Avance proyecto final programación II

Laura Valentina Gómez Alzate

David Gómez Ramírez

Nicolás Muñoz Viveros



Profesor: Jorge Iván Triviño

Universidad del Quindío

Facultad de Ingeniería

Ingeniería de Sistemas y Computación

Programación II

Armenia, Quindío, 2024

Licencia Creative Commons

**Enunciado:**

Sistema de Gestión de Droguería

El objetivo del programa es implementar un sistema de gestión para una droguería que permita manejar productos, usuarios, ventas, devoluciones y el procesamiento de pagos con impuestos y descuentos. Este sistema debe seguir varios patrones de diseño como d, Decorator, Strategy, Chain of Responsibility y Command.

Funcionalidades Principales

1. Gestión de Productos:

- La droguería ofrece una variedad de productos categorizados en Medicamentos, Salud Sexual y Aseo Personal.

- Cada producto tiene atributos como código, nombre, marca, precio y fecha de vencimiento.

- Los productos pueden ser clonados para evitar la duplicación de objetos.

2. Gestión de Usuarios:

- Los usuarios son representados con atributos como nombre, ID, celular y correo electrónico.

- La droguería puede agregar y gestionar usuarios.

3. Sistema de Ventas:

- Las ventas consisten en productos y se asocian a un cliente específico.

- Se puede registrar ventas y calcular el precio total de las mismas.

- El sistema utiliza el patrón Strategy para aplicar diferentes estrategias de precios, como descuentos o precios normales.

4. Procesamiento de Pagos:

- Los pagos pueden ser decorados con impuestos y descuentos usando el patrón Decorator.

- La interfaz MetodoPago define el contrato para procesar los pagos.

- Clases concretas como PagoBase, ImpuestoDecorator y DescuentoDecorator implementan las diferentes formas de procesar los pagos.

5. Manejo de Devoluciones:

- Las devoluciones se manejan utilizando el patrón Chain of Responsibility.

- La interfaz DevolucionHandler define el contrato para manejar devoluciones.

- Clases concretas como DevolucionEmpleado y DevolucionSupervisor manejan las devoluciones en diferentes niveles.

6. Comandos de Venta:

- Las acciones sobre las ventas, como agregar o remover detalles, se manejan utilizando el patrón Command.

- Las clases DetalleVentaCommand y RemoveDetalleVentaCommand implementan los comandos específicos para manejar los detalles de la venta.

7. Patrón Singleton:

- La clase SingletonDrogueria asegura que solo una instancia de la droguería exista en el sistema.

Diagramas y Patrones de Diseño Utilizados

- Decorator: Para agregar funcionalidad adicional a los pagos, como impuestos y descuentos.

- Strategy: Para permitir diferentes estrategias de precios en las ventas.

- Chain of Responsibility: Para manejar el proceso de devoluciones en diferentes niveles de autoridad.

- Command: Para encapsular acciones sobre las ventas, permitiendo su ejecución y deshacer.

-Builder: para construir venta

-singleton: drogueria solo debe manejar una instancia de la misma

- prototype: crea un clon de la clase detalle venta

Clases y sus Relaciones

1. Producto: Representa los productos con atributos y métodos de clonación.

2. ListaProductos: Mantiene una lista de productos disponibles en la droguería.

3. Usuario: Representa a los usuarios del sistema.

4. Ventas: Gestiona las ventas, sus detalles y calcula el precio total.

5. MetodoPago: Interfaz que define el método para procesar pagos.

6. PagoBase: Clase base para los pagos.

7. PagoDecorator: Clase abstracta que extiende MetodoPago y permite decorar los pagos con funcionalidades adicionales.

8. ImpuestoDecorator y DescuentoDecorator: Clases concretas que añaden impuestos y descuentos a los pagos.

9. DevolucionHandler: Interfaz para manejar devoluciones.

10. DevolucionEmpleado y DevolucionSupervisor: Clases que manejan devoluciones en diferentes niveles.

11. Command: Interfaz para ejecutar y deshacer comandos sobre las ventas.

12. DetalleVentaCommand y RemoveDetalleVentaCommand: Clases concretas que implementan los comandos para manejar detalles de ventas.

13. SingletonDrogueria: Clase que implementa el patrón Singleton para asegurar una única instancia de la droguería.

Implementación

El programa deberá implementar todas las clases y métodos indicados en el diagrama UML, asegurándose de seguir las mejores prácticas de diseño de software y los patrones de diseño mencionados.

**Requisitos a cumplir**

* En la factura de compra aparecer fecha, nombre cliente, producto comprado, valor del producto
* Se debe poder hacer la operación de venta y registrarlas
* Guardar el usuario
* hacer operacion crud