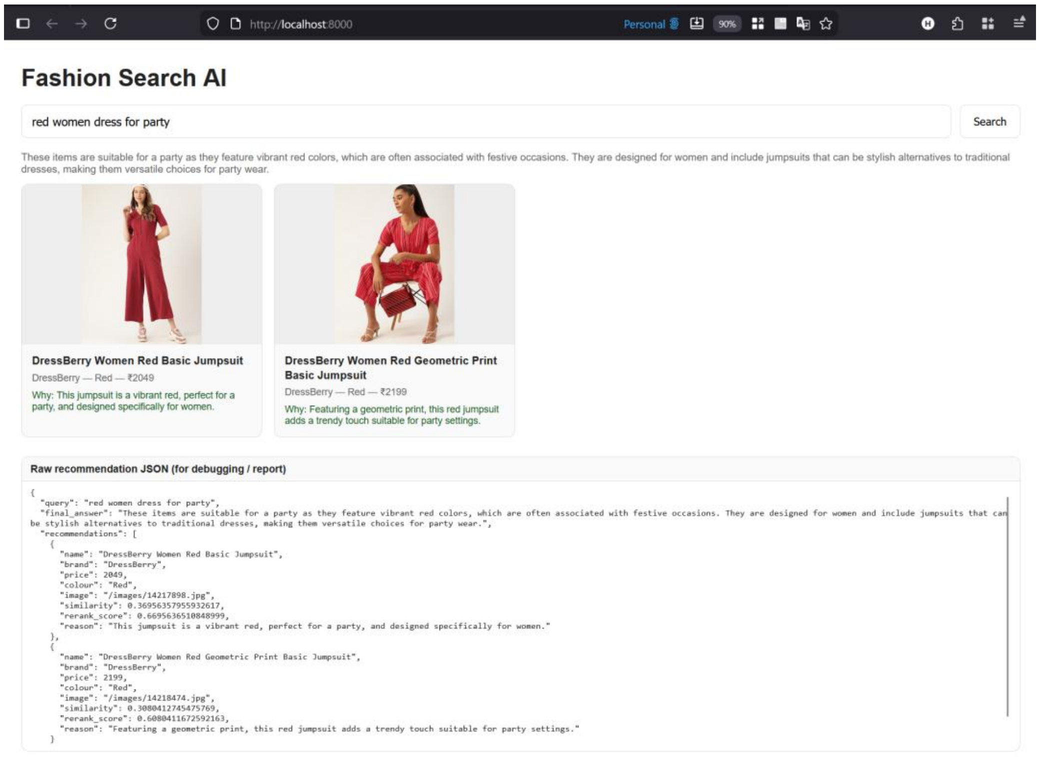
The objective of this project is to revisit the semantic search problem from a more production-oriented perspective, leveraging LangChain to simplify orchestration, reduce boilerplate code, and improve system clarity, while preserving the core idea of semantic retrieval and explainable recommendations.

2 Objectives

The goal of the **Fashion Search AI** project is to build an end-to-end generative recommendation system that allows users to search for a fashion catalog using natural language queries and receive:

- Relevant product recommendations based on semantic similarity
- Clear, human-readable explanations describing why each product matches the query

The system is designed to reflect a realistic e-commerce search scenario, where users may search using incomplete, vague, or intent-driven phrases (e.g., *“red women dress for party”*), rather than exact product names.



Fashion Search AI

red women dress for party

These items are suitable for a party as they feature vibrant red colors, which are often associated with festive occasions. They are designed for women and include jumpsuits that can be stylish alternatives to traditional dresses, making them versatile choices for party wear.

DressBerry Women Red Basic Jumpsuit
DressBerry — Red — ₹2049
Why: This jumpsuit is a vibrant red, perfect for a party, and designed specifically for women.

DressBerry Women Red Geometric Print Basic Jumpsuit
DressBerry — Red — ₹2199
Why: Featuring a geometric print, this red jumpsuit adds a trendy touch suitable for party settings.

Raw recommendation JSON (for debugging / report)

```
{
  "query": "red women dress for party",
  "final_answer": "These items are suitable for a party as they feature vibrant red colors, which are often associated with festive occasions. They are designed for women and include jumpsuits that can
be stylish alternatives to traditional dresses, making them versatile choices for party wear.",
  "recommendations": [
    {
      "name": "DressBerry Women Red Basic Jumpsuit",
      "brand": "DressBerry",
      "price": 2049,
      "colour": "Red",
      "image": "/images/14217898.jpg",
      "similarity": 0.36956357955932617,
      "rerank_score": 0.6695636510848999,
      "reason": "This jumpsuit is a vibrant red, perfect for a party, and designed specifically for women."
    },
    {
      "name": "DressBerry Women Red Geometric Print Basic Jumpsuit",
      "brand": "DressBerry",
      "price": 2199,
      "colour": "Red",
      "image": "/images/14218474.jpg",
      "similarity": 0.3080412745475769,
      "rerank_score": 0.6080411672592163,
      "reason": "Featuring a geometric print, this red jumpsuit adds a trendy touch suitable for party settings."
    }
```

Figure 1: example “red women dress for party”

3 Why LangChain?

LangChain plays a central role in this project and is not used merely as a wrapper library, but as the core orchestration framework.

