



UNIVERSIDAD  
NACIONAL  
AUTÓNOMA DE  
NICARAGUA,  
MANAGUA  
UNAN-MANAGUA

**CENTRO UNIVERSITARIO REGIONAL DE CHONTALES**  
**“Cornelio Silva Argüello”**  
**UNAN Managua, CUR Chontales**

*2025: “Eficiencia y Calidad para seguir en victoria”*

**DEPARTAMENTO DE CIENCIAS TECNOLÓGICAS Y SALUD**  
**INGENIERÍA EN SISTEMAS DE INFORMACIÓN**  
**I AÑO, II SEMESTRE**

**TEMA**

Desarrollo de un sistema de escritorio para la Farmacia “Vida Sana” en Juigalpa.

**GRUPO 1**

**DOCENTES:**

- José Jonathan Moreno Castro
- Joel Isidro Solano Duartes
- Yesenia Sobeyda Téllez Gómez
- María Jorlenis Ríos Pérez
- Saira María Urbina Cienfuegos

**INTEGRANTES:**

- Jasser Josafaph Bermúdez Suazo
- Ana Gabriela Irías Murillo
- Eduardo Jesús Gudiel González
- Francis Nallely Ayala Calero

**Septiembre, 2025**  
*¡Universidad del Pueblo y para el Pueblo!*

# Índice

## Contenido

<b>1.</b>	<b>Tema .....</b>	4
<b>2.</b>	<b>Objetivos .....</b>	4
<b>3.</b>	<b>Descripción de la problemática .....</b>	4
<b>3.1</b>	<b>Propuesta de Solución .....</b>	4
<b>4.</b>	<b>Funcionalidades del sistema.....</b>	5
<b>5.</b>	<b>Mapeo del software de Vendedor .....</b>	6
<b>6.</b>	<b>Planificación SCRUM .....</b>	7
<b>6.1.</b>	<b>Tabla de roles del equipo de desarrollo.....</b>	7
<b>6.2</b>	<b>Requerimientos funcionales .....</b>	8
<b>6.3</b>	<b>Historias de Usuario .....</b>	8
<b>6.4</b>	<b>Formato Product Backlog Priorizado .....</b>	13
<b>6.5</b>	<b>Sprint Backlog 1.....</b>	13
<b>6.6</b>	<b>Burn down Chart del Sprint 1 .....</b>	16
<b>6.7</b>	<b>Resultados del Sprint 1 .....</b>	16
<b>6.8</b>	<b>Sprint Backlog 2 .....</b>	17
<b>6.6.1</b>	<b>Burn Down Chart del Sprint 2 .....</b>	19
<b>6.6.2</b>	<b>Resultados del Sprint 2 .....</b>	19
<b>6.6</b>	<b>Sprint Backlog 3.....</b>	20
<b>7.</b>	<b>Interfaces de usuario figma y enlace.....</b>	23
<b>7.1</b>	<b>Enlace figma: .....</b>	25
	<a href="https://www.figma.com/design/Y9iRhPMGjEVMgRi2ViadmV/Inventario?node-id=0-1&amp;t=ZrwalQ7V1AvZzLzZ-1">https://www.figma.com/design/Y9iRhPMGjEVMgRi2ViadmV/Inventario?node-id=0-1&amp;t=ZrwalQ7V1AvZzLzZ-1 .....</a>	25
<b>8.</b>	<b>Estructura del proyecto en NetBeans.....</b>	26
<b>9.</b>	<b>Captura de interfaz y programación .....</b>	27
<b>9.1</b>	<b>Paquete Controlador: Formulario MDI (Interfaz y código). .....</b>	27
<b>9.2</b>	<b>Paquete Vista: un formulario para cada entidad para realizar todas las transacciones (Interfaz y código). .....</b>	28
<b>11.</b>	<b>Referencias Bibliográficas.....</b>	40
<b>12.</b>	<b>Anexos.....</b>	41

<b>12.1 Informe de sesión de trabajo Corte I.....</b>	42
<b>12.2 Informe de sesión de trabajo Corte II .....</b>	43
<b>12.2.1 Evidencias de trabajo .....</b>	44
teral.....	44
.....	44
<b>12. 3 Informe de sesión de trabajo Corte III.....</b>	44
<b>12.3.1 Evidencias de trabajo.....</b>	46
<b>13. Cronograma de actividades.....</b>	46

## **1. Tema**

Desarrollo de un sistema de escritorio para la Farmacia “Vida Sana” en Juigalpa

## **2. Objetivos**

Identificar los procesos actuales de gestión en la farmacia “Vida Sana” que nos permita la identificación de requerimientos funcionales para el desarrollo de la aplicación de escritorio.

Diseñar un sistema de escritorio que integre los módulos de ventas y productos donde incluya interfaces de uso práctico, dando cumplimiento a las necesidades del cliente.

Desarrollar una aplicación de escritorio para la farmacia “Vida Sana”, que automatice el registro de ventas, control de medicamentos y generación de reportes, para hacer más eficiente las ventas, controlar mejor los medicamentos y mejorar la atención al cliente.

## **3. Descripción de la problemática**

En la farmacia Vida Sana ubicada en Juigalpa, genera 30.000 C\$ mensuales de venta la cual tiene los siguientes problemas: gestión manual de la empresa dedicada a la comercialización y distribución de medicamentos y productos de cuidado personal. Esta institución requiere una solución tecnológica que le permita mejorar el control de inventario, optimizar el proceso de ventas y facilitar la generación de reportes administrativos garantizando un servicio más eficiente para los clientes. El sistema propuesto consiste en una aplicación de escritorio para farmacias, diseñada para optimizar la gestión de medicamentos, control de inventario y atención al cliente.

### **3.1 Propuesta de Solución**

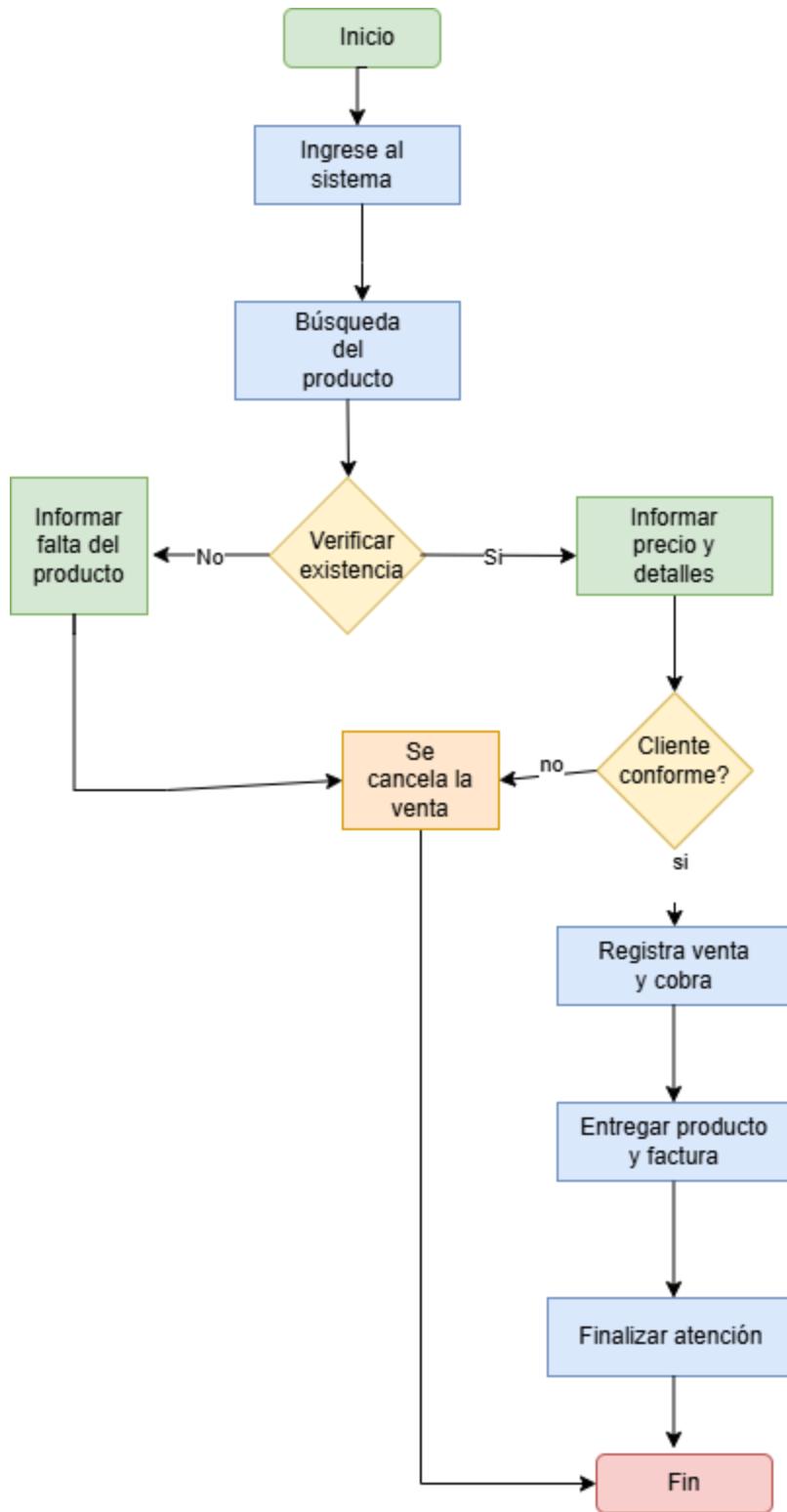
Este sistema permite a los administradores y empleados realizar registros de productos, verificar existencias, gestionar ventas, emitir facturas y recibos, así como llevar el control de fechas de caducidad. El objetivo principal es agilizar los procesos internos de la farmacia, reducir errores humanos y garantizar una mejor atención al usuario final. Además, el sistema contribuye a la toma de decisiones mediante reportes automatizados sobre movimientos de inventario y ventas.

El sistema integra tres áreas fundamentales: venta y atención al cliente, para registrar transacciones, emitir comprobantes y calcular totales; inventario y bodega, para controlar entradas, salidas y alertas sobre desabastecimiento; y administrativa, para generar reportes, gestionar proveedores. En conclusión, automatiza procesos claves que optimizan la gestión de la farmacia.

#### 4. Funcionalidades del sistema

No.	Tipo de usuario	Funcionalidad	Descripción	Encargado
1	Administrador/ Vendedor	Gestión de productos	Registrar,consultar, actualizar y eliminar productos de la ED	Eduardo Guidiel
2	Administrador	Gestion de usuarios	Registrar y eliminar usuarios en el sistema	Ana Gabriela Irias
3	Empleado	Gestión de ventas de productos	Registrar o listar ventas y emitir facturas de ventas realizadas	Jasser Bermudez
4	Adminisreador/ Vendedor	Actualizar Stock	Actualizar nuevas cantidades de productos en el sistema	Jasser Bermudez
5	Empleado	Consultar productos	Consultar disponibilidad de producto en tiempo real	Francis Ayala
6	Administrador	Generar reportes	Los reportes incluyen parámetros como: el rango de fechas y por categoría.	Ana Gabriela Irias
7	Administrador/ Vendedor	Inicio de sesión	Iniciar sesión con un usuario creado dentro del sistema y que pueda acceder a los módulos que le corresponde.	Eduardo Gudiel

## 5. Mapeo del software de Vendedor



## 6. Planificación SCRUM

### 6.1. Tabla de roles del equipo de desarrollo

Rol	Nombre y Apellidos	Funciones
Product owner	Maria Aragón	Listar las necesidades y prioridades del producto y las toma decisiones sobre que se construye primero.
Scrum Master	Francis Ayala	Motivar al equipo de desarrollo para alcanzar los objetivos del proyecto.
DeveloPmet Team	Eduardo Gudiel Ana Irias Jasser Bermúdez	Desarrollar la lógica del sistema. Diseñar la interfaz del usuario, asegurando una experiencia visual y funcional adecuada. Probar las funcionalidades del sistema, identificar errores y verificar que el producto final cumpla con los requerimientos establecidos antes de su entrega.
Stakeholders	José Sequeira	Se encarga en observar los resultados de cada una de las acciones

## 6.2 Requerimientos funcionales

Requerimientos funcionales	
Nº	Descripción
1	Permitir registrar nuevos medicamentos, actualizar precios, eliminar productos obsoletos y consultar inventario.
2	Registrar y eliminar usuarios en el sistema.
3	Registrar ventas, hacer cálculos automáticos de totales, cambios y emitir facturas de ventas realizadas
4	Realizar actualizaciones con la nueva cantidad del producto y especificar qué usuario la realizó.
5	consultar la disponibilidad y ver los detalles del producto.
6	Generar reportes de productos, de ventas y que puedan ser exportados a pdf.
7	Iniciar sesión con el nombre, correo, rol y contraseña para poder acceder a los módulos del sistema.

## 6.3 Historias de Usuario

HU01– Gestión de Productos			
Como	Administrador		
Quiero	Registrar, consultar, actualizar y eliminar medicamentos y productos disponibles		
Para	Mantener un control ordenado del inventario y resolver problemas relacionados con información incorrecta o desactualizada		
Validación		Impacto	32

<b>(Opcional)</b>	<ul style="list-style-type: none"> <li>- Registrar un nuevo medicamento con sus datos principales.</li> <li>- Consultar la lista de productos y detalles.</li> <li>- Actualizar información (precio, vencimiento).</li> <li>- Eliminar productos descontinuados o vencidos.</li> </ul>	<b>Prioridad</b>	5
		<b>Estimación</b>	14 horas

## HU02-Agregar usuarios

<b>Como</b>	Administrador							
<b>Quiero</b>	Gestionar nuevos usuarios en el sistema							
<b>Para</b>	Darles acceso con un rol definido y controlar quién puede operar en el sistema							
<b>Validación (Opcional)</b>	<p>Puede agregar usuarios y visualizar cada uno de los detalles ingresados.</p> <p>El sistema permite ingresar nombre completo, correo y rol.</p> <p>Registra al usuario si los datos son válidos.</p> <p>El sistema evita registrar correos duplicados.</p>	<table border="1" style="width: 100px; margin-left: auto; margin-right: auto;"> <tr> <td>Impacto</td> <td>16</td> </tr> <tr> <td>Prioridad</td> <td>3</td> </tr> <tr> <td>Estimación</td> <td>12 Hrs</td> </tr> </table>	Impacto	16	Prioridad	3	Estimación	12 Hrs
Impacto	16							
Prioridad	3							
Estimación	12 Hrs							

### HU03- Registrar ventas

<b>Como</b>	Empleados		
<b>Quiero</b>	Registrar las ventas realizadas y emitir facturas a los clientes		
<b>Para</b>	Mantener control de las transacciones y entregar comprobantes válidos a los compradores		
<b>Validación (Opcional)</b>	El sistema permite registrar una venta con productos, cantidades y precios	Impacto	32
	Se genera automáticamente la factura (impresa o digital).	Prioridad	5
	Las ventas quedan almacenadas para consultas y reportes.	Estimación	12hrs
	El inventario se actualiza de manera automática después de cada venta.		

