Manual Técnico del Sistema de Gestión de Productos

1. Descripción General

El sistema está diseñado para gestionar productos en una tienda o inventario. Permite agregar, buscar, eliminar productos, registrar ventas y mantener un historial de acciones mediante bitácoras. Está desarrollado en **Java** usando colecciones dinámicas (`Vector`) y manejo de archivos para persistencia de datos.

---

2. Objetivo

Proporcionar una herramienta que permita gestionar de manera eficiente un inventario de productos y registrar las ventas realizadas, manteniendo un historial de acciones realizadas por el usuario.

---

3. Requerimientos

- **Software**:
  - Java JDK 11 o superior
  - IDE opcional: NetBeans, Eclipse, IntelliJ
- **Archivos**:
  - `bitacora.txt` → almacena el historial de acciones.
  - `ventas.txt` → almacena el registro de ventas.

---

4. Estructura del Código

4.1 Clase `Personaje`

Define la estructura de un producto en el inventario:

```
static class Personaje {
    int codUnico;
    String nombre;
    String categoria;
    int precio;
    int stock;
```

```
    public Personaje(int codUnico, String nombre, String categoria, int
precio, int stock) {
        this.codUnico = codUnico;
        this.nombre = nombre;
        this.categoria = categoria;
        this.precio= precio;
        this.stock = stock;
    }
}
```

- **codUnico**: identificador único del producto.
- **nombre**: nombre del producto.
- **categoria**: categoría del producto (camisas, pantalones, accesorios, etc).
- **precio**: precio unitario del producto.
- **stock**: cantidad disponible en inventario.

4.2 Vectores principales

```
private static Vector<Personaje> personajes = new Vector<>();
private static Vector<String> bitacoraActual = new Vector<>();
```

- `personajes`: almacena todos los productos registrados.
- `bitacoraActual`: almacena las acciones realizadas durante la sesión.

---

5. Funcionalidades Principales

5.1 Agregar Producto

Permite registrar un nuevo producto en el inventario, validando precio y stock positivos.

```
public static void agregarproducto() {
    int nuevocodUnico = personajes.size() + 1;
    System.out.println("Nombre del Producto: ");
    String producto = scanner.nextLine().trim();
    System.out.println("Categoria: ");
    String categoria = scanner.nextLine();
    System.out.println("Ingrese el precio: ");
    int precio = Integer.parseInt(scanner.nextLine());
    System.out.println("Ingrese la cantidad de stock: ");
    int stock = Integer.parseInt(scanner.nextLine());

    Personaje nuevoPersonaje = new Personaje(nuevocodUnico, producto,
categoria, precio,stock);
    personajes.add(nuevoPersonaje);

    System.out.println("\nProducto agregado exitosamente!");
```

```
    registrarBitacora("Agregar Producto", true);
}
```

5.2 Buscar Producto

Permite buscar por código, nombre o categoría.

```
public static void buscarproducto() {
    System.out.println("Seleccione el tipo de búsqueda:");
    System.out.println("1. Buscar por Código");
    System.out.println("2. Buscar por Nombre");
    System.out.println("3. Buscar por Categoría");
    int opcion = Integer.parseInt(scanner.nextLine());

    switch (opcion) {
        case 1:
            int codUnico = Integer.parseInt(scanner.nextLine());
            mostrarDatosProducto(personajes.get(codUnico - 1));
            break;
        case 2:
            String nombreBuscar =
scanner.nextLine().trim().toLowerCase();
            for (Personaje p : personajes) {
                if (p.nombre.toLowerCase().contains(nombreBuscar)) {
                    mostrarDatosProducto(p);
                }
            }
            break;
        case 3:
            String categoriaBuscar =
scanner.nextLine().trim().toLowerCase();
            for (Personaje p : personajes) {
                if (p.categoria.toLowerCase().contains(categoriaBuscar))
{
                    mostrarDatosProducto(p);
                }
            }
            break;
    }
}
```

Método auxiliar `mostrarDatosProducto`:

```
private static void mostrarDatosProducto(Personaje personaje) {
    System.out.println("Código: " + personaje.codUnico);
    System.out.println("Nombre: " + personaje.nombre);
    System.out.println("Categoría: " + personaje.categoria);
    System.out.println("Precio: " + personaje.precio);
    System.out.println("Stock: " + personaje.stock);
}
```

5.3 Eliminar Producto

Elimina un producto después de solicitar confirmación.

```
private static void eliminarProdcuto() {
    int codUnico = Integer.parseInt(scanner.nextLine());
    System.out.print("¿Está seguro que desea eliminar este Producto?
(S/N): ");
    String confirmacion = scanner.nextLine().trim().toUpperCase();
    if (confirmacion.equals("S")) {
        personajes.remove(codUnico-1);
        System.out.println("Producto eliminado correctamente.");
        registrarBitacora("Eliminar Producto", true);
    }
}
```

5.4 Registrar Venta

Registra una venta, descuenta stock y genera un archivo de ventas:

```
public static void registrarVenta() {
    int codProducto = Integer.parseInt(scanner.nextLine());
    Personaje producto = personajes.get(codProducto - 1);
    int cantidadVendida = Integer.parseInt(scanner.nextLine());

    if (cantidadVendida <= producto.stock) {
        producto.stock -= cantidadVendida;
        int totalVenta = cantidadVendida * producto.precio;
        try (FileWriter writer = new FileWriter("ventas.txt", true)) {
            writer.write("Producto: " + producto.nombre +
                         " | Cantidad: " + cantidadVendida +
                         " | Total: Q" + totalVenta + "\n");
        }
    }
    registrarBitacora("Registrar Venta", true);
}
```

5.5 Bitácora de Acciones

Cada acción se registra en memoria y luego en archivo `bitacora.txt`:

```
public static void registrarBitacora(String accion, boolean correcto) {
    Date fecha = new Date();
    String estado = correcto ? "Correcta" : "Errónea";
    String entrada = fecha + " | Acción: " + accion + " | Estado: " +
estado + " | Usuario: Jhostin Ramírez";
    bitacoraActual.add(entrada);
}
public static void guardarBitacoraEnArchivo() {
    try (PrintWriter pw = new PrintWriter(new FileWriter("bitacora.txt",
true))) {
        for (String entrada : bitacoraActual) {
            pw.println(entrada);
```

```
        }
    }
}
```

5.6 Menú Principal

Controla la navegación del sistema:

```
public static void menuprincipal() {
    salir = false;
    while (!salir) {
        System.out.println("1. Agregar Producto");
        System.out.println("2. Buscar Producto");
        System.out.println("3. Eliminar Producto");
        System.out.println("4. Registrar Venta");
        System.out.println("6. Ver Bitácoras");
        System.out.println("7. Ver Datos del Estudiante");
        System.out.println("8. Salir");
        leermenuprincipal();
    }
}
```