**Resumen del Proyecto Perfulandia SPA**

**1. Introducción**

Perfulandia SPA es un sistema web para la gestión de una tienda de perfumes que incluye módulos para productos, clientes, carrito de compras, ventas, cupones y usuarios con roles. Está desarrollado con Spring Boot, utiliza MySQL para la base de datos y cuenta con seguridad mediante Spring Security.

**2. Modelo de Datos y Tablas Principales**

**Tabla Producto**

* Guarda la información de los perfumes disponibles.
* Campos clave: id, nombre, marca, fragancia, capacidad, precio, categoria, stock.
* Un producto puede estar en múltiples carritos o ventas.

**Tabla Cliente**

* Registra los usuarios clientes del sistema.
* Campos: id, nombre, correo, contrasena, direccion, telefono, fechaRegistro.
* Un cliente puede tener un carrito activo y varias ventas.

**Tabla Carrito**

* Representa los productos que un cliente ha agregado para comprar.
* Campos: id, clienteId (relacionado con Cliente), productoId (relacionado con Producto), cantidad.
* Cada registro indica un producto específico en el carrito de un cliente.

**Tabla Venta**

* Registra las ventas realizadas.
* Campos importantes: id, clienteId, total, fechaVenta, etc.
* Relaciona productos con cantidades mediante tablas o listas internas (según implementación).
* Guarda el historial de compras de cada cliente.

**Tabla Cupon**

* Permite aplicar descuentos a las ventas.
* Campos: id, codigo, descripcion, porcentajeDescuento, fechaExpiracion.
* Puede aplicarse a una venta para modificar el total.

**Tabla Usuario**

* Gestiona los usuarios del sistema con roles (ADMIN, CLIENTE, etc.).
* Campos: id, nombre, correo, contrasena, rol.
* Controla permisos y accesos a funcionalidades.

**3. Relaciones Principales**

* **Cliente-Carrito**: Uno a muchos. Un cliente puede tener múltiples productos en su carrito.
* **Carrito-Producto**: Muchos a uno. Cada carrito referencia un producto.
* **Cliente-Venta**: Uno a muchos. Un cliente puede tener varias ventas.
* **Venta-Producto**: Muchos a muchos, modelado mediante una lista de productos y cantidades.
* **Venta-Cupon**: Uno a uno o uno a muchos, según si una venta puede usar múltiples cupones (normalmente uno).
* **Usuario**: Controla acceso y roles, no directamente relacionada a ventas o productos pero fundamental para seguridad.

**4. Lógica del Sistema**

**Gestión de Productos**

* CRUD de productos: creación, actualización, listado, búsqueda por categoría, marca, fragancia o nombre.
* Control de stock actualizado según ventas o devoluciones.

**Gestión de Carrito**

* Los clientes agregan productos con cantidades al carrito.
* Pueden ver, eliminar o actualizar el carrito.
* El carrito sirve para preparar la venta.

**Gestión de Ventas**

* Se registra la venta formal tomando los productos del carrito.
* Calcula totales y aplica descuentos de cupones si corresponde.
* Guarda fecha y estado de la venta.
* Permite procesar devoluciones con motivo.

**Gestión de Cupones**

* Aplicación y validación de códigos de descuento.
* Modifica el total de la venta en porcentaje o monto fijo.

**Seguridad**

* Control de acceso mediante Spring Security.
* Roles con diferentes permisos (ADMIN puede gestionar usuarios, productos; CLIENTE puede comprar y ver su historial).
* Endpoints protegidos y autenticación con usuario y contraseña.

**5. Uso y Flujo General**

1. **Cliente** se registra o inicia sesión.
2. Navega productos, busca o filtra según preferencia.
3. Agrega productos al **Carrito**.
4. Confirma compra, creando una **Venta**.
5. Puede aplicar un **Cupón** para descuento.
6. Recibe confirmación de compra.
7. Puede consultar historial de ventas o hacer devoluciones.
8. Administradores gestionan productos, usuarios y cupones desde panel privado.

**6. Consideraciones Técnicas**

* Backend desarrollado con Java y Spring Boot.
* Base de datos MySQL con JPA/Hibernate para gestión ORM.
* Seguridad gestionada con Spring Security.
* API REST con endpoints para cada entidad principal.
* Uso de DTOs para enviar datos específicos, como detalles del carrito.
* Validaciones en servicios para mantener integridad de datos.