### HU04-Actualizar Stock

<b>Como</b>	Administrador, vendedor		
<b>Quiero</b>	Actualizar el stock de los productos en el sistema		
<b>Para</b>	Mantener el inventario al día y evitar faltantes o excesos de productos		
<b>Validación (Opcional)</b>	Puede modificar la cantidad de stock disponible y visualizar el historial de cambios realizados en los productos.	Impacto	20
		Prioridad	3
		Estimación	12 hrs

HU05- Consultar Productos			
<b>Como</b>	Vendedor		
<b>Quiero</b>	Consultar la lista de productos registrados en el sistema de la farmacia.		
<b>Para</b>	Visualizar información importante como nombre, código, cantidad disponible, precio y fecha de vencimiento, con el fin de gestionar correctamente el inventario y realizar ventas precisas.		
<b>Validación (Opcional)</b>	El sistema debe mostrar correctamente todos los productos almacenados en la base de datos y permitir la búsqueda por nombre o código.	<b>Impacto</b>	20
		<b>Prioridad</b>	3
		<b>Estimación</b>	12 hrs

HU06- Reportes			
<b>Como</b>	Administrador		
<b>Quiero</b>	Recibir alertas de los medicamentos que están próximos a vencer		
<b>Para</b>	Retirar productos antes de que caduquen		
Validación (opcional)	El sistema notifica medicamentos con fecha de vencimiento	<b>Impacto</b>	16
	El administrador puede consultar un listado de los próximos a vencer.	<b>Prioridad</b>	3
		<b>Estimación</b>	10 Hrs

	Las alertas se muestran en el sistema o por notificación (según configuración)		
--	--	--	--

HU07- Inicio de Sesión			
<b>Como</b>	Usuario		
<b>Quiero</b>	Iniciar sesión en el sistema de la farmacia mediante mi nombre de usuario y contraseña.		
<b>Para</b>	Acceder a mis funciones de trabajo de forma segura y proteger la información del sistema de usuarios no autorizados.		
<b>Validación (Opcional)</b>	El sistema debe verificar que el usuario y la contraseña coincidan con los datos registrados en la base de datos. En caso contrario, mostrar un mensaje de error.	<b>Impacto</b>	16
	<b>Prioridad</b>	3	
	<b>Estimación</b>	9 Hrs	

#### 6.4 Formato Product Backlog Priorizado

ID	Historia de usuarios	Importancia	Spring programado	Fecha de inicio	Fecha fin
HU01	Gestión de productos	160	1	20/10/2025	23/10/2025
HU03	Registro de ventas	160	1	24/10/2025	27/10/2025
HU04	Actualizar stock	60	2	28/10/2025	31/10/2025
HU02	Agregar Usuarios	60	2	31/10/2025	02/11/2025
HU05	Consultar productos	48	3	02/11/2025	04/11/2025
HU06	Reportes	48	3	05/11/2025	10/10/2025
HU07	Inicio de sesión	48	3	11/11/2025	14/11/2025

#### 6.5 Sprint Backlog 1

ID	Historia de Usuario	Tarea	Tiempo	Responsable
HU01	Agregar productos	Diseño de la interfaz de registro de productos	4hrs	Eduardo Gudiel

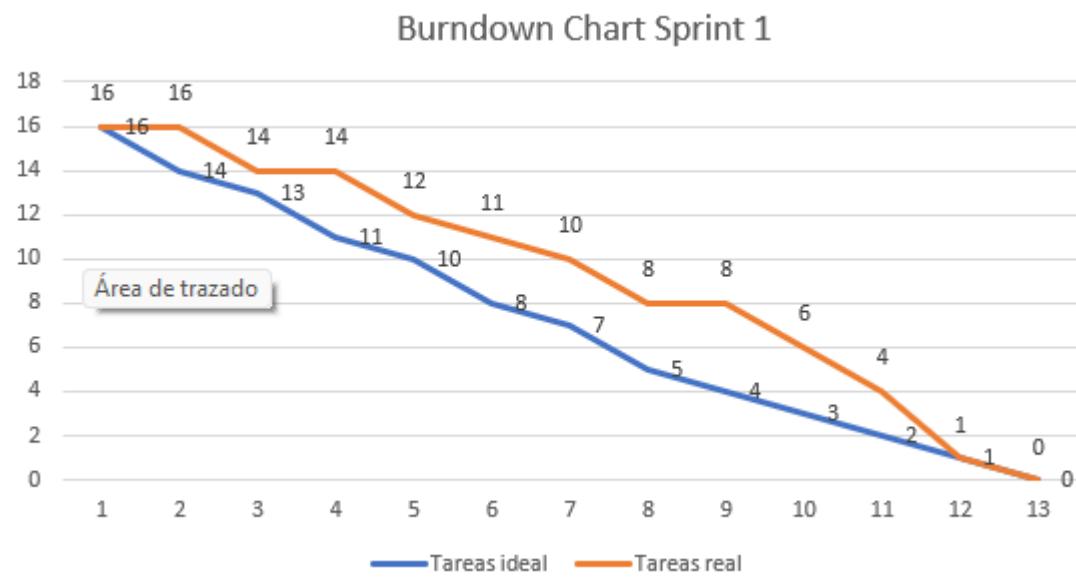
		Programar validación de campos obligatorios y del tipo correcto	2 hrs	Eduardo Gudiel
		Programar la estructura de datos: arreglos y clases	2hrs	Francis Ayala
		Programar la actualización de un producto	1hr	Jasser Bermúdez
		Programar el registro del producto en la estructura de datos.	2hrs	Francis Ayala
		Programar la eliminación del producto.	1hr	Jasser Bermúdez
		programar la visualización de productos.	1hr	Ana Irias
		Pruebas de funcionalidad.	1 hr	Jasser Bermúdez
<b>HU03</b>	Registrar Venta	Diseño de la interfaz de venta (formulario con buscador y detalle).	3 hrs	Ana Irias

	Programar validaciones de datos de entrada.	4 hrs	Ana Irias
	Programar la búsqueda de productos para agregar al detalle de venta.	2 hrs	Eduardo Gudiel
	Programar las validaciones de la venta (que la cantidad no supere al stock).	1hrs	Francis Ayala
	Programar actualización de inventario.	2 hrs	Jasser Bermúdez
	Programar registro de venta.	1hrs	Ana Irias
	Programar generación de factura .	2hr	Eduardo Gudiel
	Realizar pruebas de validación .	2 hrs	Francis Ayala

## 6.6 Burn down Chart del Sprint 1

En este Sprint, al cual en base una perspectiva optimista se le ha asignado una duración de 13 días, se considera un equipo dedicado al desarrollo de 3 miembros con un trabajo ideal de 5 horas al día.

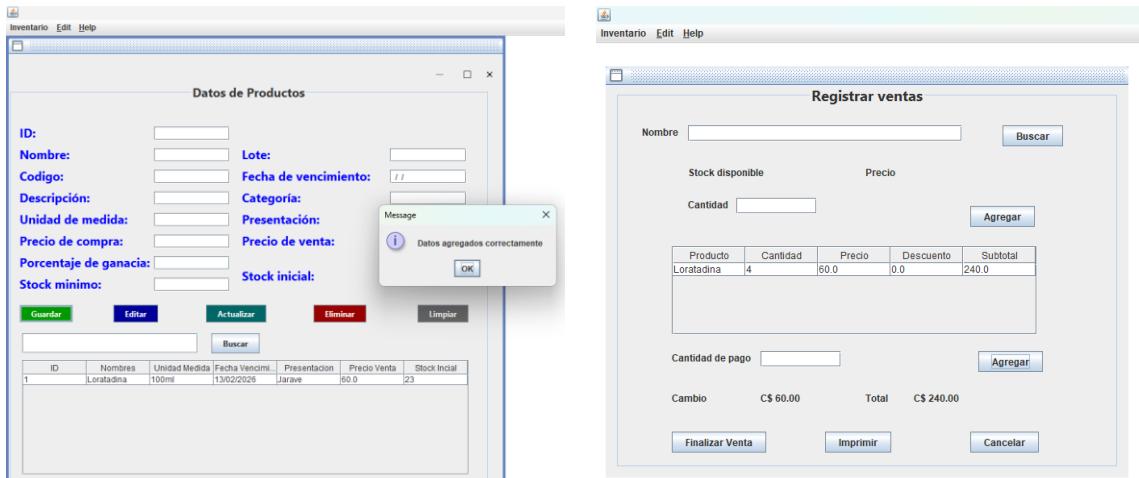
La figura muestra el contraste de la pendiente de trabajo ideal versus la real. Como se observa al inicio del sprint 1 se contabilizan 16 tareas por realizar y se lograron culminar en el tiempo establecido, nos indica que en el siguiente sprint lo podemos calcular de la misma manera asignando la misma cantidad de tareas.



## 6.7 Resultados del Sprint 1

La figura permite observar la creación del formulario de agregar productos que muestra al usuario los productos agregados. La interfaz de productos es un módulo totalmente funcional a como se observa en la figura ya se pueden agregar productos en el sistema también se pueden eliminar, editar y actualizar.

La figura de registrar ventas muestra que se puede realizar una búsqueda de productos, agregarlos a la venta y hacer cálculos automáticos de totales y cambio.



## 6.8 Sprint Backlog 2

ID	Historia de usuario	Tarea	Tiempo	Responsable
HU04	Actualizar stock	Diseño de la interfaz de actualizar stock (campos, botones y navegación lateral).	3 hrs	Francis Ayala
		Programar la validación de los campos obligatorios y del tipo correcto.	2 hrs	Eduardo Gudiel
		Programar el registro del producto en la estructura de datos.	1 hrs	Ana Irias
		Programar la actualización de un producto.	1 hrs	Ana Irias

		Programar los botones de acción (actualizar y cancelar).	2 hrs	Jasser Bermúdez
		Prueba de funcionalidad.	1 hrs	Eduardo Gudiel
<b>HU02</b>	Agregar Usuarios	Diseño de la interfaz de usuarios agregados.	3 hrs	Ana Irias
		Programar la carga y visualización de la lista de usuarios (nombre, correo, rol, estado) en la tabla.	1 hrs	Ana Irias
		Programar la funcionalidad del botón Editar a los usuarios en la tabla.	1 hrs	Eduardo Gudiel
		Implementar la validación de los usuarios que pueden hacer uso del sistema.	1 hrs	Francis Ayala
		Programar la funcionalidad para registrar un nuevo usuario con validación de datos (incluyendo rol).	2 hrs	Jasser Bermúdez
		Programar la funcionalidad del botón “Editar”.	1 hrs	Eduardo Gudiel
		Programar la lógica para actualizar los datos y el “Estado” del usuario en la estructura de datos.	2 hrs	Jasser Bermúdez
		Programar el botón “Eliminar”.	1 hrs	Francis Ayala

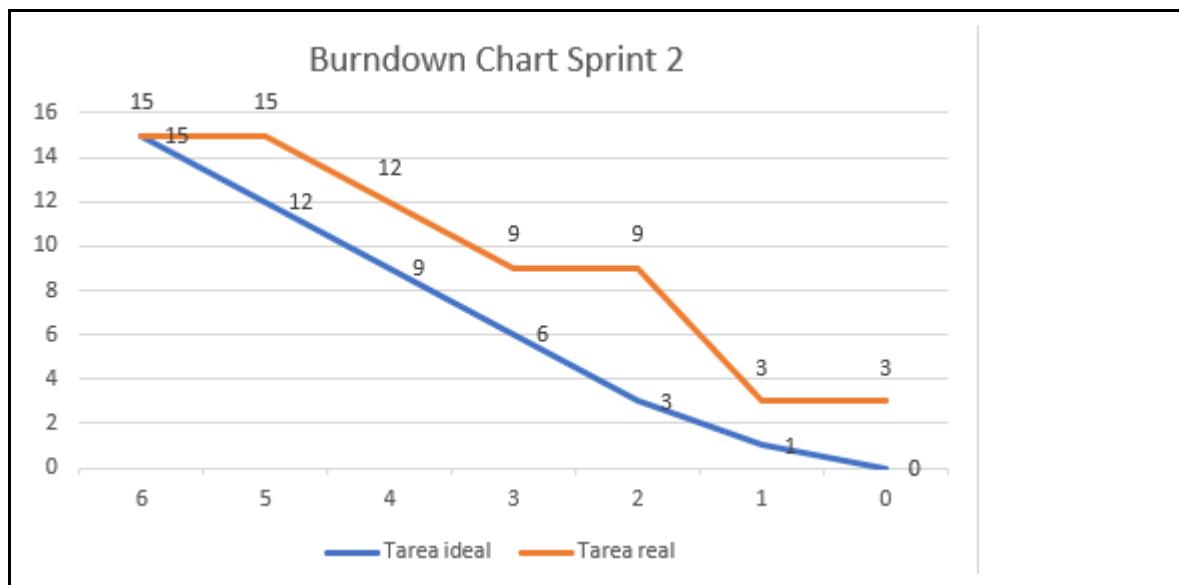
		Prueba de funcionalidad.	1 hrs	Eduardo Gudiel
--	--	--------------------------	-------	----------------

### 6.6.1 Burn Down Chart del Sprint 2

Al Sprint se le ha asignado una duración de 6 días, se considera un equipo dedicado al desarrollo de 3 miembros con un trabajo ideal de 3 horas al día.

La figura muestra el contraste de la pendiente de trabajo ideal versus la real. Como se observa al inicio del sprint 2 se contabilizan 15 tareas por realizar, pero en términos reales 3 no llegan a culminarse en el tiempo establecido.

La funcionalidad que quedó inconclusa en este Sprint deberá desarrollarse fuera del sprint.

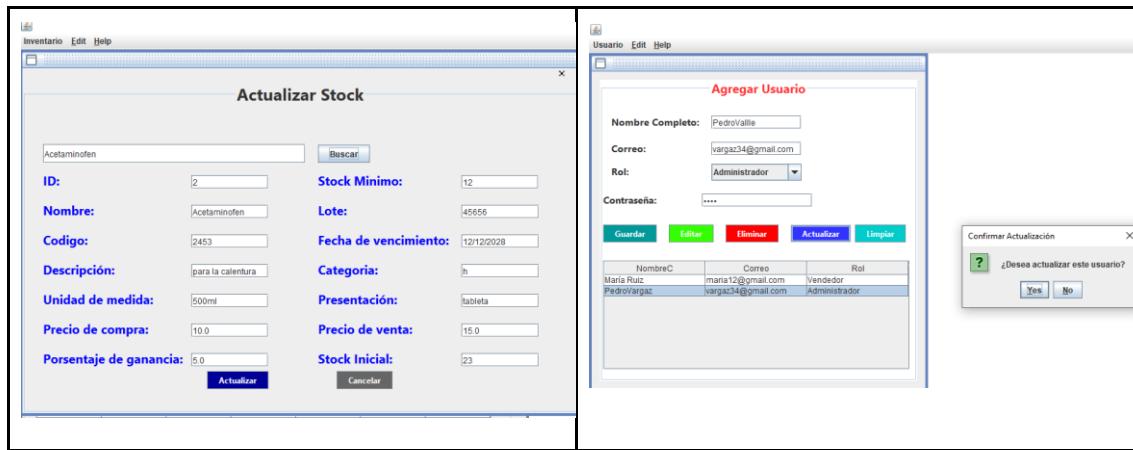


### 6.6.2 Resultados del Sprint 2

La figura permite observar la creación del formulario para actualizar el stock de productos, el cual muestra al usuario la cantidad disponible y permite realizar modificaciones de manera rápida. La interfaz de stock

es un módulo totalmente funcional; como se aprecia en la figura, es posible incrementar o disminuir existencias, así como guardar los cambios realizados en el sistema.

Asimismo, la figura correspondiente al módulo de agregar usuarios muestra un formulario en el que se pueden registrar nuevos usuarios ingresando datos esenciales como nombre, correo y rol dentro del sistema. Este módulo permite guardar, editar y actualizar información de los usuarios registrados, garantizando un control adecuado de quienes tienen acceso al sistema, la acción de los botones eliminar, limpiar y validar la creación de usuarios aun esta inconcluida la funcionalidad.



