Product Requirements & Specification Document

Project Name

BookMart - Capstone E-commerce Backend

Description

BookMart is a simple e-commerce backend for a bookstore, built with Java and Spring Boot. It provides essential APIs for user authentication, book management, order processing, and error handling, with MySQL as the database. The system is designed for easy deployment and maintainability.

---

1. Goals & Objectives

Goal	Description
User Management	Register, authenticate, and authorize users
Book Catalog	CRUD operations for books
Order Processing	Place and manage orders
Error Handling	Consistent error responses
Deployment	Ready for deployment on standard cloud/VPS environments

---

2. Functional Requirements

2.1 User Authentication & Authorization

Requirement	Description
User Registration	Users can register with email & password
User Login	JWT-based authentication
Role-based Authorization	Roles: USER, ADMIN
Password Security	Passwords hashed and salted

2.2 Book Management

Requirement	Description
List Books	Publicly accessible book listing
View Book Details	Publicly accessible
Add/Edit/Delete Book	ADMIN only

2.3 Order Management

Requirement	Description
Place Order	Authenticated users can place orders

View Orders	Users view their orders; ADMIN views all
Order Status	Track status: PLACED, SHIPPED, DELIVERED

2.4 Error Handling

Requirement	Description
Consistent Error Responses	Standardized error format (JSON)
Input Validation	Validate all user inputs

2.5 Deployment

Requirement	Description
Environment Variables	Configurable DB credentials, JWT secret
Docker Support	Dockerfile for containerized deployment

---

3. Non-Functional Requirements

Category	Specification
Performance	API response < 500ms for standard operations
Security	JWT, password hashing, input validation
Scalability	Stateless backend, supports horizontal scaling
Maintainability	Clear code structure, RESTful conventions
Documentation	API documentation (Swagger/OpenAPI)

---

4. System Architecture

4.1 High-Level Diagram

```
[Client] <-> [Spring Boot REST API] <-> [MySQL Database]
```

4.2 Main Components

Component	Description
Authentication	JWT-based login, registration
Book Service	CRUD for books
Order Service	Order placement and management
User Service	User profile and role management
Error Handler	Global exception handling

5. Data Model Overview

Entity	Fields
User	id, email, password, role, created_at
Book	id, title, author, description, price, stock, created_at

Order	id, user_id, order_date, status, total_amount
OrderItem	id, order_id, book_id, quantity, price

---

6. API Endpoints (Summary)

Method	Endpoint	Description	Auth	Role
POST	/api/auth/register	Register new user	No	-
POST	/api/auth/login	User login (JWT)	No	-
GET	/api/books	List all books	No	-
GET	/api/books/{id}	Get book details	No	-
POST	/api/books	Add new book	Yes	ADMIN
PUT	/api/books/{id}	Update book	Yes	ADMIN
DELETE	/api/books/{id}	Delete book	Yes	ADMIN
POST	/api/orders	Place order	Yes	USER
GET	/api/orders	List user orders	Yes	USER
GET	/api/admin/orders	List all orders	Yes	ADMIN

---

7. Error Handling Specification

```
{
  "timestamp": "2024-01-01T12:00:00Z",
  "status": 400,
  "error": "Bad Request",
  "message": "Validation failed",
  "path": "/api/books"
}
```

---

8. Deployment & Environment

Aspect	Specification
Database	MySQL 8.x
Java Version	Java 17+
Build Tool	Maven/Gradle
Containerization	Dockerfile provided
Config	.env or application.properties for secrets

---

9. Out of Scope

- Payment processing
- Frontend/UI
- Advanced analytics or reporting
- Third-party integrations

---

10. Acceptance Criteria

- All endpoints function as specified
- Authentication and authorization enforced
- Error responses are consistent
- Application deploys via Docker
- API documentation is available

End of Document