Key reasons for choosing LangChain:

- **Abstraction of common LLM workflows**
  LangChain abstracts repetitive tasks such as embedding models, vector stores, and LLM invocation into reusable building blocks.
- **Seamless vector database integration**
  LangChain provides native integration with Chroma, enabling persistent vector stores to be reused across application restarts without manual client management.
- **LLM-agnostic design**
  The system can switch between an online LLM (e.g., OpenAI) and an offline rule-based or local generation strategy without changing the retrieval pipeline.
- **Cleaner and more modular codebase**

  Compared to the previous assignment, LangChain significantly reduces glue code and improves the readability of the overall system.

Comparison with previous manual implementation:

Aspect	**Previous Assignment**	**This Project (LangChain)**
Embedding handling	Manually implemented	Framework-managed embeddings
Vector search	Direct database API calls	LangChain vector store abstraction
LLM integration	Custom prompt + API calls	Prompt templates + LLM chains
Pipeline extensibility	Tightly coupled	Modular and configurable
Production readiness	Limited	Significantly improved

This demonstrates a clear evolution from **concept-focused implementation to a framework-driven, production-ready design.**

4 Dataset Summary

- Source: **Myntra Fashion Dataset** (Kaggle)
- Type: Product metadata and descriptions
- #Rows: ~14,000 usable records
- Product images including
- Price currency: INR (Indian Rupees)

5 Overall System Design

The Fashion Search AI system follows a modular, end-to-end design built using **FastAPI**, **LangChain**, and **Chroma**.

At a high level, the system consists of:

- A backend service exposing a recommendation API
- A persistent semantic vector store for fast retrieval
- A generative component to produce human-readable explanations
- A lightweight web-based user interface for interaction
- For development and tracing so, there is technical panel for displaying technical information

LangChain acts as the **central coordination layer**, connecting embedding models, the vector store, and the LLM-based generation logic into a unified pipeline.

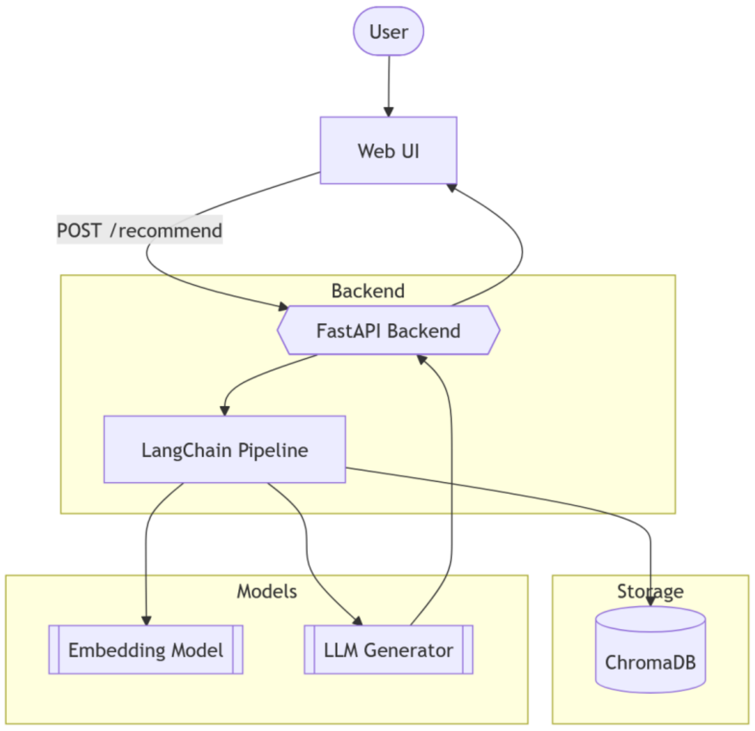


Figure 2: Overall System Architecture of Fashion Search AI

6 Conceptual System Layers

Although the system is presented to users as a single intelligent search experience, it can be conceptually divided into the following layers:

6.1 Data and Embedding Layer

Fashion product descriptions are converted into dense vector embeddings using a Sentence Transformer model (all-MiniLM-L6-v2). These embeddings capture semantic meaning beyond simple keywords and are stored persistently in a Chroma vector database.

6.2 Semantic Retrieval and Ranking Layer

When a user submits a query, LangChain retrieves the most semantically similar products from the vector store. A lightweight hybrid reranking strategy is then applied to improve relevance based on explicit attributes such as color, product type, and gender.

6.3 Generative Recommendation Layer

The top-ranked products are passed to a LangChain-driven generative component. This module produces a structured JSON response that includes:

- a concise summary of the results
- a short explanation for each recommendation explaining its relevance to the user query

This layer improves transparency and user trust by clearly communicating why certain products are recommended.

