# **Documento de Diseño de la Aplicación de Gestión de Productos**

## **1. Descripción General**

La aplicación permite gestionar una lista de productos mediante un archivo XML. Los usuarios pueden agregar, buscar, modificar, eliminar productos y exportar la lista a un archivo CSV. La aplicación utiliza Java y hace uso de las bibliotecas estándar para manejar XML y CSV.

## **2. Estructura del Proyecto**

### **Clases Principales**

1. **Productos**: Esta clase contiene el método main y la lógica para manejar la interacción con el usuario. Permite realizar las operaciones sobre los productos y manejar la entrada y salida de datos.

### **Métodos Clave**

* agregarProducto(Scanner scanner): Permite al usuario agregar un nuevo producto al archivo XML.
* buscarProducto(Scanner scanner): Busca un producto por su nombre e imprime sus detalles.
* modificarProducto(Scanner scanner): Modifica el precio o la cantidad de un producto existente.
* eliminarProducto(Scanner scanner): Elimina un producto del archivo XML.
* exportarACSV(): Exporta la lista de productos a un archivo CSV.
* cargarXML(): Carga el archivo XML y crea uno nuevo si no existe.
* guardarXML(Document doc): Guarda el documento XML actualizado.
* crearElemento(Document doc, String nombre, String valor): Crea un nuevo elemento XML.

## 

## **3. Interacción con el Fichero XML**

### **Estructura del Archivo XML**

El archivo XML tiene la siguiente estructura

<productos>

<producto>

<nombre>Nombre del producto</nombre>

<precio>19.99</precio>

<cantidad>50</cantidad>

</producto>

<!-- Más productos -->

</productos>

### **Carga y Guardado**

* **Carga**: Al iniciar la aplicación, se intenta cargar el archivo productos.xml. Si el archivo no existe, se crea uno nuevo con la estructura inicial.
* **Guardado**: Cada vez que se realiza una operación que modifica la lista de productos, se guarda el documento XML actualizado. Esto asegura que los cambios se persistan.

## **4. Decisiones de Diseño**

* **Uso de XML**: Se eligió XML como formato de almacenamiento por su capacidad de representar estructuras jerárquicas y su facilidad para ser manipulado en Java.
* **Excepciones**: Se maneja adecuadamente cualquier excepción que pueda surgir durante las operaciones de entrada/salida, proporcionando mensajes de error claros al usuario.
* **Interfaz de Usuario Simple**: La interfaz de línea de comandos es sencilla y permite al usuario interactuar de manera efectiva con la aplicación sin complicaciones adicionales.

## **5. Manejo de Excepciones**

El código incluye bloques try-catch para manejar excepciones al cargar, guardar y manipular el archivo XML. Se proporcionan mensajes de error que indican qué salió mal, lo que ayuda en la depuración y mejora la experiencia del usuario.

## **6. Futuras Mejoras**

* **Validaciones**: Incluir validaciones de entrada más robustas para asegurar que los datos introducidos por el usuario sean correctos.
* **Interfaz Gráfica**: Desarrollar una interfaz gráfica de usuario (GUI) para hacer la aplicación más accesible y fácil de usar.