Phase 1: Requirement Analysis

AI-Driven SalesStore

Software Requirement Specification (SRS) for the AI-Driven SalesStore.

1 Functional Requirements (The Core)

Refining your initial list to meet industry standards.

1.1 Advanced Identity Management (Auth)

- **Requirement:** Instead of simple JWT implementation, use an Identity Provider (IdP).
- **Enterprise Twist:** Implement Keycloak or Spring Authorization Server. This handles "Role-Based Access Control (RBAC)" (Customer, Admin, Seller) securely and allows for future Single Sign-On (SSO).

1.2 Product Catalog & Taxonomy

- **Requirement:** Rich product cards with images, prices, ratings, and variant selection (size/color).
- **Enterprise Twist:** Support Inventory Management (locking items during checkout so you don't oversell) and Category Hierarchies (Electronics > Laptops > Gaming).

1.3 Shopping Experience

- **Requirement:** Home, Profile, Cart, Wishlist.
- **Enterprise Twist:** Persistent Cart (Redis-backed) so items remain if the user logs in from a different device.

1.4 Financial Operations

- **Requirement:** Payment Gateway (Stripe/Razorpay) and PDF Invoice generation.
- **Enterprise Twist:** Audit Trails. Every transaction must be logged immutably for accounting purposes.

2 AI-Driven Requirements (The "Non-Generic" Edge)

These features transform it from a standard CRUD app to a smart platform.

2.1 Semantic Search (RAG)

- **Requirement:** Users can search by intent, not just keywords.
- **Example:** Searching "something to wear for a summer wedding" should show floral dresses or light suits, even if the word "wedding" isn't in the title.
- **Tech:** Spring AI + PostgreSQL (pgvector).

2.2 Intelligent Seller Assistant

- **Requirement:** Help sellers list products faster.
- **Feature:** Seller uploads an image of a shoe; the AI analyzes it and auto-generates a Title, Description, and Tags.
- **Tech:** Multimodal LLM (Gemini Pro Vision or GPT-4o).

2.3 Conversational Shopping Agent

- **Requirement:** A persistent chat widget where users can ask questions about products.
- **Feature:** "Does this laptop support 64GB RAM?" The AI checks the technical specs document and answers.
- **Tech:** RAG with Document Ingestion.

2.4 Smart Review Summarization

- **Requirement:** Instead of showing 500 reviews, show a generated summary.
- **Feature:** "Pros: Great battery, bright screen. Cons: Fans are loud under load."

3 Non-Functional Requirements (The "Enterprise" Standard)

This is what interviewers look for when you say "Enterprise Ready".

3.1 Scalability & Resilience

- **Service Discovery:** Use Netflix Eureka or Consul so services can find each other dynamically.
- **Circuit Breakers:** Implement Resilience4j. If the "Inventory Service" is down, the "Product Page" should still load (just without stock status), rather than crashing the whole site.
- **API Gateway:** Use Spring Cloud Gateway as the single entry point to route traffic and handle rate limiting.

3.2 Observability (Monitoring)

- **Requirement:** You must know what is happening inside the system.
- **Tech:**
  - **Tracing:** Zipkin or Micrometer Tracing (to track a request across microservices).
  - **Metrics:** Prometheus & Grafana (to see CPU usage, request latency).

- **Logging:** Centralized logging (ELK Stack or Loki) – optional but impressive.

3.3 Performance

- **Caching:** Use Redis for frequently accessed data (like Product Categories or the Home Page) to reduce database load.

3.4 DevOps & Automation

- **Containerization:** Every service (Auth, Product, AI, UI) must have a Dockerfile.
- **Orchestration:** Use Docker Compose for local development (simulating a production cluster).

4 Proposed High-Level Tech Stack

- **Backend:** Java 21, Spring Boot 3.4
- **AI:** Spring AI, Ollama (local LLM) or OpenAI/Gemini API
- **Database:** PostgreSQL (Data + Vectors), Redis (Cache)
- **Frontend:** React.js or Angular (with Tailwind CSS for UI)
- **Architecture:** Microservices (Auth, Product, Order, Payment, AI-Service, Gateway)