

# **Assignment 2 – Object Design**

## **Long Chau Pharmacy Management System (LC-PMS)**

### **Semester 2, 2025**



**Swinburne University of Technology**  
*School of Software and Electrical Engineering*

**ASSIGNMENT AND PROJECT COVER SHEET**

---

Subject Code: SWE30003

Unit Title: Software Architectures and Design

Assignment Number and Title: 2, Object Design

Due Date: 4th July 2025

Tutorial Day and Time: \_Wednesday, 1 PM - 5PM\_\_\_\_

Project Group: \_12\_\_\_\_\_

Tutor: \_\_\_\_\_Dr. Duc Minh Le\_\_\_\_\_

---

**To be completed as this is a group assignment**

We declare that this is a group assignment and that no part of this submission has been copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part been written for us by another person.

ID Number	Name	Signature
104852250	Nguyễn Chí Cường	Cuong
104224985	Hồ Lê Nam	Nam
104999568	Phạm Quang Minh	Minh

---

Marker's Comments:

Total Mark: \_\_\_\_\_

---

Extension Certification:

This assignment has been given an extension and is now due on: \_\_\_\_\_

Signature of Convener: \_\_\_\_\_

## **Table of Contents**

1. Executive Summary
  2. Introduction
  3. Problems
  4. Outlook of Solution
  5. Assumptions
  6. Evidence of Problem Analysis
    - 6.1. Problem Analysis
    - 6.2. Simplifications
  7. Candidate Classes
  8. UML Diagram
  9. Justification
  10. CRC Cards
  11. Design Quality
  12. Pattern
  13. Bootstrap Process
  14. Verification
  15. References
- 

### **1. Executive Summary**

This document presents the object-oriented design for the Long Chau Pharmacy Management System (LC-PMS), based on the case study introduced in Assignment 1. The purpose is to establish a scalable and maintainable system design that follows the principles of Responsibility-Driven Design (RDD).

The report identifies candidate classes, their responsibilities, relationships, and bootstrap processes. Through CRC cards, the document details each class's purpose and interaction. Furthermore, design heuristics and common design patterns are applied to ensure a high-quality system architecture that meets business needs.

---

## 2. Introduction

The Long Chau Pharmacy Management System (LC-PMS) was introduced to address inefficiencies in managing pharmacy sales, customer prescriptions, inventory tracking, and employee operations. The current manual system results in errors, slow transaction processing, and poor tracking of medicines and customer history.

This document proposes a robust object-oriented design that improves inventory control, streamlines prescription management, and enhances the customer shopping experience. The system adheres to Responsibility-Driven Design principles and uses UML diagrams and CRC cards to structure a clear, modular solution.

---

## 3. Problems

The major problems LC-PMS aims to solve:

- **3.1. Inventory Management Issues:** Manual tracking leads to out-of-stock or expired medicines.
  - **3.2. Prescription Verification Delays:** Pharmacists lack a fast, systematic method for verifying prescriptions.
  - **3.3. Customer Relationship Management:** Customer records, loyalty tracking, and prescription history are poorly maintained.
  - **3.4. Employee Operation Management:** Inefficient employee shift and task assignment.
- 

## 4. Outlook of Solution

The solution involves a modular system that:

- Automates inventory tracking and expiry monitoring.
- Provides a fast prescription verification process.
- Manages customer profiles and loyalty points.
- Organizes employee scheduling and tasks.

- Ensures scalability and flexibility for future enhancements.

UML diagrams, CRC cards, and class justifications are used to visualize system components and promote clear, maintainable architecture.

---

## **5. Assumptions**

- Pharmacists and employees are trained to use digital systems.
  - The system operates on both web and local pharmacy devices.
  - A secure, real-time inventory database is available.
  - Customers can register via email or phone.
  - Prescription documents can be verified digitally.
  - Payment processing integrates with existing POS systems.
  - Each transaction generates a unique identifier.
  - Sensitive customer and prescription data are securely encrypted.
- 

## **6. Evidence of Problem Analysis**

### **6.1. Problem Analysis**

Manual operations at Long Chau Pharmacies have caused:

- Frequent stock discrepancies.
- Long customer wait times for prescription verification.
- Poor tracking of customer medication history.
- Employee scheduling conflicts.