## 6.6 Sprint Backlog 3

ID	Historia de Usuario	Tarea	Tiempo	Responsable
HU06	Reportes de venta	Diseñar la interfaz del reporte de ventas.	3 hrs	Francis Ayala
		Programar la generación automática del reporte (por fecha o producto).	hrs	Eduardo Gudiel

		Permitir exportar o imprimir el reporte.	2 hrs	Jasser Bermúdez
		Validar que solo usuarios autorizados puedan acceder.	1 hrs	Ana Irias
<b>HU05</b>	Consultar Producto	Diseño de la interfaz de consulta de productos	3 hrs	Eduardo Gudiel
		Programar la visualización de la lista de productos en la estructura de datos.	2 hrs	Eduardo Gudiel
		Programar el funcionamiento del buscador (búsqueda de productos por nombre o código).	1 hrs	Francis Ayala
		Programar el filtro y ordenamiento de productos en la tabla.	2 hrs	Ana Irias

		Programar los botones de acción Editar y Eliminar en las filas de productos.	2 hrs	Jasser Bermúdez
		Programar validación al realizar consultas (evitar campos vacíos).	1 hrs	Jasser Bermúdez
		Realizar pruebas de validación.	1 hrs	Francis Ayala
HU07	Inicio de Sesión	Diseñar la interfaz del formulario (Usuario contraseña y botón de ingreso).	3 hrs	Francis Ayala
		Redirigir al menú principal si el inicio es correcto.	2 hrs	Eduardo Gudiel
		Conectar la estructura de datos para verificar usuario y contraseña.	2 hrs	Jasser Bermúdez

		Implementar opción para cerrar sesión para salir del sistema segura.	1 hrs	Ana Irias
--	--	--	-------	-----------

## 7. Interfaces de usuario figma y enlace.

**Agregar Nuevo productos**

ID	Lote		
Nombre del producto: Acetaminofen	01304		
Código de barra / SKU: 5901234123457	Fecha de vencimiento: 01/09/2028		
+ Agre. productos	Categoría: Analgésico		
Inicio de sesión	Presentación: Tabletas		
Unidad de medida: 600 mg			
Cons. producto	Precio de compra:		
Reg.ventas	Precio de venta:		
Porcentaje de ganancia:	Stock inicial:		
Stock mínimo:			
Reportes			
Guardar	Modificar	Actualizar	Eliminar
Usuarios	Q		
Actualizar Stock			

**Consultar Productos**

Código	Nombre	Categoría	Lote	Fecha vencimiento	Stock venta	Precio compra	Precio venta
SKU001	Ibuprofeno	Analgésico A	L001	30/08/2028	20.00	16.67	24
SKU002	Amoxicilina	Antibiótico B	L00102	20/10/2026	35.00	28.17	42
SKU003	Ácido ascórbico	Vitamina C	V001042	09/07/2028	200.00	166.67	240
SKU004	Dolo-Neurafid	Analgésico D	L0002	25/12/2027	40.00	33.33	48
SKU005	Doxiciclina	Antibiótico E	L0005	15/12/2027	35.00	29.17	42

Editar      Eliminar

**Agregar Usuarios**

Nombre Completo:	<input type="text"/>
Correo:	<input type="text"/>
Rol:	Administrador <input type="button" value="▼"/>
Contraseña:	<input type="text"/>

Nombre Completo	Correo	Rol
Juan Pérez	jperez12@gmail.com	Administrador
Ana Gómez	agomez78@gmail.com	Vendedor
Pedro Martinez	pmartinez9@gmail.com	Vendedor

**Reporte de ventas**

Reportes por mes

Ventas totales C\$ 15,250.00	Número de ventas 235	Productos más vendidos Ibuprofeno, Antimicina
---------------------------------	-------------------------	--

Detalle de transacciones

ID	Fecha	Producto	Código	Lote	Cantidad	Precio	Método pago	Total
01	02/04/25	Tepcin	70021625	250116	5	C\$ 6	Efectivo	C\$ 30
02	02/04/25	Loratadina	89011385	E022502	2	C\$ 4	Efectivo	C\$ 8
03	03/04/25	Amoxicilina	89011382	1124401	10	C\$ 5	Efectivo	C\$ 50

**Registrar ventas**

Código o nombre	<input type="text"/>	<input type="button" value="Buscar"/>	
Stock disponible	50	Precio	C\$ 5
Cantidad	<input type="text" value="10"/>	<input type="button" value="Agregar"/>	

Producto	Cantidad	Precio	Descuento	Subtotal
Amoxicilina	3	C\$ 5	0	C\$ 15

Total  
Tipo de pago  
Cambio

**Inicio de sesión**

**Inicio de sesión**

Correo	<input type="text"/>
Contraseña	<input type="text"/>

Recordarme

¿Olvidó su contraseña? Crear una cuenta

**Inicio**

**Bienvenido**



**FARMACIA  
VIDA SANA**

**Actualizar Stock**



**Actualizar Stock**

Nombre del medicamento:	Paracetamol 500mg
Categoría :	Analgésico
Presentación	Tabletas
Concentración	500mg
Cantidad actual en stock	250
Nueva cantidad agregar o quitar:	
Tipo de operación:	Agregar
Fecha de vencimiento	30/12/2028
Motivo de actualización	
Usuario:	

Fecha y hora: 20/09/2025 10:20

**Botones:** Actualizar, Cancelar

**Reporte vencimiento**

**REPORTE DE VECIMIENTO DE PRODUCTOS**

Filtrar por fecha de vencimiento

Desde: \_\_\_\_\_ Hasta: \_\_\_\_\_ Generar

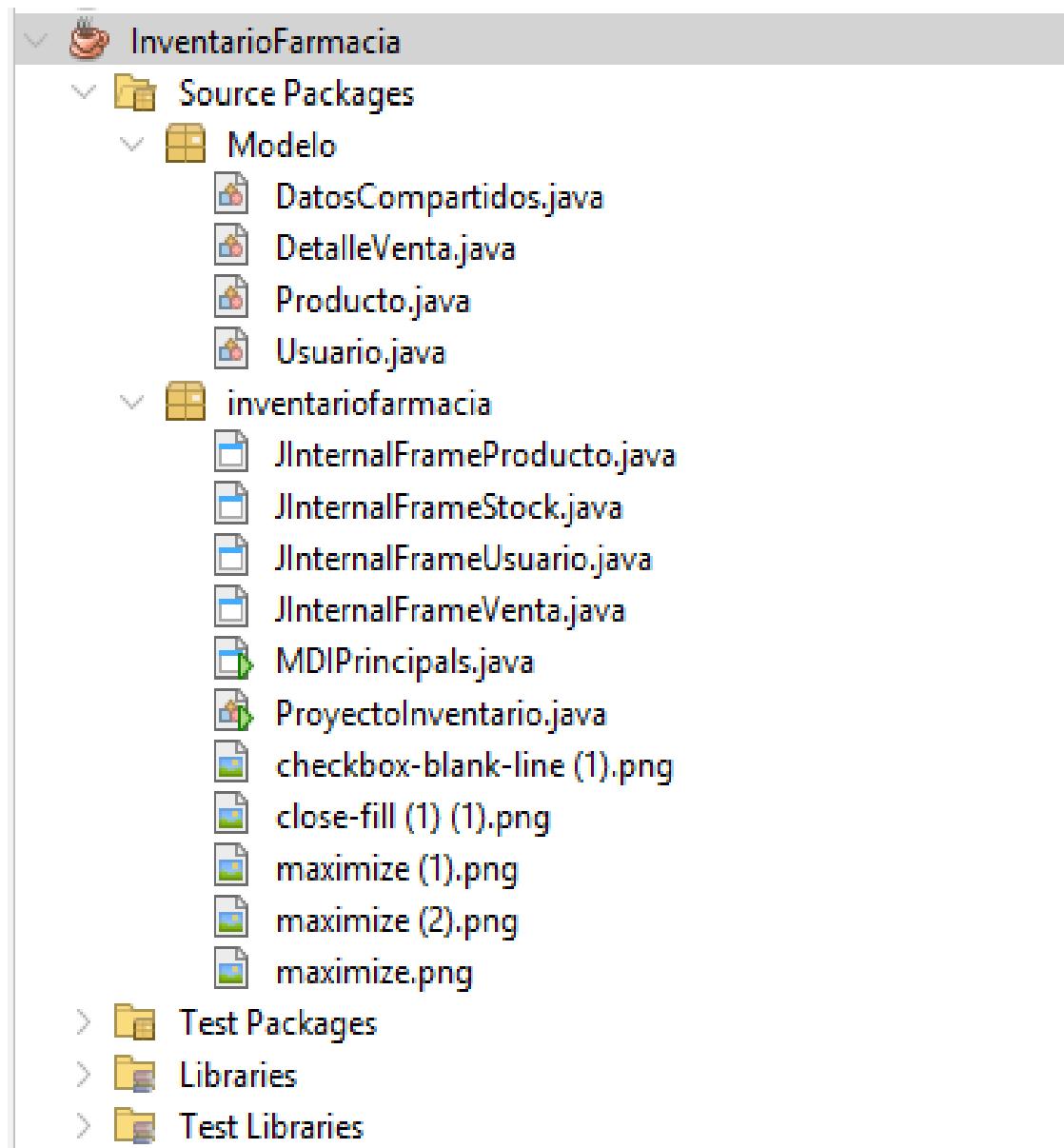
Código	Nombre	Categoría	Fecha de Venc.	Stock
00125	Paracetamol	Analgesico	15/06/2026	50
00103	Omeprazol	Antulcerica	20/07/2026	20
00155	Tetramocin	Antibiotico	08/08/2026	35
00130	Gengenol	Analgesico	09/10/2027	80
00142	Paracetamol	Analgesico	12/11/2026	60

**Botón:** Salir

## 7.1 Enlace figma:

<https://www.figma.com/design/Y9iRhPMGjEVMgRi2ViadmV/Inventario?node-id=0-1&t=ZrwalQ7V1AvZzLzZ-1>

