SOFTWARE ENGINEERING

PUNJAB

افکار تازہ سے نمود

ASSIGNMENT-03(CEP)
REPORT

Submitted by:

Muhammad Arham – BSCE22007

Muhammad Moiz Ahmad – BSCE22029

Alishba Zahid – BSCE22035

Retail Store Management System UML Design Report

1. Introduction

This report presents the complete UML-based software design for the Retail Store Management System. The documentation is divided into two main sections: Structural Modeling and Dynamic Modeling. Each section contains diagrams and design considerations that help developers understand system behavior, architecture, and data flow.

2. Structural Modeling

This section focuses on the static architecture of the system.

2.1 Class Diagram

The class diagram defines the major system classes, including both abstract and concrete classes. It models the attributes, methods, and relationships between entities such as Admin, Product, Invoice, etc.

- Illustrates the static structure of the system.
- Consists of abstract and concrete classes.
- Incorporates design principles such as encapsulation, interface segregation, and use of design patterns.

Key Classes:

- User (Abstract)
- Admin, Cashier, Customer (Concrete, inherit from User)
- Product, Inventory, Order, Invoice, Report
- Service Classes: AuthenticationService, BackupService, DiscountService, LoyaltyProgramService, PurchaseReturnService

All services use interfaces for abstraction. For example:
- IAuthenticationService is implemented by AuthenticationService
- IDiscountApplier is implemented by DiscountService
Interfaces ensure loose coupling and testability.

2.2 Design Pattern Diagrams

Design patterns are implemented for maintainability and flexibility. The following patterns are used:
- Observer Pattern: For low stock alert notifications
- Iterator Pattern: To traverse collections such as reports
- Facade Pattern: Unified interfaces for services (e.g., AuthenticationService)
- Decorator Pattern: Applying promotions dynamically to invoices

3. Dynamic Modeling

This section focuses on system behavior and interaction over time.

3.1 Activity Diagrams

Activity diagrams describe workflows for Admin, Cashier, and Customer roles. It includes branching, parallel activities, and exception flows. Example: Inventory update, order processing, and login activity.

- Describes workflows for Admin, Cashier, and Customer.
- Each activity node maps directly to class methods or use cases.
- Branching conditions for different user roles and system decisions.

3.2 Sequence Diagrams

- Describes object interactions over time for key use cases.
- Use cases covered include:
  * User Login
  * Product Management
  * Order Processing
  * Invoice Generation
  * Loyalty Enrollment
  * Product Return
  * Report Generation

Features:

- Includes alt, opt, loop, and par where needed.
- Interfaces and service method calls shown explicitly.
- Consistent with the object and class diagrams.

3.3 State Diagrams

State diagrams show state transitions for key objects based on events and actions. Examples:
- Product: InStock → LowStock → OutOfStock
- Order: Created → Paid → Completed → Returned

3.4 Object Class Diagram

Object diagrams show instantiated objects at runtime to verify relationships and attribute values.
They demonstrate the actual structure of objects as represented during system execution.

- Shows runtime instances of objects and how they interact.
- Validates relationships defined in the class diagram.
- Demonstrates practical interaction patterns among :Admin, :Cashier, :Customer, and various service objects.

3.5 Use Case Diagram

Use case diagram shows interactions between actors (Admin, Cashier, Customer) and system functionality. Includes:
- Product management
- Purchase processing
- Stock tracking
- Loyalty enrollment
- Report generation

4. Folder Structure Summary

The complete design is organized as follows:

Folder: Structural Modeling

- Class Diagram
- Design Pattern Diagrams

Folder: Dynamic Modeling

- Use Case Diagram
- Sequence Diagrams
- State Diagrams
- Activity Diagrams
- Object Class Diagram

Folder: Architecture Pattern

-application_layer.png

-arichtecture pattern.png

-layered_architecture.png

-layered_architecture_Retail_Shop_Managment_System.png(Complete diagram of Architecture Pattern)

5. Consistency & Validation

Object-Class-Sequence Consistency

- All sequence diagrams are derived from the class diagram methods.
- Object interactions validate runtime instances and their behavior.
- Interfaces are used consistently across structure and behavior.

6. Conclusion

This comprehensive UML report provides a modular and scalable design for a Retail Store Management System. By separating structure from behavior and applying industry design patterns, the system is flexible, reusable, and maintainable.