Design Patterns and Software Development Techniques in the E-commerce Application

Introduction

This document outlines the key design patterns and software development techniques utilized in the implementation of an e-commerce application built using Spring Boot, Java SDK version 17, and Jakarta EE. The application features functionalities such as order management, payment management, customer reviews, stock management, and delivery management. Understanding these patterns and techniques will help maintain and extend the application effectively.

Design Patterns

1.1. Model-View-Controller (MVC) Pattern

Description:

MVC is a design pattern that separates an application into three main logical components: Model, View, and Controller. Each component handles specific development aspects of the application.

Application:

Model: Represents the domain-specific data and business logic in the application. In this project, classes like Order, Payment, Review, Delivery, and User serve as models.

View: Represents the UI components. For a RESTful web service, views are typically JSON or XML responses, rendered by Spring's @ResponseBody annotations.

Controller: Handles user input and interactions, updating the model and view. Controllers such as OrderController, PaymentController, and others, handle HTTP requests and responses.

Example:

A computer screen shot of a program code

Description automatically generated

1.2. Service Layer Pattern

Description:

The Service Layer pattern defines an application boundary with a layer that encapsulates the business logic and ensures business rules are consistent across multiple channels.

Application:

Service interfaces and their implementations (e.g., OrderService and OrderServiceImpl) encapsulate the business logic to promote separation of concerns and reusability.

Example:

A screen shot of a computer program

Description automatically generated

1.3. Repository Pattern

Description:

The Repository pattern abstracts the data access layer, providing a collection-like interface for accessing domain objects.

Application:

Repository interfaces such as OrderRepository, PaymentRepository, etc., extend Spring Data JPA's JpaRepository, interacting with the database transparently.

Example:

A black screen with colorful text

Description automatically generated

1.4. Dependency Injection (DI) Pattern

Description:

Dependency Injection is a technique in which an object receives other objects that it depends on. DI allows for decoupled code and easier testing.

Application:

Spring's @Autowired annotation is used to inject dependencies, such as service classes into controllers and repositories into services.

Example:

A black screen with white text

Description automatically generated

2. Software Development Techniques

2.1. RESTful Web Services

Description:

Representational State Transfer (REST) is an architectural style for designing networked applications. RESTful services expose data and operations as resources through HTTP methods.

Application:

Controllers handle HTTP requests and return JSON responses. Methods are mapped to CRUD operations using HTTP methods (GET, POST, PUT, DELETE).

Example:

A computer screen shot of text

Description automatically generated

2.2. Unit Testing

Description:

Unit testing involves testing individual units or components of the software to ensure they work as expected.

Application:

Using JUnit and Mockito to create tests for controllers and services, verifying that responses and interactions are correct.

Example:

A screen shot of a computer code

Description automatically generated

2.3. Spring Boot

Description:

Spring Boot simplifies the development of production-ready applications by providing defaults for boilerplate configuration.

Application:

Uses Spring Boot for building the e-commerce application, taking advantage of Spring Boot's auto-configuration, dependency management, and embedded servlet container.

2.4. Persistence with JPA

Description:

Java Persistence API (JPA) is a specification for managing relational data in Java applications.

Application:

Annotates entity classes with JPA annotations to define table mappings and uses Spring Data JPA repositories for data access.

Example:

A screen shot of a computer program

Description automatically generated

Conclusion

The design patterns and software development techniques employed in this e-commerce application ensure scalability, maintainability, and ease of testing. By adhering to these patterns and techniques, we can develop robust and well-architected applications that meet modern software development standards.  
  
Overview

This code uses annotations from the Springdoc OpenAPI 3 library to document an endpoint in a Spring Boot web application. The annotations provide details about what the endpoint does, and the expected responses. Here's a detailed explanation of each part:

A screen shot of a computer code

Description automatically generated