## 8. Estructura del proyecto en NetBeans



## 9. Captura de interfaz y programación

### 9.1 Paquete Controlador: Formulario MDI (Interfaz y código).

```
5 package inventariofarmacia;
6
7 /**
8 * 
9 * $author lapt0
10 */
11 public class MDIPrincipals extends javax.swing.JFrame {
12
13 /**
14 * Creates new form MDIPrincipals
15 */
16 public MDIPrincipals() {
17     initComponents();
18 }
19
20 /**
21 * This method is called from within the constructor to initialize the form.
22 * WARNING: Do NOT modify this code. The content of this method is always
23 * regenerated by the Form Editor.
24 */
25 @SuppressWarnings("unchecked")
26 // Generated Code
27
28 private void GestionUsuariosMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
29     JInternalFrameUsuario jframeUsuario = new JInternalFrameUsuario();
30     desktopPane.add(jframeUsuario);
31     jframeUsuario.show();
32 }
33
34 private void gestionProductoMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
35     JInternalFrameProducto jframeProducto = new JInternalFrameProducto();
36     desktopPane.add(jframeProducto);
37     jframeProducto.show();
38 }
39
40 private void RegistroVentaMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
41     JInternalFrameVenta jframeVenta = new JInternalFrameVenta();
42     desktopPane.add(jframeVenta);
43     jframeVenta.show();
44 }
45
46 private void actualizarStockMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
47     JInternalFrameStock jframeStock = new JInternalFrameStock();
48     desktopPane.add(jframeStock);
49     jframeStock.show();
50 }
51
52 /**
53 * Sparam args the command line arguments
54 */
55 public static void main(String args[]) {
56     /* Set the Nimbus look and feel */
57     /* Look and feel setting code (optional) */
58
59     /* Create and display the form */
60     java.awt.EventQueue.invokeLater(new Runnable() {
61         public void run() {
62             new MDIPrincipals().setVisible(true);
63         }
64     });
65 }
66
67 /**
68 * 
69 * $author lapt0
70 */
71
72 /**
73 * 
74 */
75
76 /**
77 * 
78 */
79
80 /**
81 * 
82 */
83
84 /**
85 * 
86 */
87
88 /**
89 * 
90 */
91
92 /**
93 * 
94 */
95
96 /**
97 * 
98 */
99
100 /**
101 * 
102 */
103
104 /**
105 * 
106 */
107
108 /**
109 * 
110 */
111
112 /**
113 * 
114 */
115
116 /**
117 * 
118 */
119
120 /**
121 * 
122 */
123
124 /**
125 * 
126 */
127
128 /**
129 * 
130 */
131
132 /**
133 * 
134 */
135
136 /**
137 * 
138 */
139
140 /**
141 * 
142 */
143 }
```

```
144 private void gestionProductoMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
145     JInternalFrameProducto jframeProducto = new JInternalFrameProducto();
146     desktopPane.add(jframeProducto);
147     jframeProducto.show();
148 }
149
150 private void RegistroVentaMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
151     JInternalFrameVenta jframeVenta = new JInternalFrameVenta();
152     desktopPane.add(jframeVenta);
153     jframeVenta.show();
154 }
155
156 private void actualizarStockMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
157     JInternalFrameStock jframeStock = new JInternalFrameStock();
158     desktopPane.add(jframeStock);
159     jframeStock.show();
160 }
161
162 /**
163 * Sparam args the command line arguments
164 */
165 public static void main(String args[]) {
166     /* Set the Nimbus look and feel */
167     /* Look and feel setting code (optional) */
168
169     /* Create and display the form */
170     java.awt.EventQueue.invokeLater(new Runnable() {
171         public void run() {
172             new MDIPrincipals().setVisible(true);
173         }
174     });
175 }
176
177 /**
178 * 
179 * $author lapt0
180 */
181
182 /**
183 * 
184 */
185
186 /**
187 * 
188 */
189
190 /**
191 * 
192 */
193
194 /**
195 * 
196 */
197
198 /**
199 * 
200 */
201
202 /**
203 * 
204 */
205
206 /**
207 * 
208 */
209
210 /**
211 * 
212 */
213
214 /**
215 * 
216 */
217
218 /**
219 * 
220 */
221
222 /**
223 * 
224 */
225
226 /**
227 * 
228 */
229
230 /**
231 * 
232 */
233
234 /**
235 * 
236 */
237
238 /**
239 * 
240 */
241
242 /**
243 * 
244 */
245
246 /**
247 * 
248 */
249
250 /**
251 * 
252 */
253
254 /**
255 * 
256 */
257
258 /**
259 * 
260 */
261
262 /**
263 * 
264 */
265
266 /**
267 * 
268 */
269
270 /**
271 * 
272 */
273
274 /**
275 * 
276 */
277
278 /**
279 * 
280 */
281
282 /**
283 * 
284 */
285
286 /**
287 * 
288 */
289
289 /**
290 * 
291 */
292
293 /**
294 * 
295 */
296
297 /**
298 * 
299 */
299
299 /**
300 * 
301 */
302
303 /**
304 * 
305 */
306
307 /**
308 * 
309 */
310
311 /**
312 * 
313 */
314
315 /**
316 * 
317 */
318
319 /**
320 * 
321 */
322
323 /**
324 * 
325 */
326
327 /**
328 * 
329 */
330
331 /**
332 * 
333 */
334
335 /**
336 * 
337 */
338
339 /**
339 * 
340 */
341
342 /**
343 * 
344 */
345
346 /**
347 * 
348 */
349
349 /**
350 * 
351 */
352
353 /**
354 * 
355 */
356
357 /**
358 * 
359 */
360
361 /**
362 * 
363 */
364
365 /**
366 * 
367 */
368
369 /**
369 * 
370 */
371
372 /**
373 * 
374 */
375
376 /**
377 * 
378 */
379
379 /**
380 * 
381 */
382
383 /**
384 * 
385 */
386
387 /**
388 * 
389 */
390
391 /**
392 * 
393 */
394
395 /**
396 * 
397 */
398
399 /**
399 * 
400 */
401
402 /**
403 * 
404 */
405
406 /**
407 * 
408 */
409
409 /**
410 * 
411 */
412
413 /**
414 * 
415 */
416
417 /**
418 * 
419 */
420
421 /**
422 * 
423 */
424
425 /**
426 * 
427 */
428
429 /**
429 * 
430 */
431
432 /**
433 * 
434 */
435
436 /**
437 * 
438 */
439
439 /**
440 * 
441 */
442
443 /**
444 * 
445 */
446
447 /**
448 * 
449 */
450
451 /**
452 * 
453 */
454
455 /**
456 * 
457 */
458
459 /**
459 * 
460 */
461
460 /**
461 * 
462 */
463
464 /**
465 * 
466 */
467
468 /**
469 * 
470 */
471
472 /**
473 * 
474 */
475
476 /**
477 * 
478 */
479
479 /**
480 * 
481 */
482
483 /**
484 * 
485 */
486
487 /**
488 * 
489 */
490
489 /**
490 * 
491 */
492
493 /**
494 * 
495 */
496
497 /**
498 * 
499 */
499
499 /**
500 * 
501 */
502
503 /**
504 * 
505 */
506
507 /**
508 * 
509 */
510
509 /**
510 * 
511 */
512
513 /**
514 * 
515 */
516
517 /**
518 * 
519 */
520
519 /**
520 * 
521 */
522
523 /**
524 * 
525 */
526
527 /**
528 * 
529 */
530
529 /**
530 * 
531 */
532
533 /**
534 * 
535 */
536
535 /**
536 * 
537 */
538
539 /**
540 * 
541 */
542
540 /**
541 * 
542 */
543
544 /**
545 * 
546 */
547
545 /**
546 * 
547 */
548
549 /**
549 * 
550 */
551
550 /**
551 * 
552 */
553
554 /**
555 * 
556 */
557
556 /**
557 * 
558 */
559
559 /**
559 * 
560 */
561
560 /**
561 * 
562 */
563
564 /**
565 * 
566 */
567
566 /**
567 * 
568 */
569
569 /**
569 * 
570 */
571
570 /**
571 * 
572 */
573
574 /**
575 * 
576 */
577
576 /**
577 * 
578 */
579
579 /**
579 * 
580 */
581
580 /**
581 * 
582 */
583
584 /**
585 * 
586 */
587
586 /**
587 * 
588 */
589
589 /**
589 * 
590 */
591
590 /**
591 * 
592 */
593
594 /**
595 * 
596 */
597
596 /**
597 * 
598 */
599
599 /**
599 * 
600 */
601
600 /**
601 * 
602 */
603
604 /**
605 * 
606 */
607
606 /**
607 * 
608 */
609
609 /**
609 * 
610 */
611
610 /**
611 * 
612 */
613
614 /**
615 * 
616 */
617
615 /**
616 * 
617 */
618
618 /**
619 * 
620 */
621
620 /**
621 * 
622 */
623
624 /**
625 * 
626 */
627
626 /**
627 * 
628 */
629
629 /**
629 * 
630 */
631
630 /**
631 * 
632 */
633
634 /**
635 * 
636 */
637
636 /**
637 * 
638 */
639
639 /**
639 * 
640 */
641
640 /**
641 * 
642 */
643
644 /**
645 * 
646 */
647
646 /**
647 * 
648 */
649
649 /**
649 * 
650 */
651
650 /**
651 * 
652 */
653
654 /**
655 * 
656 */
657
656 /**
657 * 
658 */
659
659 /**
659 * 
660 */
661
660 /**
661 * 
662 */
663
664 /**
665 * 
666 */
667
666 /**
667 * 
668 */
669
669 /**
669 * 
670 */
671
670 /**
671 * 
672 */
673
674 /**
675 * 
676 */
677
676 /**
677 * 
678 */
679
679 /**
679 * 
680 */
681
680 /**
681 * 
682 */
683
684 /**
685 * 
686 */
687
686 /**
687 * 
688 */
689
689 /**
689 * 
690 */
691
690 /**
691 * 
692 */
693
694 /**
695 * 
696 */
697
696 /**
697 * 
698 */
699
699 /**
699 * 
700 */
701
700 /**
701 * 
702 */
703
704 /**
705 * 
706 */
707
706 /**
707 * 
708 */
709
709 /**
709 * 
710 */
711
710 /**
711 * 
712 */
713
714 /**
715 * 
716 */
717
715 /**
716 * 
717 */
718
718 /**
719 * 
720 */
721
720 /**
721 * 
722 */
723
724 /**
725 * 
726 */
727
726 /**
727 * 
728 */
729
729 /**
729 * 
730 */
731
730 /**
731 * 
732 */
733
734 /**
735 * 
736 */
737
736 /**
737 * 
738 */
739
739 /**
739 * 
740 */
741
740 /**
741 * 
742 */
743
744 /**
745 * 
746 */
747
746 /**
747 * 
748 */
749
749 /**
749 * 
750 */
751
750 /**
751 * 
752 */
753
754 /**
755 * 
756 */
757
756 /**
757 * 
758 */
759
759 /**
759 * 
760 */
761
760 /**
761 * 
762 */
763
764 /**
765 * 
766 */
767
766 /**
767 * 
768 */
769
769 /**
769 * 
770 */
771
770 /**
771 * 
772 */
773
774 /**
775 * 
776 */
777
776 /**
777 * 
778 */
779
779 /**
779 * 
780 */
781
780 /**
781 * 
782 */
783
784 /**
785 * 
786 */
787
786 /**
787 * 
788 */
789
789 /**
789 * 
790 */
791
790 /**
791 * 
792 */
793
794 /**
795 * 
796 */
797
796 /**
797 * 
798 */
799
799 /**
799 * 
800 */
801
800 /**
801 * 
802 */
803
804 /**
805 * 
806 */
807
806 /**
807 * 
808 */
809
809 /**
809 * 
810 */
811
810 /**
811 * 
812 */
813
814 /**
815 * 
816 */
817
815 /**
816 * 
817 */
818
818 /**
819 * 
820 */
821
820 /**
821 * 
822 */
823
824 /**
825 * 
826 */
827
826 /**
827 * 
828 */
829
829 /**
829 * 
830 */
831
830 /**
831 * 
832 */
833
834 /**
835 * 
836 */
837
836 /**
837 * 
838 */
839
839 /**
839 * 
840 */
841
840 /**
841 * 
842 */
843
844 /**
845 * 
846 */
847
846 /**
847 * 
848 */
849
849 /**
849 * 
850 */
851
850 /**
851 * 
852 */
853
854 /**
855 * 
856 */
857
856 /**
857 * 
858 */
859
859 /**
859 * 
860 */
861
860 /**
861 * 
862 */
863
864 /**
865 * 
866 */
867
866 /**
867 * 
868 */
869
869 /**
869 * 
870 */
871
870 /**
871 * 
872 */
873
874 /**
875 * 
876 */
877
876 /**
877 * 
878 */
879
879 /**
879 * 
880 */
881
880 /**
881 * 
882 */
883
884 /**
885 * 
886 */
887
886 /**
887 * 
888 */
889
889 /**
889 * 
890 */
891
890 /**
891 * 
892 */
893
894 /**
895 * 
896 */
897
896 /**
897 * 
898 */
899
899 /**
899 * 
900 */
901
900 /**
901 * 
902 */
903
904 /**
905 * 
906 */
907
906 /**
907 * 
908 */
909
909 /**
909 * 
910 */
911
910 /**
911 * 
912 */
913
914 /**
915 * 
916 */
917
915 /**
916 * 
917 */
918
918 /**
919 * 
920 */
921
920 /**
921 * 
922 */
923
924 /**
925 * 
926 */
927
926 /**
927 * 
928 */
929
929 /**
929 * 
930 */
931
930 /**
931 * 
932 */
933
934 /**
935 * 
936 */
937
936 /**
937 * 
938 */
939
939 /**
939 * 
940 */
941
940 /**
941 * 
942 */
943
944 /**
945 * 
946 */
947
946 /**
947 * 
948 */
949
949 /**
949 * 
950 */
951
950 /**
951 * 
952 */
953
954 /**
955 * 
956 */
957
956 /**
957 * 
958 */
959
959 /**
959 * 
960 */
961
960 /**
961 * 
962 */
963
964 /**
965 * 
966 */
967
966 /**
967 * 
968 */
969
969 /**
969 * 
970 */
971
970 /**
971 * 
972 */
973
974 /**
975 * 
976 */
977
976 /**
977 * 
978 */
979
979 /**
979 * 
980 */
981
980 /**
981 * 
982 */
983
984 /**
985 * 
986 */
987
986 /**
987 * 
988 */
989
989 /**
989 * 
990 */
991
990 /**
991 * 
992 */
993
994 /**
995 * 
996 */
997
996 /**
997 * 
998 */
999
999 /**
999 * 
1000 */
1001
1000 /**
1001 * 
1002 */
1003
1004 /**
1005 * 
1006 */
1007
1006 /**
1007 * 
1008 */
1009
1009 /**
1009 * 
1010 */
1011
1010 /**
1011 * 
1012 */
1013
1014 /**
1015 * 
1016 */
1017
1015 /**
1016 * 
1017 */
1018
1018 /**
1019 * 
1020 */
1021
1020 /**
1021 * 
1022 */
1023
1024 /**
1025 * 
1026 */
1027
1026 /**
1027 * 
1028 */
1029
1029 /**
1029 * 
1030 */
1031
1030 /**
1031 * 
1032 */
1033
1034 /**
1035 * 
1036 */
1037
1036 /**
1037 * 
1038 */
1039
1039 /**
1039 * 
1040 */
1041
1040 /**
1041 * 
1042 */
1043
1044 /**
1045 * 
1046 */
1047
1046 /**
1047 * 
1048 */
1049
1049 /**
1049 * 
1050 */
1051
1050 /**
1051 * 
1052 */
1053
1054 /**
1055 * 
1056 */
1057
1056 /**
1057 * 
1058 */
1059
1059 /**
1059 * 
1060 */
1061
1060 /**
1061 * 
1062 */
1063
1064 /**
1065 * 
1066 */
1067
1066 /**
1067 * 
1068 */
1069
1069 /**
1069 * 
1070 */
1071
1070 /**
1071 * 
1072 */
1073
1074 /**
1075 * 
1076 */
1077
1076 /**
1077 * 
1078 */
1079
1079 /**
1079 * 
1080 */
1081
1080 /**
1081 * 
1082 */
1083
1084 /**
1085 * 
1086 */
1087
1086 /**
1087 * 
1088 */
1089
1089 /**
1089 * 
1090 */
1091
1090 /**
1091 * 
1092 */
1093
1094 /**
1095 * 
1096 */
1097
1096 /**
1097 * 
1098 */
1099
1099 /**
1099 * 
1100 */
1101
1100 /**
1101 * 
1102 */
1103
1104 /**
1105 * 
1106 */
1107
1106 /**
1107 * 
1108 */
1109
1109 /**
1109 * 
1110 */
1111
1110 /**
1111 * 
1112 */
1113
1114 /**
1115 * 
1116 */
1117
1116 /**
1117 * 
1118 */
1119
1119 /**
1119 * 
1120 */
1121
1120 /**
1121 * 
1122 */
1123
1124 /**
1125 * 
1126 */
1127
1126 /**
1127 * 
1128 */
1129
1129 /**
1129 * 
1130 */
1131
1130 /**
1131 * 
1132 */
1133
1134 /**
1135 * 
1136 */
1137
1136 /**
1137 * 
1138 */
1139
1139 /**
1139 * 
1140 */
1141
1140 /**
1141 * 
1142 */
1143
1144 /**
1145 * 
1146 */
1147
1146 /**
1147 * 
1148 */
1149
1149 /**
1149 * 
1150 */
1151
1150 /**
1151 * 
1152 */
1153
1154 /**
1155 * 
1156 */
1157
1156 /**
1157 * 
1158 */
1159
1159 /**
1159 * 
1160 */
1161
1160 /**
1161 * 
1162 */
1163
1164 /**
1165 * 
1166 */
1167
1166 /**
1167 * 
1168 */
1169
1169 /**
1169 * 
1170 */
1171
1170 /**
1171 * 
1172 */
1173
1174 /**
1175 * 
1176 */
1177
1176 /**
1177 * 
1178 */
1179
1179 /**
1179 * 
1180 */
1181
1180 /**
1181 * 
1182 */
1183
1184 /**
1185 * 
1186 */
1187
1186 /**
1187 * 
1188 */
1189
1189 /**
1189 * 
1190 */
1191
1190 /**
1191 * 
1192 */
1193
1194 /**
1195 * 
1196 */
1197
1196 /**
1197 * 
1198 */
1199
1199 /**
1199 * 
1200 */
1201
1200 /**
1201 * 
1202 */
1203
1204 /**
1205 * 
1206 */
1207
1206 /**
1207 * 
1208 */
1209
1209 /**
1209 * 
1210 */
1211
1210 /**
1211 * 
1212 */
1213
1214 /**
1215 * 
1216 */
1217
1216 /**
1217 * 
1218 */
1219
1219 /**
1219 * 
1220 */
1221
1220 /**
1221 * 
1222 */
1223
1224 /**
1225 * 
1226 */
1227
1226 /**
1227 * 
1228 */
1229
1229 /**
1229 * 
1230 */
1231
1230 /**
1231 * 
1232 */
1233
1234 /**
1235 * 
1236 */
1237
1236 /**
1237 * 
1238 */
1239
1239 /**
1239 * 
1240 */
1241
1240 /**
1241 * 
1242 */
1243
1244 /**
1245 * 
1246 */
1247
1246 /**
1247 * 
1248 */
1249
1249 /**
1249 * 
1250 */
1251
1250 /**
1251 * 
1252 */
1253
1254 /**
1255 * 
1256 */
1257
1256 /**
1257 * 
1258 */
1259
1259 /**
1259 * 
1260 */
1261
1260 /**
1261 * 
1262 */
1263
1264 /**
1265 * 
1266 */
1267
1266 /**
1267 * 
1268 */
1269
1269 /**
1269 * 
1270 */
1271
1270 /**
1271 * 
1272 */
1273
1274 /**
1275 * 
1276 */
1277
1276 /**
1277 * 
1278 */
1279
1279 /**
1279 * 
1280 */
1281
1280 /**
1281 * 
1282 */
1283
1284 /**
1285 * 
1286 */
1287
1286 /**
1287 * 
1288 */
1289
1289 /**
1289 * 
1290 */
1291
1290 /**
1291 * 
1292 */
1293
1294 /**
1295 * 
1296 */
1297
1296 /**
1297 * 
1298 */
1299
1299 /**
1299 * 
1300 */
1301
1300 /**
1301 * 
1302 */
1303
1304 /**
1305 * 
1306 */
1307
1306 /**
1307 * 
1308 */
1309
1309 /**
1309 * 
1310 */
1311
1310 /**
1311 * 
1312 */
1313
1314 /**
1315 * 
1316 */
1317
1316 /**
1317 * 
1318 */
1319
1319 /**
1319 * 
1320 */
1321
1320 /**
1321 * 
1322 */
1323
1324 /**
1325 * 
1326 */
1327
1326 /**
1327 * 
1328 */
1329
1329 /**
1329 * 
1330 */
1331
1330 /**
1331 * 
1332 */
1333
1334 /**
1335 * 
1336 */
1337
1336 /**
1337 * 
1338 */
1339
1339 /**
1339 * 
1340 */
1341
1340 /**
1341 * 
1342 */
1343
1344 /**
1345 * 
1346 */
1347
1346 /**
1347 * 
1348 */
1349
1349 /**
1349 * 
1350 */
1351
1350 /**
1351 * 
1352 */
1353
1354 /**
1355 * 
1356 */
1357
1356 /**
1357 * 
1358 */
1359
1359 /**
1359 * 
1360 */
1361
1360 /**
1361 * 
1362 */
1363
1364 /**
1365 * 
1366 */
1367
1366 /**
1367 * 
1368 */
1369
1369 /**
1369 * 
1370 */
1371
1370 /**
1371 * 
1372 */
1373
1374 /**
1375 * 
1376 */
1377
1376 /**
1377 * 
1378 */
1379
1379 /**
1379 * 
1380 */
1381
1380 /**
1381 * 
1382 */
1383
1384 /**
1385 * 
1386 */
1387
1386 /**
1387 * 
1388 */
1389
1389 /**
1389 * 
1390 */
1391
1390 /**
1391 * 
1392 */
1393
1394 /**
1395 * 
1396 */
1397
1396 /**
1397 * 
1398 */
1399
1399 /**
1399 * 
1400 */
1401
1400 /**
1401 * 
1402 */
1403
1404 /**
1405 * 
1406 */
1407
1406 /**
1407 * 
1408 */
1409
1409 /**
1409 * 
1410 */
1411
1410 /**
1411 * 
1412 */
1413
1414 /**
1415 * 
1416 */
1417
1416 /**
1417 * 
1418 */
1419
1419 /**
1419 * 
1420 */
1421
1420 /**
1421 * 
1422 */
1423
1424 /**
1425 * 
1426 */
1427
1426 /**
1427 * 
1428 */
1429
1429 /**
1429 * 
1430 */
1431
1430 /**
1431 * 
1432 */
1433
1434 /**
1435 * 
1436 */
1437
1436 /**
1437 * 
1438 */
1439
1439 /**
1439 * 
1440 */
1441
1440 /**
1441 * 
1442 */
1443
1444 /**
1445 * 
1446 */
1447
1446 /**
1447 * 
1448 */
1449
1449 /**
1449 * 
1450 */
1451
1450 /**
1451 * 
1452 */
1453
1454 /**
1455 * 
1456 */
1457
1456 /**
1457 * 
1458 */
1459
1459 /**
1459 * 
1460 */
1461
1460 /**
1461 * 
1462 */
1463
1464 /**
1465 * 
1466 */
1467
1466 /**
1467 * 
1468 */
1469
1469 /**
1469 * 
1470 */
1471
1470 /**
1471 * 
1472 */
1473
1474 /**
1475 * 
1476 */
1477
1476 /**
1477 * 
1478 */
1479
1479 /**
1479 * 
1480 */
1481
1480 /**
1481 * 
1482 */
1483
1484 /**
1485 * 
1486 */
1487
1486 /**
1487 * 
1488 */
1489
1489 /**
1489 * 
1490 */
1491
1490 /**
1491 * 
1492 */
1493
1494 /**
1495 * 
1496 */
1497
1496 /**
1497 * 
1498 */
1499
1499 /**
1499 * 
1500 */
1501
1500 /**
1501 * 
1502 */
1503
1504 /**
1505 * 
1506 */
1507
1506 /**
1507 * 
1508 */
1509
1509 /**
1509 * 
1510 */
1511
1510 /**
1511 * 
1512 */
1513
1514 /**
1515 * 
1516 */
1517
1516 /**
1517 * 
1518 */
1519
1519 /**
1519 * 
1520 */
1521
1520 /**
1521 * 
1522 */
1523
1524 /**
1525 * 
1526 */
1527
1526 /**
1527 * 
1528 */
1529
1529 /**
1529 * 
1530 */
1531
1530 /**
1531 * 
1532 */
1533
1534 /**
1535 * 
1536 */
1537
1536 /**
1537 * 
1538 */
1539
1539 /**
1539 * 
1540 */
1541
1540 /**
1541 * 
1542 */
1543
1544 /**
1545 * 
1546 */
1547
1546 /**
1547 * 
1548 */
1549
1549 /**
1549 * 
1550 */
1551
1550 /**
1551 * 
1552 */
1553
1554 /**
1555 * 
1556 */
1557
1556 /**
1557 * 
1558 */
1559
1559 /**
1559 * 
1560 */
1561
1560 /**
1561 * 
1562 */
1563
1564 /**
1565 * 
1566 */
1567
1566 /**
1567 * 
1568 */
1569
1569 /**
1569 * 
1570 */
1571
1570 /**
1571 * 
1572 */
1573
1574 /**
1575 * 
1576 */
1577
1576 /**
1577 * 
1578 */
1579
1579 /**
1579 * 
1580 */
1581
1580 /**
1581 * 
1582 */
1583
1584 /**
1585 * 
1586 */
1587
1586 /**
1587 * 
1588 */
1589
1589 /**
1589 * 
1590 */
1591
1590 /**
1591 * 
1592 */
1593
1594 /**
1595 * 
1596 */
1597
1596 /**
1597 * 
1598 */
1599
1599 /**
1599 * 
1600 */
1601
1600 /**
1601 * 
1602 */
1603
1604 /**
1605 * 
1606 */
1607
1606 /**
1607 * 
1608 */
1609
1609 /**
1609 * 
1610 */
1611
1610 /**
1611
```