The proposed LC-PMS system will address these with real-time inventory updates, faster prescription processing, and comprehensive customer and employee management modules.

## **6.2. Simplifications**

- Initial inventory system uses FIFO (First In, First Out) without complex warehouse optimization.
  - Basic prescription validation; advanced drug interaction checks will be implemented later.
  - Simple point-based loyalty system.
  - Manual employee verification; future versions may integrate biometric systems.
  - Basic reporting for sales and stock levels; detailed analytics can be added later.
- 

## **7. Candidate Classes**

### **Pharmacy Management Domain**

- Account
- AccountManager
- Manager
- User
- Pharmacist
- Medicines
- Prescription
- Product

### **Transaction Domain**

- Sale

- Discount
- Transaction

### Reporting Domain

- Report

### Data Holders

- Inventory

---

## 8. UML Diagram

*(To be inserted)*

---

## 9. Justification

- **User Management:** Classes like Account, Admin, Pharmacist, and User allow the system to manage authentication, role-based access, and distinct user behaviors.
  - **Inventory and Product Tracking:** Meds, Inventory, and Statistics are essential for real-time stock management, expiry tracking, and inventory analytics.
  - **Prescription and Transaction Management:** Prescription, Sale, Payment, Discount, and Product support the full sales and prescription verification workflow.
  - **Reporting:** The Report class provides key decision-making insights.
  - **Separation of Concerns:** Each class is assigned a focused responsibility to avoid creating “god classes” and to promote maintainability.
- 

## 10. CRC Cards

## 1. Account

<b>Class name:</b> Account <b>Parent class:</b> AccountManager	
<b>Brief Description:</b> Encapsulates essential login and permission data for any user in the system.	
Responsibilities	Collaborators
KNOW the login details, such as username (contact number) and password	User
KNOW the personal details and permission	User

## 2. AccountManager

<b>Class name:</b> AccountManager <b>Parent class:</b> N/A	
<b>Brief Description:</b> Oversees the creation, editing, and overall lifecycle of "Account" objects	
Responsibilities	Collaborators
DO create new accounts	Account
DO edit account profiles	Account
DO deactivate or suspend accounts if needed	Account

## 3. Manager (Admin)



<b>Class name:</b> Manager <b>Parent class:</b> Account	
<b>Brief Description:</b> Represent an account with highest authority in the system	
Responsibilities	Collaborators
DO manage, monitor and modify the system	Report, AccountManager
DO request business operation reports	Report
ENSURE user escalated issues are remedied	Report

#### 4. User

<b>Class name:</b> User <b>Parent class:</b> Account	
<b>Brief Description:</b> User represents a user who uses the platform as a service but has no authority to manage the platform	
Responsibilities	Collaborators
KNOW the user's history of using the platform service (medicine purchased)	Account
DO purchase products from our service.	Discount (discount can be 0 - no discounts)
ENSURE user has received the product	Product

#### 5. Pharmacist

<b>Class name:</b> Pharmacist <b>Parent class:</b> Account	
<b>Brief Description:</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
DO provide prescriptions for medicine	Prescription
DO request medicine	Inventory

## 6. Medicines

<b>Class name:</b> Medicines <b>Parent class:</b> Product	
<b>Brief Description:</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
ENSURE they're accompanied by their prescription (if there's one)	Product

## 7. Prescription

<b>Class name:</b> Prescription <b>Parent class:</b> Product	
<b>Brief Description:</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
ENSURE they're accompanied with their medicine (if they're required)	Product

## 8. Product

<b>Class name:</b> Product <b>Parent class:</b> N/A	
<b>Brief Description:</b> General representation of a sellable product (medicine or others)	
Responsibilities	Collaborators
ENSURE both types of items match	Prescription, Medicine
ENSURE to reach the right user	User