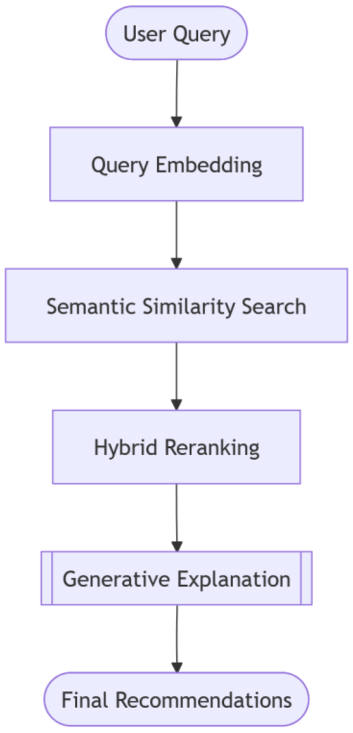


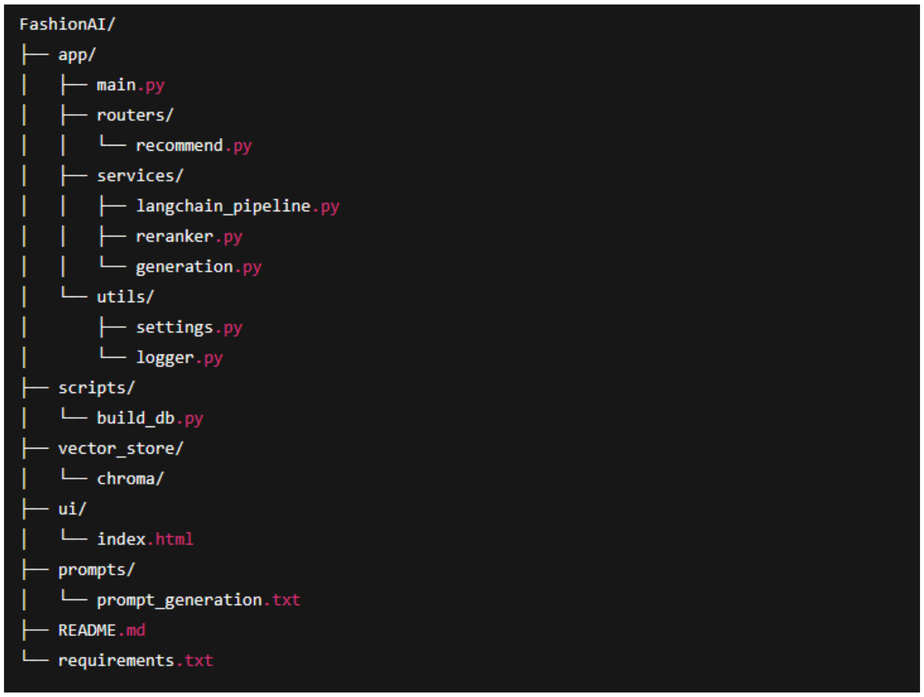
Figure 3: Semantic Spotter Search and Recommendation Workflow

7 Implementation Details

The backend is implemented using **FastAPI** and organized in a clean, modular structure.

- Key implementation aspects include:
- Precomputed embeddings stored in Chroma for fast startup and reuse
- LangChain vector store abstraction for retrieval
- Prompt templates designed to enforce structured JSON output
- A fallback generation mechanism for offline or API-key-free execution
- Environment-based configuration using .env files to avoid hardcoded values

The UI communicates with the backend through a single **/recommend** endpoint, making the system intuitive and aligned with real-world product search experiences.



```
FashionAI/
├── app/
│   ├── main.py
│   ├── routers/
│   │   └── recommend.py
│   ├── services/
│   │   ├── langchain_pipeline.py
│   │   ├── reranker.py
│   │   └── generation.py
│   └── utils/
│       ├── settings.py
│       └── logger.py
├── scripts/
│   └── build_db.py
├── vector_store/
│   └── chroma/
├── ui/
│   └── index.html
├── prompts/
│   └── prompt_generation.txt
├── README.md
└── requirements.txt
```

Figure 4: Product Structure

8 Challenges Faced

Several challenges were encountered during development:

- Dataset preprocessing required manual validation due to inconsistent records
- Library deprecations in LangChain and Chroma required dependency updates
- Designing prompts that reliably produce valid JSON outputs
- Managing development issues on Windows such as port conflicts and process handling
- Balancing semantic similarity with explicit constraints like color matching

Addressing these challenges led to a deeper understanding of both data-centric and system-level considerations in LLM-based applications.

9 Lessons Learned

This project provided several key takeaways:

- Frameworks like LangChain significantly improve development speed and code clarity
- Semantic search provides a superior user experience compared to keyword-based search

- Explainability is a crucial component of recommendation systems
- Modular system design enables easier experimentation and future extension
- Practical issues such as data quality and dependency management are critical in real-world AI projects

10 Conclusion

Fashion Search AI demonstrates how a modern generative search system can be built using LangChain to combine semantic retrieval with explainable recommendations. By transitioning from a manual pipeline in earlier assignments to a framework-driven approach, the project showcases not only technical competence but also architectural maturity.

The resulting system is robust, extensible, and well-suited for real-world fashion search applications, serving as a strong foundation for further enhancements such as personalization, filtering, and large-scale deployment.