**9.2 Paquete Vista:** un formulario para cada entidad para realizar todas las transacciones (Interfaz y código).

**Datos de Productos**

<b>ID:</b>	<input type="text"/>	<b>Lote:</b>	<input type="text"/>
<b>Nombre:</b>	<input type="text"/>	<b>Fecha de vencimiento:</b>	<input type="text"/> / /
<b>Código:</b>	<input type="text"/>	<b>Categoría:</b>	<input type="text"/>
<b>Descripción:</b>	<input type="text"/>	<b>Presentación:</b>	<input type="text"/>
<b>Unidad de medida:</b>	<input type="text"/>	<b>Precio de venta:</b>	<input type="text"/>
<b>Precio de compra:</b>	<input type="text"/>	<b>Stock inicial:</b>	<input type="text"/>
<b>Porcentaje de ganancia:</b>	<input type="text"/>		
<b>Stock mínimo:</b>	<input type="text"/>		

**Guardar**
**Editar**
**Actualizar**
**Eliminar**
**Limpiar**

**Buscar**

ID	Nombres	Unidad Medida	Fecha Vencim.	Presentación	Precio Venta	Stock Inicial

```
4  /*
5  package inventariofarmacia;
6
7  import Modelo.DatosCompartidos;
8  import Modelo.Producto;
9  import java.util.ArrayList; //Libreria a utilizar que se van agregando
10 import javax.swing.JOptionPane;
11 import javax.swing.JDialog; //Libreria para ventana de dialogo
12 import javax.swing.table.DefaultTableModel;
13 import java.awt.event.KeyEvent;
14 import javax.swing.JTextField;
15 import java.beans.PropertyVetoException;
16 import java.util.List;
17
18 /**
19  *
20  * @author lapto
21  */
22 public class JInternalFrameProducto extends javax.swing.JInternalFrame {
23
24 /**
25  * Arreglo de objetos productos
26  */
27 ArrayList<Producto> listaProducto = new ArrayList<>();
28 int indice = -1; // Variable utilizada para para Editar/Actualizar
29
30 DefaultTableModel model;
31 List<Object[]> originalOrder = new ArrayList<>();
32 // Esta variable guardará la instancia de este formulario
33 public static JInternalFrameProducto instanciaAbierta;
```

```
38     public JInternalFrameProducto() { // Constructor del formulario
39         initComponents();
40         model = (DefaultTableModel) jTableProductos.getModel();
41
42         // Le damos a la variable estatica
43         instanciaAbierta = this;
44         // Usamos el metodo para cargar la tabla
45         cargarTabla();
46
47     }
48
49     private void limpiarCampos() { //Metodo para limpiar cajas de texto
50         //se asigna un valor vacio a todas las cajas de texto
51         //Se unifica el nombre de variable asignado
52         this.JTextFieldID.setText("");
53         this.JTextFieldNombreF.setText("");
54         this.JTextFieldCodigo.setLabelText("");
55         this.JTextFieldDescripcion.setText("");
56         this.JTextFieldUnidadM.setText("");
57         this.JTextFieldPrecioI.setText("");
58         this.JTextFieldPorcentajeG.setText("");
59         this.JTextFieldStockM.setText("");
60         this.JTextFieldDote.setText("");
61         this.FechaV.setText("");
62         this.JTextFieldCategoria.setText("");
63         this.JTextFieldPresentacion.setText("");
64         this.JTextFieldPrecioO.setText("");
65         this.JTextFieldStockI.setText("");
66
67     }
68 }
```

```
69     public void cargarProductos() {
70         modelo = new DefaultTableModel();
71         String[] columnas = {"ID", "Nombres", "Unidad Medida", "Fecha Vencimiento",
72                             "Presentación", "Precio Venta", "Stock Inicial"};
73         modelo.setColumnIdentifiers(columnas);
74         for (Producto pro : DatosCompartidos.listadoProductos) {
75             // Agrega cada elemento de la lista a un arreglo
76             // Utiliza un objeto pro del tipo clase producto para obtener
77             // datos de cada atributo, atraves del metodo get
78             String[] renglon = {Integer.toString(pro.getId()), pro.getNombre(),
79                                 pro.getUnidadM(), pro.getFechaV().toString(), pro.getPresentacion(),
80                                 Double.toString(pro.getPrecioV()), Integer.toString(pro.getStockI()));
81             modelo.addRow(renglon);
82         }
83     }
84
85     jTableProductos.setModel(modelo);
86 }
87
88 // Se crea el metodo que guarda el orden original
89 private void guardarOrdenOriginal(DefaultTableModel modelo) {
90     if (modelo == null) {
91         return;
92     }
93     originalOrder.clear(); // Limpia la lista que guarda el orden original
94     for (int i = 0; i < modelo.getRowCount(); i++) {
95         Object[] fila = new Object[modelo.getColumnCount()];
96         for (int j = 0; j < modelo.getColumnCount(); j++) {
97             fila[j] = modelo.getValueAt(i, j);
98         }
99         originalOrder.add(fila);
100 }
```

```

157
158    // Crea el método para restaurar el orden original
159    public void restaurarOrdenOriginal() {
160        if (orden == null || originalOrden.equals(orden)) {
161            JOptionPane.showMessageDialog(null, "No hay un orden guardado para restaurar.", "Aviso", JOptionPane.WARNING_MESSAGE);
162        } else {
163            orden.setEstadoCount(0); // Limpia tabla
164            for (Object[] row : originalOrden) {
165                orden.addRow(row);
166            }
167        }
168    }
169
170    /**
171     * This method is called from within the constructor to initialize the form.
172     * WARNING: Do NOT modify this code. The content of this method is always
173     * regenerated by the Form Editor.
174     */
175
176    // SuggesRusKategori("unchecked")
177    Generated Code
178
179
180    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton1ActionPerformed
181        // Limpiar la tabla
182        limpiarCampos();
183    }
184
185    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton2ActionPerformed
186        // Limpiar la tabla
187        limpiarCampos();
188    }

```

```

531     private void JButtonGuardarActionActionPerformed(java.awt.event.ActionEvent evt) {
532         //Validar que todos los campos de texto no estén vacíos
533         if ((TextFieldID.getText().containsEqual("")))
534             ((TextFieldID.getText().containsEqual("")));
535             ((TextFieldCodigo.getText().containsEqual("")));
536             ((TextFieldDescripcion.getText().containsEqual("")));
537             ((TextFieldUnidadMedida.getText().containsEqual("")));
538             ((TextFieldPrecioUnitario.getText().containsEqual("")));
539             ((TextFieldPorcentajeEfectivo.getText().containsEqual("")));
540             ((TextFieldStockMinimo.getText().containsEqual("")));
541             ((TextFieldStockMaximo.getText().containsEqual("")));
542             ((TextFieldStockActual.getText().containsEqual("")));
543             ((TextFieldCategoria.getText().containsEqual("")));
544             ((TextFieldPresentacion.getText().containsEqual("")));
545             ((TextFieldPrecioVenta.getText().containsEqual("")));
546             ((TextFieldStockOK.getText().containsEqual("")));
547         //Mensaje para informar si algun campo esta vacío // Agrego import
548         JOptionPane.showConfirmDialog(null,
549             "Algunos campos estan vacíos",
550             "Revise todos los campos!",
551             JOptionPane.WARNING_MESSAGE);
552     } else {
553         //Código para obtener un valor
554         //obtiene lo que esta en la caja de texto y lo convierte a tipo de dato
555         int idProducto = Integer.parseInt(this.JTextFieldID.getText());
556         String nombreProducto = this.JTextFieldNombre.getText();
557         String descripcion = this.JTextFieldDescripcion.getText();
558         String unidadMedida = this.JTextFieldUnidadMedida.getText();
559         Double precioCompra = Double.parseDouble(this.JTextFieldPrecioUnitario.getText());

```



```
private void jTextFieldIDKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jTextFieldNombreKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();

    // Permitimos letras (mayúsculas y minúsculas), espacios, Ñ y vocales acentuadas
    if (!Character.isLetter(c)) {
        if (c != ' ') {
            if (c == 'ñ' || c == 'Ñ') {
                evt.consume(); // Ignora cualquier otro carácter
            }
        }
    }
}

private void jTextFieldCodigoKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();
    if (!Character.isLetterOrDigit(c)) {
        evt.consume(); // No permite caracteres que no sean letras o números
    }
}

private void jTextFieldStockKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jTextFieldLoteKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();
    if (!Character.isLetterOrDigit(c)) {
        evt.consume(); // No permite caracteres que no sean letras o números
    }
}

private void jTextFieldStockIKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jLabelMinimizarMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        this.setIcon(true);

        } catch (PropertyVetoException ex) {
            ex.printStackTrace();
        }
}

private void jLabelMaximizarMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        this.setMaximum(true);

        } catch (PropertyVetoException ex) {
            ex.printStackTrace();
        }
}

private void jLabelCerrarMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        this.setClosed(true);
    }
}

private void jTextFieldDescripcionKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();

    // Permite letras y espacios únicamente
    if (!Character.isLetter(car) && car != KeyEvent.VK_SPACE) {
        evt.consume();
    }
}

private void jTextFieldUnidadMKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();
    if (!Character.isLetterOrDigit(c)) {
        evt.consume(); // No permite caracteres que no sean letras o números
    }
}

private void jTextFieldPrecioKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jTextFieldPorcentajeKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();
    JTextField field = (JTextField) evt.getSource();
    String texto = field.getText();
    if (Character.isDigit(c)) {
        if (texto.contains("%")) {
            // No permitir números después del %
            evt.consume();
        } else if (c == '%') {
            if (texto.contains("%") || texto.isEmpty()) {
                // No permitir más de un % o si se escribe como primer carácter
                evt.consume();
            }
        } else {
            // Bloquear cualquier otro carácter
            evt.consume();
        }
    }
}

private void jTextFieldCategoriaKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();

    // Permite letras y espacios únicamente
    if (!Character.isLetter(car) && car != KeyEvent.VK_SPACE) {
        evt.consume();
    }
}

private void jTextFieldPresentacionKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();
    // permite letras (incluye Ñ y acentos), numeros y espacio
    if (!Character.isLetterOrDigit(c)) {
        if (c != ' ') {
            if (c == 'ñ' || c == 'Ñ' || c == 'é' || c == 'é' || c == 'í' || c == 'í' || c == 'ó' || c == 'ó' || c == 'ú' || c == 'ú') {
                evt.consume(); // Ignora cualquier otro carácter
            }
        }
    }
}

private void jTextFieldPrecioKeyTyped(java.awt.event.KeyEvent evt) {
    char car = evt.getKeyChar();
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jLabelCerraMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        this.setClosed(true);

        } catch (PropertyVetoException ex) {
            ex.printStackTrace();
        }
}

private void jLabelUserActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Verifica que la tabla sea inicializada
        if (TableProductos == null) {
            JOptionPane.showMessageDialog(this, "La tabla no está inicializada.", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        // Obtiene el modelo de la tabla
        if (model == null) {
            model = (DefaultTableModel) TableProductos.getModel();
        }

        // Verifica si la tabla tiene filas
        if (model.getRowCount() == 0) {
            JOptionPane.showMessageDialog(this, "No hay productos en la tabla.", "Aviso", JOptionPane.WARNING_MESSAGE);
            return;
        }

        // Guarda el orden original solo la primera vez
        if (originalOrder.isEmpty()) {

