Product Requirements & Specification Document

Project Name

RetailSync - Inventory & Order Management Backend

Description

Develop a Spring Boot backend for a retail inventory and order management system. The system will support CRUD operations, pagination, sorting, and DTO mapping. Endpoints must be secured with JWT. PostgreSQL will be used for persistent storage. The application will be containerized with Docker and managed with Maven.

1. Goals & Objectives

Goal	Description
Inventory Management	CRUD operations for products and stock levels
Order Management	CRUD operations for customer orders
Secure API	JWT-based authentication and authorization
Efficient Data Access	Pagination and sorting for list endpoints
DTO Mapping	Use DTOs to decouple API and persistence layers
Open-Source & Startup Ready	Clean, maintainable, and containerized for easy deployment

2. Functional Requirements

2.1 Inventory

Feature	Description
Create Product	Add new product with details and stock level
Read Product(s)	Get product(s) with pagination & sorting
Update Product	Modify product details and stock
Delete Product	Remove product from inventory

2.2 Orders

Feature	Description
Create Order	Place new order for products
Read Order(s)	Get order(s) with pagination & sorting
Update Order	Modify order details/status

Delete Order	Remove order

2.3 Authentication

Feature	Description
User Login	JWT token issuance on valid credentials
Secure Endpoints	All endpoints require valid JWT

3. Non-Functional Requirements

Requirement	Specification
Performance	< 500ms response time for standard operations
Security	JWT, password hashing, input validation
Scalability	Stateless, containerized, ready for scaling
Maintainability	Modular code, DTO usage, clear structure
Documentation	OpenAPI/Swagger for API docs

4. Technical Specifications

4.1 Architecture

- **Backend:** Java 17+, Spring Boot (latest stable)
- **Database:** PostgreSQL (latest stable)
- **Containerization:** Docker
- **Build Tool:** Maven
- **API:** RESTful, JSON
- **Security:** Spring Security, JWT

4.2 Entity Overview

Entity	Fields (examples)
Product	id, name, sku, description, price, stockQuantity
Order	id, customerName, orderDate, status, items[]
OrderItem	id, orderId, productId, quantity, price
User	id, username, password (hashed), roles

4.3 API Endpoints (Sample)

```
POST   /api/auth/login            // JWT login
GET    /api/products              // List products (pagination, sorting)
POST   /api/products              // Create product
PUT    /api/products/{id}         // Update product
```

```
DELETE /api/products/{id}       // Delete product

GET    /api/orders              // List orders (pagination, sorting)
POST   /api/orders              // Create order
PUT    /api/orders/{id}         // Update order
DELETE /api/orders/{id}         // Delete order
```

4.4 DTO Mapping

- Use MapStruct or manual mapping for DTO <-> Entity conversion
- Expose only necessary fields in API responses

4.5 Pagination & Sorting

- Standard query params: `page` , `size` , `sort`
- Example: `/api/products?page=0&size=10&sort=name,asc`

4.6 Security

- JWT required for all endpoints except `/api/auth/login`
- Passwords stored hashed (e.g., BCrypt)
- Role-based access (future extensibility)

4.7 Containerization

- Dockerfile for backend
- Docker Compose for backend + PostgreSQL

5. Acceptance Criteria

ID	Criteria
AC1	All CRUD endpoints for products and orders function as specified
AC2	Pagination and sorting work on list endpoints
AC3	All endpoints (except login) require valid JWT
AC4	Data is persisted and retrieved from PostgreSQL
AC5	Application builds with Maven and runs in Docker container
AC6	API documentation is available via Swagger/OpenAPI

6. Out of Scope

- Frontend/UI implementation
- Advanced reporting/analytics
- Multi-tenancy
- Payment processing

7. Milestones

Milestone	Description	Target Date
Project Setup	Repo, Spring Boot, Docker, DB	Week 1
Inventory CRUD	Product endpoints, tests	Week 2
Order CRUD	Order endpoints, tests	Week 3
JWT Security	Auth endpoints, JWT integration	Week 4
Pagination & Sorting	List endpoints, query params	Week 5
Containerization & Docs	Docker, Swagger, README	Week 6

8. Open Questions

- Will user registration be required, or only admin users?
- Should order status transitions be enforced (e.g., pending → shipped)?

End of Document