High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components for **RetailSync - Inventory & Order Management Backend**. The system is a Spring Boot-based backend designed to manage retail inventory and orders, supporting CRUD operations, secure access, and integration with PostgreSQL. The solution is containerized for deployment flexibility and managed with Maven.

---

1. System Architecture Overview

Architecture Description:
RetailSync follows a layered architecture comprising API, Service, Data Access, Security, and Persistence layers. The backend exposes RESTful endpoints, processes business logic, manages data persistence, and secures access via JWT authentication.

Module/Component	Role/Responsibility
API Controller	Exposes REST endpoints for inventory & order operations
Service Layer	Implements business logic, DTO mapping
Data Access Layer	Handles CRUD with PostgreSQL via Spring Data JPA
Security Module	Manages JWT authentication & endpoint protection
Persistence (DB)	Stores inventory, orders, and user data (PostgreSQL)
Configuration/Infra	Docker containerization, Maven build management

---

2. Component Interactions

Sequence Step	Interaction Description
1	Client sends HTTP request to API Controller (with JWT if required)
2	Controller validates request, delegates to Service Layer
3	Service Layer processes logic, maps entities/DTOs, calls Data Access Layer
4	Data Access Layer interacts with PostgreSQL for data operations
5	Service Layer returns response DTO to Controller
6	Controller sends HTTP response to client
7	Security Module intercepts requests, validates JWT, enforces access control

---

3. Data Flow Overview

Data Flow Step	Description
User Authentication	JWT issued on login, attached to subsequent requests
CRUD Operations	API receives request → Service processes → Data Access persists/retrieves
Pagination/Sorting	API supports query params for pagination/sorting, handled in Service/DAO
DTO Mapping	Entities mapped to DTOs for API responses and requests

---

4. Technology Stack

Layer/Function	Technology/Framework
Language	Java
Framework	Spring Boot, Spring Data JPA
Database	PostgreSQL
Security	JWT, Spring Security
Build/Dependency Mgmt	Maven
Containerization	Docker

---

5. Scalability, Reliability & Security

- **Scalability:**
  - Stateless REST API enables horizontal scaling via container orchestration.
  - Database connection pooling and efficient query design support high concurrency.
- **Reliability:**
  - Exception handling and validation at API and service layers.
  - Docker ensures consistent deployment environments.
- **Security:**
  - Endpoints protected with JWT-based authentication and authorization.
  - Sensitive data (e.g., DB credentials) managed via environment variables/secrets.

End of Document