```

```

809
810     }
811 
812     // Guarda el nombre producto solo la primera vez
813     if (!comprador.isEmpty()) {
814         guardarNombreOriginal(producto);
815     }
816 
817     // Obtiene el texto ingresado por el usuario
818     String nombreProducto = jTextFieldNombre.getText();
819 
820     if (nombreProducto.isEmpty()) {
821         JOptionPane.showMessageDialog(this, "Ingrese un nombre de producto para buscar.", "Aviso", JOptionPane.WARNING_MESSAGE);
822         return;
823     }
824 
825     // Crea un nuevo que realiza la búsqueda del producto
826     buscadorProductos(nombreProducto);
827 
828     // Limpia la caja de texto después de buscar
829     jTextFieldNombre.setText("");
830 
831 } catch (Exception e) {
832     JOptionPane.showMessageDialog(this, "Error al buscar el producto: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
833     e.printStackTrace();
834 }
```



```

5 package inventariofarmacia;
6
7 import Modelo.DatosCompartidos;
8 import Modelo.DetalleVenta;
9 import Modelo.Producto;
10 import java.beans.PropertyVetoException;
11 import java.util.ArrayList;
12 import javax.swing.JOptionPane;
13 import javax.swing.table.DefaultTableModel;
14
15 /**
16 * @author lapt0
17 */
18 public class JInternalFrameVenta extends javax.swing.JInternalFrame {
19
20     ArrayList<DetalleVenta> listaVenta = new ArrayList<>();
21     DefaultTableModel modeloTabla;
22     double total = 0;
23
24     /**
25      * Creates new form JInternalFrameVenta
26      */
27     public JInternalFrameVenta() {
28         initComponents();
29
30         inicializarTabla();
31     }
32
33 }
```

```

35     // Configurar tabla
36     private void inicializarTabla() {
37         modeloTabla = new DefaultTableModel(
38             new Object[]{"Producto", "Cantidad", "Precio", "Descuento", "Subtotal"}, 0
39         );
40         jTableProductos.setModel(modeloTabla);
41     }
42
43     // CALCULAR CAMBIO AUTOMÁTICAMENTE
44     private void txttipodepagoActionPerformed(java.awt.event.ActionEvent evt) {
45         try {
46             double pago = Double.parseDouble(txttipodepago.getText());
47             double cambio = pago - total;
48             jLabel23.setText("C$ " + String.format("%.2f", cambio));
49         } catch (NumberFormatException e) {
50             JOptionPane.showMessageDialog(this, "Ingrese una cantidad de pago válida.");
51         }
52     }
53
54 }
```

```
316 private void buscarActionPerformed(java.awt.event.ActionEvent evt) {
317     String nombre = nombrebusca.getText().trim();
318
319     Producto encontrado = DatosCompartidos.buscarProductoPorNombre(nombre);
320
321     if (encontrado != null) {
322         cerostock.setText(String.valueOf(encontrado.getStockI()));
323         ceroprecio.setText(String.valueOf(encontrado.getPrecioV()));
324     } else {
325         JOptionPane.showMessageDialog(this, "Producto no encontrado");
326         cerostock.setText("");
327         ceroprecio.setText("");
328     }
329 }
330
331 private void txtcantidadActionPerformed(java.awt.event.ActionEvent evt) {
332 }
333
334
335 private void agregarActionPerformed(java.awt.event.ActionEvent evt) {
336     String nombreBuscado = nombrebusca.getText().trim();
337     if (nombreBuscado.isEmpty()) {
338         JOptionPane.showMessageDialog(this, "Ingrese un nombre de producto.");
339         return;
340     }
341
342     Producto encontrado = null;
343     for (Producto p : DatosCompartidos.listaProductos) {
344         if (p.getNombre().equalsIgnoreCase(nombreBuscado)) {
345
371     if (cantidad > encontrado.getStockI()) {
372         JOptionPane.showMessageDialog(null, "Stock insuficiente. Solo quedan " + encontrado.getStockI() + " unidades.");
373         return;
374     }
375
376     // Se resta la cantidad del stock
377     // Escribir en el formulario que está en "DatosCompartidos.listaProductos"
378     // Esta modifica la cantidad que está en "DatosCompartidos.listaProductos"
379     int nuevoStock = encontrado.getStockI() - cantidad;
380     encontrado.setStock(nuevoStock);
381
382     // Se actualiza la tabla automáticamente
383     // Verifico si el formulario de producto está abierto
384     if (InternalFrameProducto.instanciaAbierta != null) {
385         InternalFrameProducto.instanciaAbierta.cargarTabla();
386     }
387
388     // CÓDIGO DE AVISO DE STOCK BAJO
389     // Comparo el "nuevo stock" que acabé de calcular con el
390     // "stockMinimo" que tiene guardado el producto.
391     // Si NO se tiene la variable se pone a 0
392     if (nuevoStock <= encontrado.getStockMin()) {
393         JOptionPane.showMessageDialog(null, "Aviso de Stock Bajo:\n" +
394             "(Actual Stock: " + encontrado.getNombre() + ")\n" +
395             "Quedan solo " + nuevoStock + " unidades (Mínimo: " + encontrado.getStockMin() + ").\n" +
396             "Aviso de Stock Bajo.");
397         JOptionPane.showMessageDialog(null, "OK");
398     }
399 }
400
401
402 double precio = encontrado.getPrecioV();
403 double descuento = 0;
404 double subtotal = (precio * cantidad) - descuento;
405
406 // Se actualiza el total general
407 total += subtotal;
408 lblTotal.setText("CS " + String.format("%.2f", total));
409
410 // Se crea el objeto DetalleVenta (de momento con pago y cambio = 0)
411 DetalleVenta detalle = new DetalleVenta(
412     encontrado.getNombre(),
413     cantidad,
414     precio,
415     subtotal,
416     0, // pago aún no ingresado
417     0, // cambio aún no calculado
418     total // total acumulado hasta ahora
419 );
420
421 listaVenta.add(detalle);
422
423 // Se muestra en la tabla
424 modeloTabla.addRow(new Object[]{
425     encontrado.getNombre(), cantidad, precio, descuento, subtotal
426 });
427
428 // Limpia campos
429 txtcantidad.setText("");
430 nombrebusca.setText("");
431 cerostock.setText("");
432 ceroprecio.setText("");
433
434
435 private void Agregar2ActionPerformed(java.awt.event.ActionEvent evt) {
436     try {
437         if (listaVenta.isEmpty()) {
438             JOptionPane.showMessageDialog(this, "No hay productos en la venta.");
439             return;
440         }
441
442         double pago = Double.parseDouble(txtpagodepago.getText());
443         double cambio = pago - total;
444
445         lblTotal.setText("CS " + String.format("%.2f", total));
446         lblSubTotal.setText("CS " + String.format("%.2f", cambio));
447
448         if (pago < total) {
449             JOptionPane.showMessageDialog(this,
450                 "El pago es menor al total. Faltan CS " + String.format("%.2f", total - pago));
451         } else {
452             JOptionPane.showMessageDialog(this,
453                 "Pago registrado correctamente.\nCambio: CS " + String.format("%.2f", cambio));
454         }
455
456         // Limpiar los campos de pago/cambio/total
457         for (DetalleVenta detalle : listaVenta) {
458             detalle.setPago(pago);
459             detalle.setCambio(cambio);
460             detalle.setTotal(total);
461         }
462
463         // Limpiar campo de pago
464         txtipodepago.setText("");
465
466     } catch (NumberFormatException e) {
467         JOptionPane.showMessageDialog(this, "Ingrese una cantidad de pago válida.");
468     }
469 }
470
471 private void finalizarIncidenciaActionPerformed(java.awt.event.ActionEvent evt) {
472     if (listaventas.isEmpty()) {
473         JOptionPane.showMessageDialog(this, "No hay productos en la venta.", "Advertencia", JOptionPane.WARNING_MESSAGE);
474         return;
475     }
476
477     int confirmation = JOptionPane.showConfirmDialog(this,
478         "¿Desea finalizar la venta por un total de CS " + String.format("%.2f", total) + "?",
479         "Confirmar venta",
480         JOptionPane.YES_NO_OPTION);
481
482     if (confirmation == JOptionPane.YES_OPTION) {
483         JOptionPane.showMessageDialog(this, "Venta finalizada con éxito. Total: CS " + String.format("%.2f", total));
484
485         // Se guarda la listaVenta
486         DatosCompartidos.listaVentas.add(listaVenta);
487
488         modeloTabla.setRowCount(0);
489         nombrebusca.setText("");
490         txtcantidad.setText("");
491
492         datosVentas.actualizar();
493     }
494 }
```

```

490     txtcantidad.setText("");
491     txtcantidad.setEditable(false);
492     ceroprecio.setText("CG 0");
493     cerosuma.setText("CG 0");
494     initTotal.setText("CG 0");
495     plazos.setText("CG 0");
496     total = 0;
497     listaVenta.clear();
498 }
499 else {
500     JOptionPane.showMessageDialog(this, "Venta cancelada.");
501 }
502 }
503
504 private void btnImprimirActionPerformed(java.awt.event.ActionEvent evt) {
505
506     try {
507         // Obtener el pago ingresado
508         String textoPago = txttipodePago.getText().trim();
509         if (textoPago.isEmpty()) {
510             JOptionPane.showMessageDialog(this, "Ingrese la cantidad de pago antes de imprimir la factura.");
511             return;
512         }
513
514         double pago = Double.parseDouble(textoPago);
515
516         // Obtener el total de la etiqueta lblTotal
517         String textoTotal = lblTotal.getText().replace("CG", "").trim();
518         double total = Double.parseDouble(textoTotal);
519
520         // Calcular el cambio
521
522     } catch (Exception ex) {
523         JOptionPane.showMessageDialog(this, "Error al calcular el cambio.");
524     }
525 }

```

```

526
527     // Calcula el cambio
528     double cambio = pago - total;
529
530     Factura factura;
531
532     String pagoFormatado = String.format("%,.2f", pago);
533     String totalFormatado = String.format("%,.2f", total);
534     String cambioFormatado = String.format("%,.2f", cambio);
535
536     // Crea el rute de la factura
537     String rutaFactura = String.format("FACTURA %s\\%s");
538
539     // Agrega los datos de la venta
540     for (int i = 0; i < modeloItens.getitensCount(); i++) {
541         factura.append("itens[" + i + "].getDescricao() - ");
542         factura.append("itens[" + i + "].getUnidadeM() - ");
543         factura.append("itens[" + i + "].getPrecoC() - ");
544         factura.append("itens[" + i + "].getPrecoV() - ");
545         factura.append("itens[" + i + "].getValorUnit(i, 0).append(',')");
546     }
547
548     // Agrega total, pago y cambio
549     factura.append("itens[0].getTotalFormatado() - ");
550     factura.append("itens[0].getPrecoV() - ");
551     factura.append("itens[0].getPrecoC() - ");
552
553     // Muestra la factura
554     JOptionPane.showMessageDialog(this, factura.toString(), "Factura", JOptionPane.INFORMATION_MESSAGE);
555
556 } catch (CharacterFormatException ex) {
557     JOptionPane.showMessageDialog(this, "El valor de pago no es valido. Ingrese un numero.", "Error", JOptionPane.ERROR_MESSAGE);
558 }

```



```

552 private void cancelarActionPerfomed(java.awt.event.ActionEvent evt) {
553
554     // Limpiar el campo de texto
555     txttipodePago.setText("");
556
557     // Reiniciar etiquetas
558     jLabel23.setText("CG 0");
559     lblTotal.setText("CG 0");
560
561     // resetear el total
562     total = 0;
563
564     // Mostrar mensaje
565     JOptionPane.showMessageDialog(this, "Operación cancelada.");
566 }
567
568 private void jLabelCerrarMouseClicked(java.awt.event.MouseEvent evt) {
569     try {
570         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
571     } catch (PropertyVetoException ex) {
572         ex.printStackTrace();
573     }
574 }
575
576
577
578 // Variables declaration - do not modify
579 private javax.swing.JButton Agregar;
580 private javax.swing.JButton agregar;
581 private javax.swing.JButton btnImprimir;

```

```

5 package inventariofarmacia;
6
7 import Modelo.DatosCompartidos;
8 import Modelo.Producto;
9 import java.awt.event.KeyEvent;
10 import java.beans.PropertyVetoException;
11 import javax.swing.JOptionPane;
12 import javax.swing.JTextField;
13
14 /**
15 * @author lapto
16 */
17
18 public class JInternalFrameStock extends javax.swing.JInternalFrame {
19
20     // Guardará el producto que encontramos para poder usarlo
21     // tanto en el botón "Buscar" como en el botón "Guardar Cambios".
22     private Producto productoEncontrado;
23
24     /**
25      * Creates new form JInternalFrameStock
26     */
27     public JInternalFrameStock() {
28         initComponents();
29     }
30
31
32     private void limpiarCampos() {
33         jTextFieldID.setText("");
34
35
36
37
38
39
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
567
568
569
569
570
571
572
573
574
575
576
577
577
578
579
579
580
581
582
583
584
585
586
587
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
767
768
769
769
770
771
772
773
774
775
776
777
777
778
779
779
780
781
782
783
784
785
786
787
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
817
818
819
819
820
821
822
823
824
825
826
827
827
828
829
829
830
831
832
833
834
835
836
837
838
838
839
839
840
841
842
843
844
845
846
847
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
867
868
869
869
870
871
872
873
874
875
876
877
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
895
896
897
897
898
899
899
900
901
902
903
904
905
905
906
907
907
908
909
909
910
911
912
913
913
914
915
915
916
917
917
918
919
919
920
921
922
923
923
924
925
925
926
927
927
928
929
929
930
931
932
933
933
934
935
935
936
937
937
938
939
939
940
941
942
942
943
944
944
945
946
946
947
947
948
948
949
949
950
951
951
952
953
953
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1
```

```

414     private void jTextFieldPorcentajeGKeyTyped(java.awt.event.KeyEvent evt) {
415         char c = evt.getKeyChar();
416         JTextField field = (JTextField) evt.getSource();
417         String texto = field.getText();
418
419         if (Character.isDigit(c)) {
420             if (texto.contains("%")) {
421                 // No permitir números después del %
422                 evt.consume();
423             }
424         } else if (c == '%') {
425             if (texto.contains("%") || texto.isEmpty()) {
426                 // No permitir más de un % o si se escribe como primer carácter
427                 evt.consume();
428             }
429         } else {
430             // Bloquear cualquier otro carácter
431             evt.consume();
432         }
433     }
434
435     private void jTextFieldPorcentajeG1KeyTyped(java.awt.event.KeyEvent evt) {
436         char c = evt.getKeyChar();
437         JTextField field = (JTextField) evt.getSource();
438         String texto = field.getText();
439
440         if (Character.isDigit(c)) {
441             if (texto.contains("%")) {
442                 // No permitir números después del %
443                 evt.consume();
444             }
445         }
446
447         if (c != 'ñ') {
448             if (c != 'ñ' && c != 'Ñ' && c != 'é' && c != 'É' && c != 'í' && c != 'Í' && c != 'ó' && c != 'Ó') {
449                 evt.consume(); // Ignora cualquier otro carácter
450             }
451         }
452     }
453
454     private void jTextFieldIDKeyTyped(java.awt.event.KeyEvent evt) {
455         char car = evt.getKeyChar();
456         JTextField field = (JTextField) evt.getSource();
457         String texto = field.getText();
458
459         if (Character.isDigit(car)) {
460             if (car < '0' || car > '9') {
461                 evt.consume(); // Ignora el carácter si no es un número
462             }
463         }
464
465         private void jTextFieldNombrePKKeyTyped(java.awt.event.KeyEvent evt) {
466             char c = evt.getKeyChar();
467
468             // Permitimos letras (mayúsculas y minúsculas), espacios, ñ y vocales acentuadas
469             if (!Character.isLetter(c))
470                 if (c != ' ' && c != 'ñ' && c != 'Ñ' && c != 'é' && c != 'É' && c != 'í' && c != 'Í' && c != 'ó' && c != 'Ó')
471                     evt.consume();
472     }
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530

