Trabajo de Campo - Programación Orientada a Objetos

Tema: Modelado de una Ferretería en Java

# Integrantes del grupo:

1. Cedric Pieer Pariona Miranda

2. Fernando José Valdivia Seminario

3. Gerald Ameir Galindo Ruiz

4. Pool Humberto Osores Monroy

5. Ricardo Andre Asorza Cantos

# Introducción

El presente trabajo de campo tiene como finalidad aplicar los principios de la Programación Orientada a Objetos (POO) en el modelado de un negocio: una ferretería. Se identificarán las clases principales que intervienen en su funcionamiento, sus atributos y métodos, y se presentará una implementación en Java.

# Objetivos

1. Identificar los objetos del negocio ferretería.

2. Definir atributos y métodos de cada clase.

3. Representar el modelo mediante UML.

4. Implementar el modelo en Java.

5. Fomentar el trabajo colaborativo con control de versiones en GitHub.

# Metodología

Se eligió como caso de estudio una ferretería de barrio. Se analizaron los elementos más relevantes para su funcionamiento: productos, clientes, empleados, ventas y la propia ferretería. Estos se modelaron como clases en Java. El trabajo se dividió entre los cinco integrantes del grupo para asegurar una participación equitativa.

# División del trabajo por integrante

1. Cedric Pieer Pariona Miranda: Clase Producto.java

2. Fernando José Valdivia Seminario: Clase Cliente.java

3. Gerald Ameir Galindo Ruiz: Clase Empleado.java

4. Pool Humberto Osores Monroy: Clase Venta.java

5. Ricardo Andre Asorza Cantos: Clase Ferreteria.java y Main.java

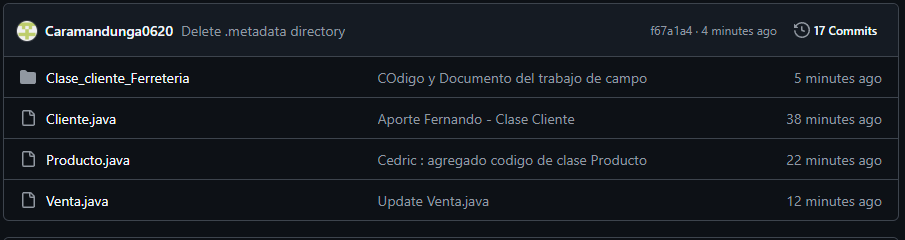
# Implementación en Java

// Producto.java  
class Producto {  
 String nombre;  
 double precio;  
 int stock;  
  
 Producto(String nombre, double precio, int stock) {  
 this.nombre = nombre;  
 this.precio = precio;  
 this.stock = stock;  
 }  
  
 void vender(int cantidad) {  
 if (cantidad <= stock) {  
 stock -= cantidad;  
 System.out.println("Venta realizada: " + cantidad + " " + nombre);  
 } else {  
 System.out.println("Stock insuficiente de " + nombre);  
 }  
 }  
  
 void reponer(int cantidad) {  
 stock += cantidad;  
 System.out.println("Se repuso " + cantidad + " unidades de " + nombre);  
 }  
}  
  
// Cliente.java  
class Cliente {  
 String nombre;  
 String dni;  
  
 Cliente(String nombre, String dni) {  
 this.nombre = nombre;  
 this.dni = dni;  
 }  
  
 void comprar(Producto producto, int cantidad) {  
 System.out.println(nombre + " compró " + cantidad + " " + producto.nombre);  
 producto.vender(cantidad);  
 }  
}  
  
// Empleado.java  
class Empleado {  
 String nombre;  
 String cargo;  
  
 Empleado(String nombre, String cargo) {  
 this.nombre = nombre;  
 this.cargo = cargo;  
 }  
  
 void atenderCliente(Cliente cliente) {  
 System.out.println(nombre + " atendió a " + cliente.nombre);  
 }  
}  
  
// Venta.java  
class Venta {  
 Cliente cliente;  
 Producto producto;  
 int cantidad;  
  
 Venta(Cliente cliente, Producto producto, int cantidad) {  
 this.cliente = cliente;  
 this.producto = producto;  
 this.cantidad = cantidad;  
 }  
  
 void registrarVenta() {  
 double monto = cantidad \* producto.precio;  
 System.out.println("Venta registrada: " + cliente.nombre + " compró " +  
 cantidad + " " + producto.nombre + " por S/" + monto);  
 }  
}  
  
// Ferreteria.java y Main.java  
class Ferreteria {  
 String nombre;  
  
 Ferreteria(String nombre) {  
 this.nombre = nombre;  
 }  
  
 void mostrarInfo() {  
 System.out.println("Bienvenido a la ferretería " + nombre);  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 Ferreteria ferreteria = new Ferreteria("Ferretería Central");  
 ferreteria.mostrarInfo();  
  
 Producto martillo = new Producto("Martillo", 25.0, 10);  
 Cliente cliente1 = new Cliente("Fernando", "12345678");  
 Empleado empleado1 = new Empleado("Ricardo", "Cajero");  
  
 empleado1.atenderCliente(cliente1);  
 cliente1.comprar(martillo, 2);  
  
 Venta venta1 = new Venta(cliente1, martillo, 2);  
 venta1.registrarVenta();  
 }  
}

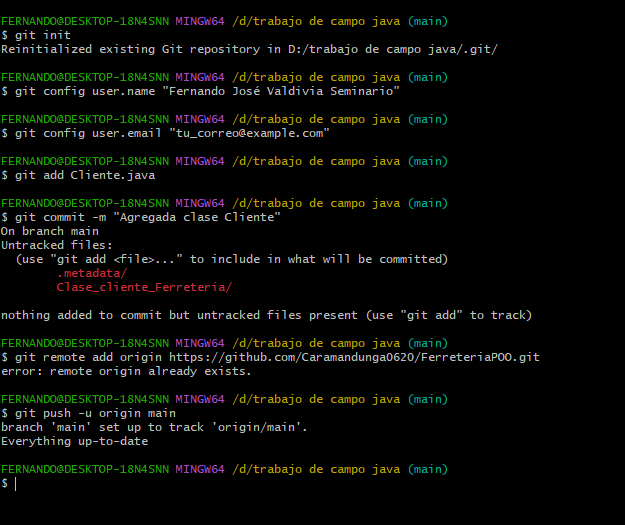
# Conclusiones

La modelación de la ferretería con POO en Java permite representar un negocio de manera estructurada. Cada integrante desarrolló una parte del proyecto, lo que refuerza el trabajo en equipo. Además, el uso de GitHub asegura control de versiones y colaboración en línea.

# Evidencia de GitHub



Fernando:



Cedric:  