## 9. Sale

<b>Class name:</b> Sale <b>Parent class:</b> N/A	
<b>Brief Description:</b> Represents a product sale transaction.	
Responsibilities	Collaborators
KNOW how many transactions have been made	Product
KNOW how much has been made	Transaction

## 10. Transaction

<b>Class name:</b> Transaction <b>Parent class:</b> N/A	
<b>Brief Description:</b> Represents the payment for a sale.	
Responsibilities	Collaborators
DO store the made transaction	Sale

ENSURE/DO accept/decline the payment depending on the conditions	User
DO notify the pharmacist that a transaction has been made	Pharmacist

## 11. Discount

<b>Class name:</b> Discount <b>Parent class:</b> N/A	
<b>Brief Description:</b> Represents discounts applied to products or sales.	
<b>Responsibilities</b>	<b>Collaborators</b>
DO discount (if applicable)	Transaction
ENSURE discount is on the correct	Medicine

## 12. Report

<b>Class name:</b> Report <b>Parent class:</b> N/A	
<b>Brief Description:</b>	
<b>Responsibilities</b>	<b>Collaborators</b>
DO return the sale summary	Sale
DO return the inventory available items, and items that ran out	Inventory
ENSURE the report reaches the managers	Manager

### 13. Inventory

<b>Class name:</b> Inventory <b>Parent class:</b> N/A	
<b>Brief Description:</b> Data holder for product and medicine stock.	
<b>Responsibilities</b>	<b>Collaborators</b>
ENSURE the correct items are entered and leave the inventory	Medicines, Pharmacist
DO track stock levels	Medicines

---

### 11. Design Quality

- Single Responsibility Principle: Each class has one responsibility.
- Open/Closed Principle: New roles or pricing models can be added without altering existing code.
- Liskov Substitution Principle: Employees can be replaced with Pharmacists without breaking the system.
- Interface Segregation Principle: Calculators have minimal interfaces.
- Dependency Inversion Principle: Price and discount calculations depend on abstract interfaces.
- Law of Demeter: Classes only communicate with direct collaborators.
- Separation of Concerns: User, Inventory, Transaction, and Reporting are distinct modules.
- Encapsulation: Sensitive data is private with controlled access.

---

### 12. Pattern

- **Singleton Pattern:** Applied to AccountManager and Inventory to ensure centralized management without duplication.
  - **Observer Pattern:** Notifications can be sent when stock levels change or prescriptions are approved.
  - **Factory Pattern:** Used to instantiate specific user roles based on account type.
  - **Strategy Pattern:** Applied in Discount to handle multiple pricing models (e.g., seasonal promotions, loyalty discounts).
- 

### 13. Bootstrap Process

- **Admin** initializes the system by creating user accounts and assigning roles via AccountManager.
  - **Inventory** is populated with initial stock of Meds.
  - **Pharmacist** is linked to the Prescription module to handle verification.
  - **User** begins interaction: searching products, placing orders, uploading prescriptions.
  - **Sale** and Payment modules handle transactions.
  - **Statistics** and Report compile real-time system data for management.
- 

### 14. Verification

#### Account Registration and Login:

- User inputs credentials.
- System verify via AccountManager.
- Valid accounts → Login successful.
- Invalid credentials → Access denied.

#### Prescription Validation:

- Customer uploads prescription.
- Pharmacists review and either approve or reject it.
- System updates order status accordingly.

#### **Sales Process:**

- Customer selects products.
- Sale calculates total, applies Discount, processes via Payment.
- Receipt generated, stock updated in Inventory.

#### **Low Stock Alert:**

- Inventory automatically checks levels.
- If below threshold, Observer notifies Admin to restock.
- Real-time update visible in Report.

---

## **15. References**

- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Fowler, M. (2004). *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley.
- Martin, R. C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.
- Larman, C. (2002). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. Prentice Hall.