E-COMMERCE APPLICATION

Create appropriate database architecture (Domain Driven Design) for an E-commerce store, showing proper understanding of model mapping. The following features should be included;

- Endpoints and view for Admin to Upload/add products, view products (in categories and as a single item), update products and delete products (CRUD) (4)
- Endpoints and view for customer for adding and removing product to a cart (2)
- Endpoints and view for customer to view products (in categories and as a single item) (2)
- Endpoint and view for customer Signup and login (2)
- Endpoint and view for admin signup and login (2)
- Endpoint for customers to like or save products they wish to purchase later (2)

An architectural diagram showing the User Journey should be included, showing all the modules all users of the application should be able to interact with. Your program should use Spring Boot MVC and also the codebase is to follow SOLID principles where necessary.

Tools

• Spring (servlet)
• Template Engine: Thymeleaf (jsp)
• JUnit/Mockito
• Spring Data JPA (jdbc)
• PostgreSQL (database)

User Stories

As a user, I should be able to
• Signup
• Login
• Like/unlike a product I wish to purchase later
• Add/remove a product to a cart
• View a product in a category and as a sole product

As an Admin, I should be able to
• Signup
• Login
• Add, delete, update, and view a product

Expectations

• Ensure your code is DRY.
• Basic OOP should be used.
• Apply various optimizations where necessary (e.g. pagination).
• Efficient SQL custom queries should be used where needed. – (Joins and Views)
• Exception Handling (Custom Exceptions) and Logging
• Custom response structure both on success and failure
• All parts of your code should be tested
• Database design should be normalized and entities should have necessary relationship mapping
• Add timestamp to your tables where necessary (for auditing purposes)

Assessment Criteria

1. Codebase is fully tested.
2. All SQL queries are efficient
3. Efficient table mappings
4. Defined response format
5. Codebase is clean and 100% DRY.