Software Requirements Specification (SRS)

Project Title: ShopSmart AI - Kubernetes-Native Product Recommender with AstraDB & LangChain

1. Introduction

ShopSmart AI is an AI-powered product recommendation system that uses Retrieval-Augmented Generation (RAG) architecture to provide intelligent and context-aware responses based on customer reviews. The system leverages LangChain for orchestration, AstraDB for vector storage, and Groq-powered LLMs for response generation.

2. Overall Description

The system processes product reviews, converts them into embeddings using a transformer-based embedding model, stores them in AstraDB, and retrieves relevant reviews based on semantic similarity when a user query is received. The retrieved context is provided to a Large Language Model to generate grounded responses.

3. System Architecture

- Data Ingestion Layer: Converts CSV reviews into structured documents.
- Embedding Layer: Generates vector embeddings using HuggingFace models.
- Vector Database Layer: Stores embeddings in AstraDB for similarity search.
- Retrieval Layer: Fetches top-k relevant documents based on semantic similarity.
- Generation Layer: Uses Groq LLM (Llama 3.1) for response generation.
- Session Memory Layer: Maintains conversational context using session IDs.

4. Functional Requirements

- FR1: The system shall ingest product review data from CSV files.
- FR2: The system shall generate embeddings for each review.
- FR3: The system shall store embeddings in AstraDB.
- FR4: The system shall retrieve relevant reviews based on user query.
- FR5: The system shall generate responses grounded in retrieved context.
- FR6: The system shall support session-based conversational memory.

5. Non-Functional Requirements

- NFR1: The system should respond within acceptable latency limits.
- NFR2: The system should support horizontal scalability (Kubernetes-ready design).
- NFR3: The system should ensure modular architecture for maintainability.
- NFR4: The system should minimize hallucination through grounded RAG architecture.

6. Future Enhancements

- Deployment on Kubernetes cluster.
- Integration of monitoring tools like Prometheus and Grafana.
- Improved prompt engineering and reranking mechanisms.
- Frontend integration with real-time chat interface.