**REQUERIMIENTOS NO FUNCIONALES**

**PRESENTADO POR**

**LUISA D. MORALES HURTADO**

**ANA SOFIA PALECHOR GOMEZ**

**ANGIE YULIANA NARVAEZ ROSERO**

**DOCENTE**

**MARIA ISABEL BASTIDAS**

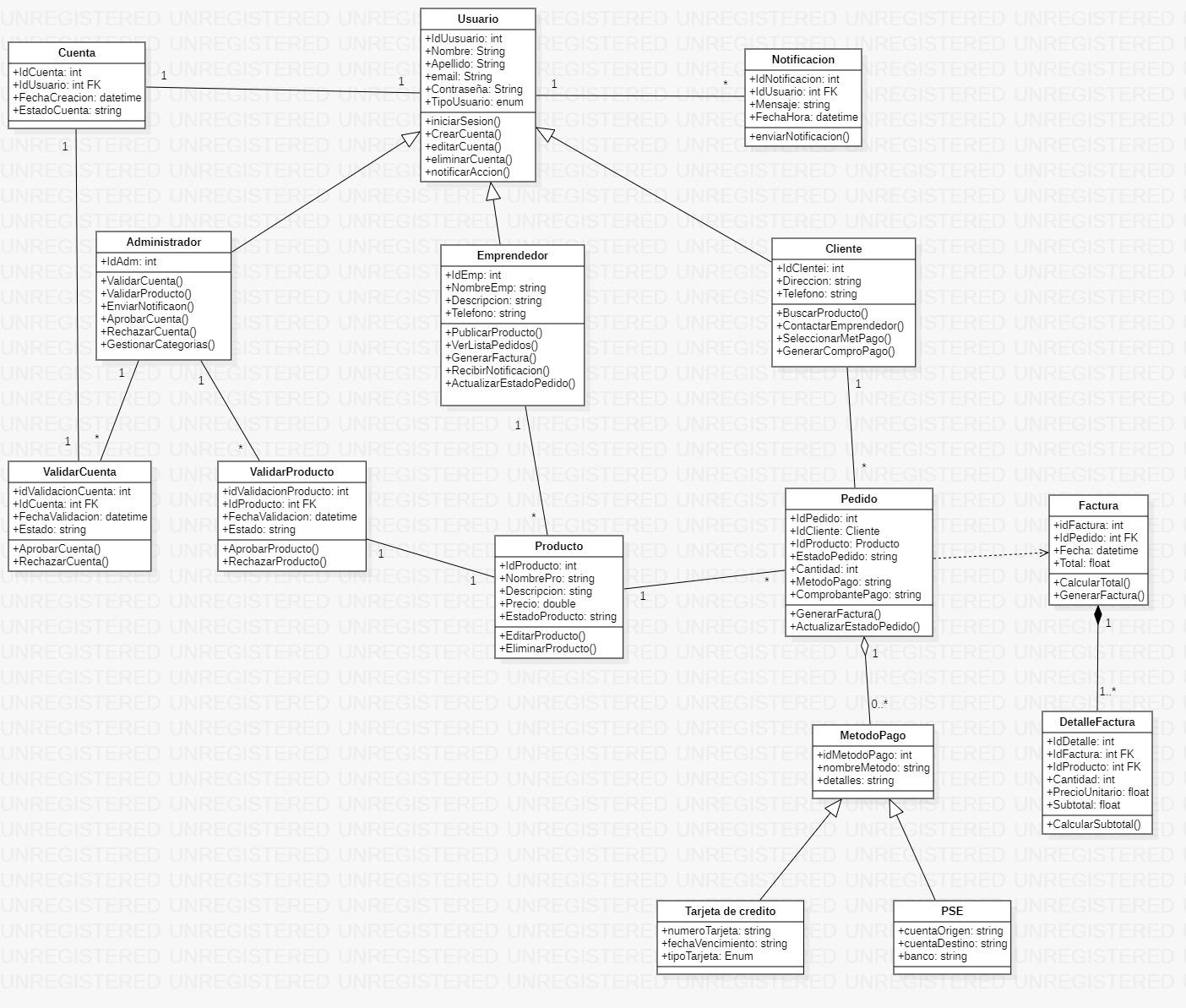


**INSTITUCIÓN UNIVERSITARIA COLEGIO MAYOR DEL CAUCA**

**INGENIERÍA INFORMÁTICA**

**POPAYÁN-CAUCA**

**2025**



Requerimientos NF

* Rendimiento: El sistema tendrá que responder en un tiempo máximo de 4 segundos para consultas, como la visualización de productos y pedidos, se implementará en la visualización de productos y pedidos
* Seguridad:El sistema tendrá que almacenar las contraseñas de los usuarios de forma encriptada, se verá reflejado en el login.

Garantizará la validación de roles (administrador, cliente, emprendedor) para evitar accesos no autorizados, se verá reflejado en el login.

* Usabilidad: El sistema ofrecerá una interfaz clara que permita al usuario navegar fácilmente según su rol.

Se mostrará mensajes de error cuando se ingresen datos incorrectos o se presenten fallos. esto se verá reflejado en los formularios

* Mantenibilidad: El sistema estará estructurado en módulos independientes que faciliten futuras modificaciones y actualizaciones. se verá reflejado con responsabilidades diferentes en cada una de las carpetas.
* Disponibilidad: El sistema deberá mostrar una página de error personalizada en caso de fallas. se verá reflejado en las funcionalidades principales de la página.
* Compatibilidad: El sistema deberá ser accesible desde los principales navegadores (Chrome,Opera, Edge).

**Principios de diseño**

Patrones Creacionales

• Factory Method

En mi proyecto se aplica el Factory Method en la creación de usuarios.

Tenemos una clase general Usuario, de la cual heredan Administrador, Cliente y

Emprendedor. Cada uno tiene responsabilidades y comportamientos diferentes dentro de

la plataforma.

El Factory Method permite que, cuando se registra un usuario, no sea necesario

especificar en el código qué tipo concreto se va a crear, sino que el sistema se encargue

de instanciar la subclase correcta según el caso.

Esto aporta desacoplamiento, ya que si mañana agregamos un nuevo tipo de usuario (por

ejemplo, un “Repartidor”), no habría que modificar el código base del sistema.

Patrones Estructurales

• Composite

El patrón Composite lo aplicamos en la relación entre Pedido y Producto.

Un pedido puede contener varios productos, pero a la vez cada producto puede tratarse de

manera individual.

De esta forma:

Producto actúa como hoja (unidad simple).

Pedido actúa como el compuesto que agrupa varios productos.

Esto nos permite recorrer un pedido, calcular totales o aplicar operaciones sin diferenciar

entre un solo producto o varios.