

## **Explicación acerca de la tercera Fase de Encuentras de todo**

**Andrés Estiven Gamboa Mena**

**Darwin Mosquera Hinestroza**

**ingeniería de sistemas**

**Martha Jiménez**

**Septiembre 2025**

## Principios solid utilizados

La aplicación de los principios SOLID es crucial para el desarrollo de un E-commerce escalable y fácil de mantener. Tu estructura de paquetes, con una clara separación entre Models y Controllers, ya sienta una base sólida al cumplir con el Principio de Responsabilidad Única (SRP). Las entidades (Productos, Factura) se encargan solo de la estructura de datos, mientras que los controladores (FacturaController) manejan la comunicación. Sin embargo, para cumplir explícitamente el SRP y aislar la lógica de negocio, es imperativo crear una capa de Servicio, que se encargue exclusivamente de las tareas complejas, como el cálculo de totales, la validación de *stock* y la gestión de la persistencia de Hibernate, liberando así a los controladores para que solo manejen la petición y la respuesta.

Adicionalmente, el Principio Abierto/Cerrado (OCP) debe guiar el diseño de la lógica para permitir futuras expansiones sin alterar el código que ya está funcionando. Esto significa que si introduces nuevas funcionalidades —como un nuevo método de pago o una tarifa de envío basada en reglas— debes implementar estas características mediante la creación de nuevas clases que cumplan una Interfaz común, en lugar de modificar con sentencias *if/else* una clase central de tu negocio. Al aplicar estos principios, tu sistema se vuelve más flexible ante cambios, facilitando la adición de módulos y asegurando que las modificaciones en un área no introduzcan errores inesperados en otras partes del código.

## Funcionalidad

### Descripción General del E-commerce

Este E-commerce está diseñado como un **sistema de ventas transaccional** centrado en la **precisión de datos** y la **gestión robusta de pedidos**. Utiliza una arquitectura orientada a la persistencia a través de **Hibernate/JPA**, lo que asegura que todas las operaciones de negocio—desde la consulta de productos hasta la finalización de una factura—sean consistentes y atómicas.

### Funcionalidades Clave y Modelo de Datos

#### 1. Catálogo de Productos Detallado:

- La entidad **Productos** es el núcleo, registrando no solo la descripción y el precio, sino también el **stock** disponible, el **estado** (activo/inactivo), y la **Categoría** (permitiendo filtros eficientes).

## 2. Procesamiento Atómico de Pedidos:

- El proceso de compra se formaliza en la entidad **Factura**, la cual se relaciona con un **Usuario** y una lista dinámica de **DetalleFactura** (la línea de pedido).
- La entidad **DetalleFactura** es fundamental, ya que registra la **cantidad** comprada de un producto específico, permitiendo el cálculo preciso del **subtotal** por artículo.

## 3. Control de Inventario en Tiempo Real:

- La lógica de negocio reside en la capa de **Servicio/Controlador**, que implementa un mecanismo de **control de stock**. Al procesar cada **DetalleFactura**, se valida y se **resta** la cantidad comprada del inventario de **Productos**. Esta operación se maneja dentro de una única **transacción** para garantizar que, si el stock falla para un solo artículo, toda la compra se revierte (rollback).

En esencia, el E-commerce es un sistema enfocado en la **confiabilidad transaccional**, construido con una separación clara de responsabilidades entre el modelo de datos (Models) y la lógica de negocio, lo que facilita el mantenimiento y la expansión futura de funcionalidades.

## Funcionalidad de la API externa de hibernate.

Hibernate tiene muchas funcionalidades, entre ellas posee la capacidad de identificar cual es una entidad, tabla, o algo relacionado a la base de datos, y su funcionalidad por medio de sesiones le permite abrirlas de manera automática para realizar una operación, ya sea para guardar (persist/save), actualizar, (update), eliminar(delete), o mostrar (get).

## Descripción de lo que es hibernate

Hibernate es un Framework de Mapeo Objeto-Relacional (ORM) para Java. Su propósito principal es eliminar la necesidad de escribir código SQL repetitivo al actuar como un puente entre tus objetos Java (tus entidades Factura, Productos, etc.) y las tablas de tu base de datos relacional.

La API de Hibernate se consume a menudo a través de la Java Persistence API (JPA), que es el estándar oficial de Java para persistencia. JPA actúa como una capa de abstracción, lo que significa que puedes usar las interfaces de JPA (EntityManager) mientras Hibernate trabaja como la implementación detrás de escena.