```

```
529 // Permite letras y espacios únicamente
530 if (!Character.isLetter(car) && car != KeyEvent.VK_SPACE) {
531     evt.consume();
532 }
533 }
534
535 private void jTextFieldPresentacionKeyTyped(java.awt.event.KeyEvent evt) {
536     char car = evt.getKeyChar();
537     // permite introducir sólo ñ y acentos), numeros y espacio
538     if (!Character.isLetterOrDigit(car))
539         car = ' ';
540     if ('ñ' <= car && car <= 'Ñ') {
541         evt.consume();
542     }
543     else if ('ñ' <= car && car <= 'Ñ') {
544         evt.consume(); // Ignora cualquier otro carácter
545     }
546 }
547
548 private void jTextFieldPrecioKeyTyped(java.awt.event.KeyEvent evt) {
549     char car = evt.getKeyChar();
550     // Solo permite números del 0 al 9
551     if (car < '0' || car > '9') {
552         evt.consume(); // Ignora el carácter si no es un número
553     }
554 }
555
556 private void jTextFieldStockIKeyTyped(java.awt.event.KeyEvent evt) {
557     char car = evt.getKeyChar();
558     // Solo permite números del 0 al 9
```

```
    // Solo permite números del 0 al 9
    if (car < '0' || car > '9') {
        evt.consume(); // Ignora el carácter si no es un número
    }
}

private void jButtonBuscarActionPerformed(java.awt.event.ActionEvent evt) {
    String nombreBuscado = jTextFieldBuscar.getText().trim();

    if (nombreBuscado.isEmpty()) {
        JOptionPane.showInputDialog(this, "Por favor, ingrese un nombre para buscar.");
        return;
    }

    // Reiniciamos la variable clave
    this.productoEncontrado = null;

    // Buscamos en la lista GLOBAL
    for (Producto p : DatosCompartidos.listaProducto) {
        if (p.getNombre().equalsIgnoreCase(nombreBuscado)) {
            this.productoEncontrado = p; //Lo encontramos! Lo guardamos en la variable clave
            break; // Salimos del bucle
        }
    }

    // llenamos las 14 cajas de texto
    if (this.productoEncontrado != null) {
        // Si lo encontramos, llenaremos todos los campos
        // Usamos los "getters" del objeto "productoEncontrado"
    }
}
```

```

877 // Usamos el "getters" del objeto "productoEncontrado"
878 // Los numeros (int, double) deben convertirse a String
879 jTextFieldID.setText(String.valueOf(productoEncontrado.getId()));
880 jTextFieldNombreP.setLabelText(productoEncontrado.getNombre());
881 jTextFieldCodigoP.setLabelText(productoEncontrado.getCodigo());
882 jTextFieldDescripcionP.setLabelText(productoEncontrado.getDescripcion());
883 jTextFieldStockP.setLabelText(String.valueOf(productoEncontrado.getStockP()));
884 jTextFieldPrecioC.setLabelText(String.valueOf(productoEncontrado.getPrecioC()));
885 jTextFieldPorcentajeI.setLabelText(String.valueOf(productoEncontrado.getPorcentajeI()));
886 jTextFieldStockM.setLabelText(String.valueOf(productoEncontrado.getStockM()));
887 jTextFieldStockD.setLabelText(String.valueOf(productoEncontrado.getStockD()));
888 jTextFieldStockA.setLabelText(String.valueOf(productoEncontrado.getStockA()));
889 jTextFieldFechV.setLabelText(productoEncontrado.getFechaV());
890 jTextFieldCategoria.setLabelText(productoEncontrado.getCategoria());
891 jTextFieldPresentacion.setLabelText(productoEncontrado.getPresentacion());
892 jTextFieldPrecioV.setLabelText(String.valueOf(productoEncontrado.getPrecioV()));

893 // El campo de stockExistencia
894 jTextFieldStockE.setLabelText(String.valueOf(productoEncontrado.getStockE()));

895 } else {
896     // Si no lo encontramos, mostramos un mensaje y limpiamos las cajas
897     JOptionPane.showMessageDialog(this, "Producto no encontrado.");
898     limpiarCampos(); // llama al metodo limpiarCampos
899 }
900 }

911 private void jButtonActualizarActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButtonActualizarActionPerformed
912
913     // 1. Verificamos que SI tengamos un producto cargado
914     if (this.productoEncontrado == null) {
915
916

```

```

    // L. Verificamos que SI tengamos un producto cargado
    if (this.productoEsCargado == null) {
        JOptionPanePane.showMessgeDialog(this, "#Primero debe buscar un producto antes de guardar.");
        return;
    }

    //Validar que todos los campos de texto no estén vacíos
    if ((TextFieldID.getText().contentEquals("")) ||
        ((TextFieldNombre.getText().contentEquals("")) ||
        ((TextFieldDescripcion.getText().contentEquals("")) ||
        ((TextFieldFamilia.getText().contentEquals("")) ||
        ((TextFieldFrecuencias.getText().contentEquals("")) ||
        ((TextFieldFrecuStock.getText().contentEquals("")) ||
        ((TextFieldStock.getText().contentEquals("")) ||
        ((TextFieldDias.getText().contentEquals("")) ||
        ((TextFieldAlmacen.getText().contentEquals("")) ||
        ((TextFieldPresentacion.getText().contentEquals("")) ||
        ((TextFieldFrecuV.getText().contentEquals("")) ||
        ((TextFieldFrecuStock.getText().contentEquals("")))) {
            //Mensaje para informar si algún campo está vacío // Agregó import
            JOptionPanePane.showMessgeDialog(this,
                "Algunos campos están vacíos",
                "Revise todos los campos",
                JOptionPane.WARNINg_MESSAGE);
        }
    }

    return;
}

// Recuperamos y convertimos toutes los nuevos valores

```

```
644    try {
645        // Recuperamos y convertimos todos los nuevos valores
646        String nombreProducto = jTextFieldNombre.getText().trim();
647        String codigo = jTextFieldCodigo.getText().trim();
648        String descripcion = jTextFieldDescripcion.getText().trim();
649        String unidaMedida = jTextFieldUnidad.getText().trim();
650        String lote = jTextFieldLote.getText().trim();
651        String fechaVencimiento = Fecha.getFecha().trim();
652        String categoria = jTextFieldCategoria.getText().trim();
653        String presentacion = jTextFieldPresentacion.getText().trim();
654        // creamos los datos numericos
655        double precioCompra = Double.parseDouble(jTextFieldPrecio.getText().trim());
656        double porcentaje = Double.parseDouble(jTextFieldPorcentaje.getText().trim());
657        int stockM = Integer.parseInt(jTextFieldStockM.getText().trim());
658        double precioV = Double.parseDouble(jTextFieldPrecioV.getText().trim());
659        int stock = Integer.parseInt(jTextFieldStock.getText().trim());
660        id = Integer.parseInt(jTextFieldID.getText().trim());
661
662        this.productoEncontrado.setNombre(nombreProducto);
663        this.productoEncontrado.setCodigo(codigo);
664        this.productoEncontrado.setDescripcion(descripcion);
665        this.productoEncontrado.setUnidad(unidaMedida);
666        this.productoEncontrado.setLote(lote);
667        this.productoEncontrado.setFechaV(fechaVencimiento);
668        this.productoEncontrado.setCategoria(categoría);
669        this.productoEncontrado.setPresentacion(presentacion);
670        this.productoEncontrado.setPrecioC(precioCompra);
671        this.productoEncontrado.setPorcentaje(porcentaje);
672        this.productoEncontrado.setStock(stockM);
673        this.productoEncontrado.setPrecioV(precioV);
```

```

    this.producto.setNombre(producto.getNombre());
    this.producto.setPrecio(producto.getPrecio());
    this.producto.setStock(producto.getStock());
    this.producto.setVendido(producto.getVendido());
}

// Añade el producto al sistema de productos para actualizarse la tabla
if (!this.producto.getNombre().equals("") & !this.producto.getNombre() == null) {
    JInternalFrameProductos.instanciaActualizaCarpetilla();
}

// Notificación en mensaje confirmando la actualización
JOptionPane.showMessageDialog(this, "Producto " + nombreProducto + " actualizado correctamente.", "Básico", JOptionPane.INFORMATION_MESSAGE);

}

class ClienteActualizacionException extends Exception {
    // Constructor por defecto
    public ClienteActualizacionException() {
        super("Error de formato en campo número, verifica que los campos Número (precio, stock) contenga solo números válidos.");
    }
    // Constructor personalizado
    public ClienteActualizacionException(String message) {
        super(message);
    }
}

// Sección (Exception e) {
    // Mensaje de error
    JOptionPane.showMessageDialog(this, "Ocurrió un error interpretando el actualizar: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
}

}

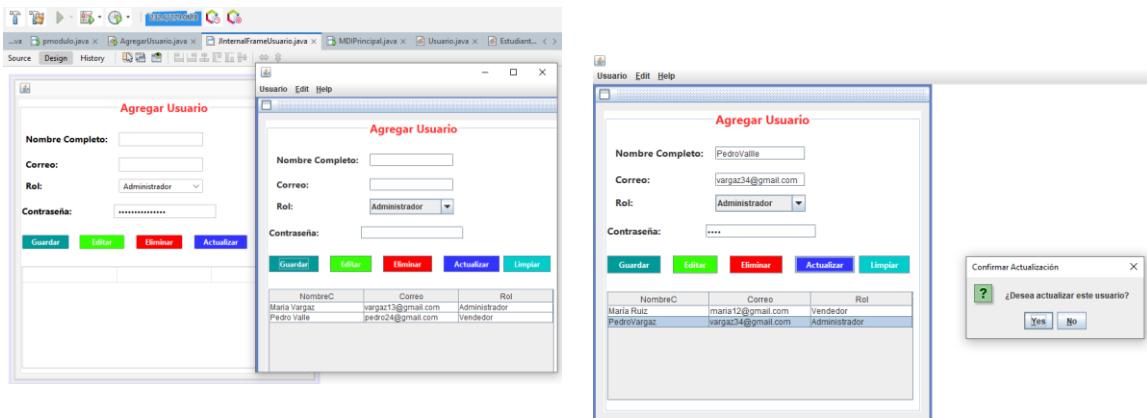
private void buttonActualizarActionPerformed(java.awt.event.ActionEvent evt) {
    String nombreCompleto = nombreCompletoField.getText();
    String precioString = precioField.getText();
    String stockString = stockField.getText();
    String vendidoString = vendidoField.getText();
}

```

```

701     private void jLabelCerrarMouseClicked(java.awt.event.MouseEvent evt) {
702         try {
703             this.setClosed(true);
704
705         } catch (PropertyVetoException ex) {
706             ex.printStackTrace();
707         }
708     }
709
710
711     // Variables declaration - do not modify
712     private javax.swing.JFormattedTextField FechaV;
713     private javax.swing.JButton jButtonActualizar;
714     private javax.swing.JButton jButtonBuscar;
715     private javax.swing.JButton jButtonCancelar;
716     private javax.swing.JLabel jLabel1;
717     private javax.swing.JLabel jLabel10;
718     private javax.swing.JLabel jLabel11;
719     private javax.swing.JLabel jLabel12;
720     private javax.swing.JLabel jLabel13;
721     private javax.swing.JLabel jLabel14;
722     private javax.swing.JLabel jLabel15;
723     private javax.swing.JLabel jLabel16;
724     private javax.swing.JLabel jLabel17;
725     private javax.swing.JLabel jLabel18;
726     private javax.swing.JLabel jLabel19;
727     private javax.swing.JLabel jLabel2;
728     private javax.swing.JLabel jLabel3;
729     private javax.swing.JLabel jLabel4;
730     private javax.swing.JLabel jLabel5;
731     private javax.swing.JLabel jLabel6;
732     private javax.swing.JLabel jLabel7;
733     private javax.swing.JLabel jLabel8;
734     private javax.swing.JLabel jLabel9;
735     private javax.swing.JLabel jLabelCerrar;

```



**Agregar Usuario**

NombreCompleto	Correo	Rol
Pedro Martinez	pedro24@gmail.com	Vendedor
Maria Vargaz	vargaz13@gmail.com	Administrador

**Eliminar**

¿Está seguro de eliminar la fila número 0?

Yes No

**Advertencia**

Seleccione una fila de la tabla para editar

**Source**

```

private void llenarTabla() {
    DefaultTableModel modelo = new DefaultTableModel();
    //Arreglo con nombre de las columnas de la tabla
    String [] columnas = {"NombreC","Correo","Rol"};
    //Establece los nombre definidos de las columnas
    modelo.setColumnIdentifiers(columnas);
    //Ciclo para recorrer la lista a un arreglo
    for(Usuario u: listaUsuario){
        //Agrega cada elemento de la lista a un arreglo
        //Utiliza un objeto pro del tipo clase usuario para obtener
        // dato del cada atributo, a traves de metodo get
        String [] renglon = {u.getNombreCompleto(), u.getCorreo(),
            u.getRol(),u.getPassword()};
        modelo.addRow(renglon);
    }
    //Agrega todos los elementos de arreglo renglon a la tabla usuario
    jTableUsuario.setModel(modelo);
}

```

**Design**

**Source**

```

private void jButtonGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    //Obtener los valores de los campos
    String nombre = jTextFieldNombreC.getText().trim();
    String correo = jTextFieldCorreo.getText().trim();
    String rol = jBoxRol.getSelectedItem().toString();
    String contraseña = new String(jFieldContraseña.getPassword());

    //Validar que no estén vacios
    if (nombre.isEmpty() || correo.isEmpty() || contraseña.isEmpty()) {
        JOptionPane.showMessageDialog(null,
            "Por favor, complete todos los campos",
            "Campos Vacíos",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    // Agrega el nuevo Usuario en la lista
    Usuario nuevoUsuario = new Usuario (nombre, correo, rol, contraseña);
    listaUsuario.add(nuevoUsuario);

    // Actualizar la tabla
    llenarTabla();

    // Mostrar mensaje de éxito
    JOptionPane.showMessageDialog(null, "Usuario guardado correctamente");
    limpiarCampos();
}

```

**Source**

```

private void jButtonEditarActionPerformed(java.awt.event.ActionEvent evt) {
    // Obtener la fila seleccionada en la tabla
    int filaSeleccionada = jTableUsuario.getSelectedRow();

    // Validar si se seleccionó una fila
    if (filaSeleccionada == -1) {
        JOptionPane.showMessageDialog(null,
            "Seleccione una fila de la tabla para editar",
            "Advertencia",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    // Obtener el usuario seleccionado de la lista
    Usuario usuario = listaUsuario.get(filaSeleccionada);

    // Pasar los datos del usuario a los campos de texto
    jTextFieldNombreC.setText(usuario.getNombreCompleto());
    jTextFieldCorreo.setText(usuario.getCorreo());
    jBoxRol.setSelectedItem(usuario.getRol());
    jFieldContraseña.setText(usuario.getPassword());
}

```

```

private void jButtonEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int fila = this.JTableUsuario.getSelectedRow(); // Se obtiene el #fila seleccionado

    // Se verifica que esté seleccionada una fila de la tabla
    if (fila == -1) {
        JOptionPane.showInputDialog(rootPane, "Seleccione un registro de la tabla");
        return;
    }

    // Se construye el mensaje de confirmación incluyendo el número de fila.
    // Se usa la variable "fila" directamente para que comience desde 0
    int resp = JOptionPane.showConfirmDialog(null, "¿Está seguro de eliminar la fila número " + fila + "?", "Eliminando...", JOptionPane.YES_NO_OPTION,
        JOptionPane.QUESTION_MESSAGE);

    // Si la respuesta es si se elimina
    if (resp == JOptionPane.YES_OPTION) {
        // Se elimina la fila quinta de la lista la fila seleccionada
        // La numeración de la fila de la tabla es igual a las posiciones de la lista
        listaUsuario.remove(fila);
        // Se llama al método llenarTabla para mostrar los datos actuales de la lista
        llenarTabla();
        JOptionPane.showInputDialog(null, "Usuario eliminado correctamente");
    }
}

