Trabajo practico final de Laboratorio III

Supermercado Market



Barcala, Andrea

Castillo, Maximiliano

AÑO 2023

**Resumen del Proyecto - Supermercado**

El programa desarrollado es un sistema de gestión para un supermercado que permite a los clientes realizar compras de productos; y al encargado realizar tareas relacionadas con la administración, ventas, y operaciones del supermercado.

El sistema consta de varios menús que brindan diferentes opciones a los usuarios, como clientes, y encargado. A continuación, se resume la funcionalidad principal de cada uno de los menús:

Menú Principal: Permite al usuario seleccionar entre las opciones de Cliente, Empleado o Salir del programa.

Menú Cliente: Los clientes pueden registrarse en el supermercado e ingresar al sistema.

Menú Ingresar Cliente: Proporciona opciones específicas para los clientes que han iniciado sesión, como realizar compras, modificar su información personal, ver su historial de compras y solicitar la baja de la cuenta.

Menú Modificar Información Personal: Permite al cliente realizar cambios en su nombre de usuario, contraseña, nombre, apellido, domicilio y DNI.

Menú Empleado: Los empleados tienen acceso a funciones administrativas y de ventas, como ver la lista de clientes registrados, ver las facturas generadas, ver los productos disponibles, comprar productos del supermercado, agregar nuevos productos, eliminar productos del inventario, buscar información de clientes, dar de baja o alta a un cliente y ajustar los precios de los productos.

El sistema se ha desarrollado utilizando programación orientada a objetos en Java y utiliza la entrada del usuario a través de la consola. Proporciona una interfaz de menú intuitiva que permite a los usuarios realizar diversas operaciones dentro del supermercado.

Este sistema de gestión de supermercado automatiza y simplifica muchas tareas relacionadas con la administración y ventas, lo que mejora la eficiencia y la experiencia del cliente. Además, brinda un control al encargado para supervisar las operaciones y tomar decisiones.

**Informe Técnico - Aplicación de Supermercado**

**Descripción del Sistema**

La aplicación de Supermercado es un sistema de gestión que permite a los clientes realizar compras de productos, y al encargado administrar el inventario, supervisar y controlar las operaciones del supermercado.

**Clases y Relaciones**

**Clase Abstracta Usuario**

La clase abstracta **Usuario** representa a un usuario genérico del sistema. Tiene los atributos **usuario** y **contraseña**, y define un método abstracto **comprar()**. Las dos clases hijas de **Usuario** son **Cliente** y **Encargado**.

**Clase Cliente**

La clase **Cliente** representa a un cliente del supermercado. Implementa la interfaz **Oferta**, lo que implica que debe proporcionar una implementación para el método **descuento()**. Los atributos de la clase incluyen un enum de **Genero** (FEMENINO y MASCULINO), **nombre**, **dni**, **apellido**, **domicilio** y **cantidadDeComprasRealizadas**.

Los métodos más importantes son:

* **comprar()**: Implementación del método abstracto heredado de la clase **Usuario**.
* **descuento()**: Implementación del método de la interfaz **Oferta** que calcula el descuento para el cliente una vez superadas las 5 compras.

**Clase Encargado**

La clase **Encargado** representa al encargado del supermercado. Hereda de la clase **Usuario**. No tiene atributos adicionales, pero hereda el método **comprar()** de la clase padre, para realizar la suba del stock de los productos que agota el cliente.

**Clase Factura**

La clase **Factura** representa una factura generada por el cliente para una compra en el supermercado. Tiene los atributos **usuario** (cliente o empleado asociado), **productos** (un **HashMap** que mapea el id del producto a un objeto **Producto**), **id**, **fecha**, **metodoDePago** (enum con opciones EFECTIVO, CREDITO, DEBITO) y **precioFinal**.

Los métodos más importantes son:

* **agregarProductoFactura()**: Permite agregar un producto a la factura.
* **elegirMetodoDePago()**: Permite seleccionar el método de pago para la factura.
* **obtenerPrecioFinal()**: Calcula el precio final de la factura.