```

```

private void jButtonActualizarActionPerformed(java.awt.event.ActionEvent evt) {
    int filaSeleccionada = JTableUsuario.getSelectedRow();

    // Validar si el usuario seleccionó una fila
    if (filaSeleccionada == -1) {
        JOptionPane.showInputDialog(null, "Seleccione una fila de la tabla para actualizar",
            "Advertencia",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    // Obtener los nuevos valores desde los campos
    String nombre = jTextFieldNombre.getText().trim();
    String correo = jTextFieldCorreο.getText().trim();
    String rol = jBoxRol.getSelectedItem().toString();
    String contraseña = new String(jPasswordFieldContraseña.getPassword());

    if (nombre.isEmpty() || correo.isEmpty() || contraseña.isEmpty()) {
        JOptionPane.showInputDialog(null, "Por favor, complete todos los campos",
            "Campos Vacíos",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    // Confirmar la actualización
    int confirmacion = JOptionPane.showConfirmDialog(null, "¿Desea actualizar este usuario?",
        "Confirmar Actualización",
        JOptionPane.YES_NO_OPTION);
}

```

```

private void jTextFieldNombreKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();

    // Solo letras y espacio
    if (Character.isLetter(c) || c == KeyEvent.VK_SPACE || c == KeyEvent.VK_BACK_SPACE) {
        evt.consume();
        JOptionPane.showMessageDialog(this, "Solo se permite letras en el nombre");
    }
}

private void jTextFieldCorreοKeyTyped(java.awt.event.KeyEvent evt) {
    char c = evt.getKeyChar();

    // Solo letras y espacio
    if (!Character.isLetter(c) && c != KeyEvent.VK_SPACE && c != KeyEvent.VK_BACK_SPACE) {
        evt.consume();
        JOptionPane.showMessageDialog(this, "Solo se permite letras en el correo");
    }
}

```

```

private void jButtonLimpiarActionPerformed(java.awt.event.ActionEvent evt) {
    // Llama al método limpiar campos
    limpiarCampos();
}

public void mouseClicked(java.awt.event.MouseEvent evt) {
    int fila = JTableUsuario.getSelectedRow();
    if (fila > 0) {
        // Obtener datos del modelo de la tabla
        String nombre = JTableUsuario.getModel().getValueAt(fila, 0).toString();
        String correo = JTableUsuario.getModel().getValueAt(fila, 1).toString();
        String rol = JTableUsuario.getModel().getValueAt(fila, 2).toString();

        // Establecer los datos en los campos de texto y combobox
        jTextFieldNombre.setText(nombre);
        jTextFieldCorreο.setText(correo);
        jBoxRol.setSelectedItem(rol);

        // NOTA: La contraseña no se suele cargar de nuevo en el campo de contraseña
        // por seguridad, se deja vacío para que el usuario la reintroduzca si quiere cambiarla.
        jPasswordFieldContraseña.setText("");
    }
}

```

## 10. Enlace del repositorio GitHub (Commit realizados Imagen)

## 10.1 Commit realizados

### Commits

main ▾

-o- Commits on Nov 18, 2025

- Agrego diapositivas del corte III**  
eg9656517-collab committed 3 days ago
- Agrego evidencias de trabajo colavoretivo**  
eg9656517-collab committed 3 days ago
- Agrego el sistema de escritorio desarrollada en el lenguaje Java**  
eg9656517-collab committed 3 days ago
- Agrego el documento de la metodología que se utilizó**  
eg9656517-collab committed 3 days ago
- Agrego el mapeo de procesos del Software**  
eg9656517-collab committed 3 days ago

-o- Commits on Nov 15, 2025

- Agregando metodología Scrum**  
eg9656517-collab committed last week

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instala la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\laptop\OneDrive\Documents\Proyecto Farmacia\Proyecto para una farmacia> git log --oneline
6a7cf1f (HEAD -> main, origin/main) Agrego diapositivas del corte III
76ecbae Agrego evidencias de trabajo colavoretivo
001c61e Agrego el sistema de escritorio desarrollada en el lenguaje Java
ef0c2bb Agrego el documento de la metodología que se utilizó
085bf4f Agrego el mapeo de procesos del Software
6464cdc Agregando metodología Scrum
PS C:\Users\laptop\OneDrive\Documents\Proyecto Farmacia\Proyecto para una farmacia>
```

## 11. Referencias Bibliográficas

Ambulatoria del Hospital El Carmen de Maipú, para avanzar en la automatización de servicios farmacéuticos

<https://repositorio.uchile.cl/handle/2250/186964>

Ardila Muñoz, S. A. (2024). Diseño, desarrollo e implementación de un componente para las plataformas web y escritorio de la empresa Sistemas y Computadores SYC SA.

<https://repositorio.uts.edu.co:8080/xmlui/handle/123456789/14827>

Agudelo Guisao, R. (2025). Diseño y desarrollo de un software que fomente el acceso a los servicios farmacéuticos asistenciales (dispensación, educación para la salud y farmacovigilancia) en pacientes que consultan Establecimientos Farmacéuticos Minoristas y Servicios Farmacéuticos ambulatorios.

<https://bibliotecadigital.udea.edu.co/entities/publication/9e35bfda-3456-4e7e-835b-900d7a298990>

García, L. (2021). Propiedades antioxidantes de extractos botánicos en el tratamiento de patologías dermatológicas. *Farmacia Moderna*, 15(3), 45-58.

<https://doi.org/xx.xxxx/fm.2021.15.3.45>.

González, M. J. F., Cabrera, C. a. Z., Prado, R. E. S., Japón, G. E. R., & Dávila, K. E. D. (2024). Farmacia Hospitalaria. In *Juan Cuevas eBooks*.

<https://doi.org/10.56470/978-9942-627-95-7>

Verdugo, R. M., Lamas, M. C., Latorre, A. D., Piqueres, R. F., Fernández-Llamazares, C., Vega, E. N., & Goitia, B. T. (2021). Desarrollo de la norma Q-PEX de certificación de calidad de la atención farmacéutica a pacientes externos de los servicios de Farmacia. *Journal of Healthcare Quality Research*, 36(6), 324–332.

<https://doi.org/10.1016/j.jhqr.2021.03.010>

Manrique Rodríguez, D. S. (2025). Machine Lear Ning para la gestión de inventarios de medicamentos en el Hospital José Agurto Tello de Chosica.

<https://repositorio.utp.edu.pe/handle/20.500.12867/11476>

Rojas Raimundo, J. D. (2024). Implementación de un sistema de escritorio para el control de mermas en la empresa Perú Pharma.

<https://repositorio.upla.edu.pe/handle/20.500.12848/7738>

Quisaguano Casa, B. A. (2022). *Desarrollo de sistema para la gestión del inventario en FARMECC: Desarrollo de un sistema de escritorio* (Bachelor's thesis, Quito: EPN, 2022.).

<https://bibdigital.epn.edu.ec/handle/15000/23106>

Villagrán Gómez, C. E. (2021). Implementación de un sistema informático en la farmacia Ambulatoria del Hospital El Carmen de Maipú, para avanzar en la automatización de servicios farmacéuticos.

<https://repositorio.uchile.cl/handle/2250/186964>

Román Ramos, F. J. (2024). Implementación de un sistema informático de control de compra, venta y stock de productos farmacéuticos para la farmacia “El Progreso”–Chimbote; 2024. Minoristas y Servicios Farmacéuticos ambulatorios.

<https://repositorio.uladech.edu.pe/handle/20.500.13032/39882>

Delgado, A. E. M., Rosero, G. A. B., Estrella, K. D. O., & Arias, E. B. (2025). Trayectoria profesional y laboral de los egresados del programa de Tecnología de Regencia de Farmacia. *Boletín Informativo CEI*, 12(1), 33-35. 2024. Minoristas y Servicios Farmacéuticos ambulatorios.

<https://revista.umariana.rdu.co/index.php/BoletinInformativoCEI/article/view/4512>

## 12. Anexos

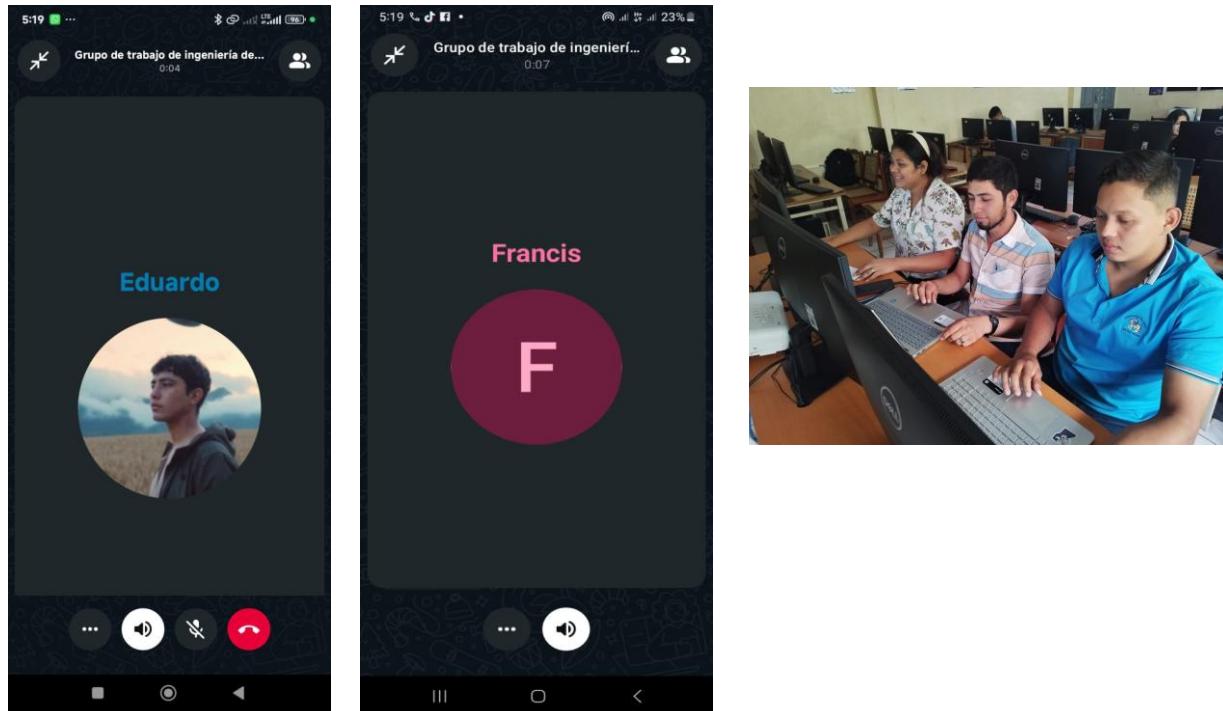
## **12.1 Informe de sesión de trabajo Corte I**

No	Nombres y apellidos	Aportes realizados por integrantes de equipos	Observaciones (incumplimientos llegadas tardes, ausencia, no aporta al proyecto de equipo)
1	Jasser Josafaph Bermúdez Suazo	Tareas o funciones que el sistema automatizará, tomando en cuenta sus características	
2	Ana Gabriela Irías Murillo	Área de la empresa que hará uso del sistema de escritorio	
3	Eduardo Jesús Gudiel González	Objetivos a cumplir (con base en el análisis, diseño y desarrollo de la solución)	
4	Francis Nallely Ayala Calero	Empresa, organización o institución para la cual se desarrollará el sistema de escritorio	

## 12.2 Informe de sesión de trabajo Corte II

No	Nombres y apellidos	Aportes realizados por integrantes de equipos	Observaciones (incumplimientos llegadas tardes, ausencia, no aporta al proyecto de equipo)
1	Jasser Josafaph Bermúdez Suazo	Tabla de roles, historia de usuario de agregar productos y mapeo del software	
2	Ana Gabriela Irías Murillo	Tabla de historia de usuario (consultar productos)	
3	Eduardo Jesús Gudiel González	Tabla de historia de usuario (agregar usuarios, usuarios agregados), requerimientos funcionales, tabla del Sprint Backlog (2, 4), diseño de interfaces, mapeo del software e investigación.	
4	Francis Nallely Ayala Calero	Tabla de historias de usuario (reportes, inicio de sesión, actualizar stock), tabla de requerimientos funcionales, tabla del Sprint Backlog (1, 3 y 5), diseño de interfaces, mapeo del software e investigación.	

### 12.2.1 Evidencias de trabajo

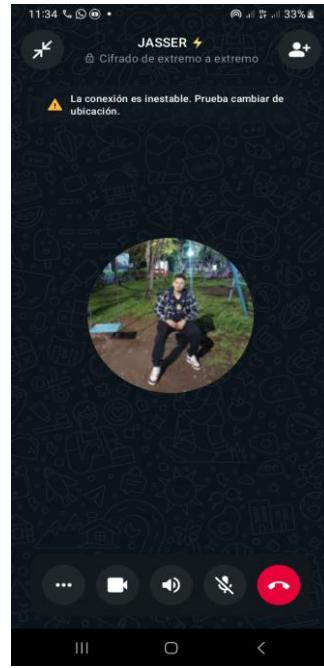


### 12. 3 Informe de sesión de trabajo Corte III

No	Nombres y apellidos	Aportes realizados por integrantes de equipos	Observaciones (incumplimientos llegadas tardes, ausencia, no aporta al proyecto de equipo)
1	Jasser Josaphaph Bermúdez Suazo	Actualizó el mapeo del software y programó el formulario de registrar ventas.	
2	Ana Gabriela Irías Murillo	Diseñó el formulario de actualizar stock e hizo el índice del documento.	

3	Eduardo Jesús Gudiel González	Se encargó de terminar la tabla de requerimientos funcionales, organizó las tablas de historias de usuario, completó el sprint 1 y programó el formulario de Agregar Productos e hizo la conexión del formulario de ventas con el de agregar productos y programó la funcionalidad del formulario actualizar Stock	
4	Francis Nallely Ayala Calero	Se encargó de la organización de los Sprint Backlog 2 y 3 y programó el formulario de Agregar Usuario y Se encargó de la elaboración del BurnDown Chart del Sprint 2.	

### **12.3.1 Evidencias de trabajo**



### **13. Cronograma de actividades.**

Actividades /Tareas	Fechas inicio	Fechas fin	Hora de trabajo	Responsable de actividades
Selección del caso de estudio	09/08/2025	16/08/2025	2hrs	Equipo
Funcionalidades del sistema	17/08/2025	17/08/2025	3hrs	Equipo
objetivos y descripción de la problemática	18/08/2025	18/08/2025	1hrs	Equipo
Mapeo del Software	19/08/2025	19/08/2025	2hrs	Equipo

Planificación Scrum				
Tabla de roles del Equipo de desarrollo	6/09/2025	06/09/2025	2hrs	Equipo
Requerimientos funcionales	8/09/2025	8/09/2025	1hrs	Equipo
Historias de usuarios	11/09/2025	11/09/2025	3hrs	Equipo
Formato de product	15/09/2025	15/09/2025	2hrs	Equipo
Backlog priorizado	17/09/2025	17/09/202	3hrs	Equipo
Formato Sprint Backlog	18/09/2025	18/09/2025	1hrs	Equipo
Interfaz de usuario Figma	20/09/2025	20/09/2025	2hrs	Equipo
Creación de proyecto NetBeans	21/09/2025	21/09/2025	1hrs	Equipo
Paquete controlador	22/09/2025	22/09/2025	3hrs	Equipo
Paquete vista	23/09/2025	23/09/2025	1hrs	Equipo
Paquete modelo	25/09/2025	25/09/2025	3 hrs	Equipo
Referencias bibliográficas	24/09/2025	24/09/202	3hrs	Equipo
Enlace repositorio GitHub	26/09/2025	26/09/202	4hrs	Equipo
Anexos	26/09/2025	26/09/2025	2hrs	Equipo
Informes de sesión de trabajo corte I	27/09/2025	27/09/2025	3hrs	Equipo

Informe de sesión de trabajo corte II	27/09/2025	27/09/202	1hrs	Equipo
Informe de sesión de trabajo corte III	27/09/2025	27/09/202	2hrs	Equipo
Cronograma de actividades	27/09/2025	27/09/202	2hrs	Equipo