**Clase Producto**

La clase **Producto** representa un producto disponible en el supermercado. Tiene los atributos **nombre**, **precio**, **stock** e **id**. No tiene métodos adicionales.

**Clase Supermercado**

La clase **Supermercado** es la clase principal que contiene y gestiona todas las listas y operaciones del supermercado. Tiene los atributos **clientes** (lista de objetos **Cliente**), **productos** (lista de objetos **Producto**), **facturas** (lista de objetos **Factura**) y **encargado**.

Algunos de los métodos más importantes son:

* **agregarCliente()**: Agrega un cliente a la lista de clientes del supermercado.
* **bajaCliente ()**: Pone a un cliente en inactivo en la lista de clientes del supermercado.
* **buscarCliente ()**: Busca un cliente en la lista de clientes por su nombre de usuario.
* **agregarProducto()**: Agrega stock de un producto de la lista de productos disponibles en el supermercado.
* **eliminarProducto()**: Elimina un producto de la lista de productos disponibles en el supermercado.
* **agregarFactura()**: Agrega una factura a la lista de facturas del supermercado.
* **comprarUnProductoNuevo():** Agrega un producto nuevo a la lista de productos.
* **ingresarEncargado():** Permite al encargado ingresar al sistema.
* **ingresarCliente():** Permite al cliente ingresar al sistema.
* Todos los métodos relacionados a ver las listas.
* Todos los métodos relacionados a las modificaciones del cliente.

Estos son solo algunos de los métodos que tiene la clase **Supermercado**.

**Decisiones de Diseño**

* Se utilizó la herencia para modelar la relación entre las clases **Cliente**, **Empleado** y **Encargado** con la clase abstracta **Usuario**, ya que comparten algunos atributos y comportamientos comunes.
* Se implementó la interfaz **Oferta** en la clase **Cliente** para permitir que los clientes tengan descuentos especiales en sus compras.
* Se utilizó una estructura de datos **HashMap** en la clase **Factura** para almacenar los productos comprados, utilizando el ID del producto como clave.
* Se creó la clase **Leer** con métodos estáticos para facilitar la lectura de datos desde la entrada estándar en varias clases, evitando la necesidad de crear múltiples instancias de **Scanner**.

Estas decisiones de diseño fueron tomadas en base a los requerimientos y necesidades del sistema, buscando una estructura coherente y modular que facilite la implementación y mantenimiento del código.

**Toma de decisiones para la persistencia de datos**

La clase Supermercado es la clase principal que contiene y gestiona todas las listas y operaciones del supermercado, estas son las que guardamos en el archivo.

1. Persistencia de datos con Jackson: Se utilizó la librería Jackson para lograr la persistencia de datos en un archivo JSON. Esto permitió guardar y cargar la información del supermercado de manera confiable. Al utilizar la librería Jackson, fue necesario agregar las dependencias correspondientes en el archivo de configuración del proyecto y manejar las excepciones relacionadas con la lectura y escritura de archivos.
2. Tratamiento de fechas con librería específica: Para manejar el atributo LocalDate en la clase Factura, fue necesario implementar una librería específica. Esto se debió a que la librería Jackson no proporciona soporte directo para la serialización y deserialización de objetos LocalDate. Se seleccionó una librería adecuada que permitiera convertir las fechas en formato LocalDate a un formato compatible con JSON y viceversa.
3. Uso de anotaciones Jackson para la deserialización: Al tener dos hijos (Cliente y Empleado) al deserializar el objeto Usuario, fue necesario agregar un atributo "tipo" a los hijos para que el archivo pueda ser utilizado correctamente. Se utilizó la anotación @JsonTypeInfo para indicar el uso de un nombre para identificar el tipo de objeto y @JsonSubTypes para asignar un nombre a cada subtipo de Usuario. Esto permitió que Jackson pudiera deserializar correctamente los objetos Cliente y Empleado.

La combinación de la librería Jackson para la persistencia de datos en formato JSON, la implementación de una librería específica para el tratamiento de fechas LocalDate y el uso de herencia entre las clases Cliente y Empleado proporcionó una solución completa y eficiente para el funcionamiento del supermercado.