



UNIVERSIDAD
SAN IGNACIO
DE LOYOLA

Facultad de Ingeniería

Ingeniería de Sistemas de Información, Software y Ciencia de Datos

**FashionFlow: Lenguajes para Automatización de Producción, Logística y Ventas en
Nancy's Collection**

Trabajo de Investigación

Grupo 2

Ramos Niño Neira Hugo Alexander.....[2310120]

Solari Hipólito, María Sofía.....[2320632]

Veliz García, Alejandra.....[2311640]

Curso:

Compiladores

Sección:

PREISF07C01M

Docente:

Díoses Zárate, Javier Antonio

Lima-Perú

2025-02

Agradecimientos

Queremos expresar nuestro agradecimiento al profesor Javier Antonio Dioses Zárate por su guía durante la realización de este proyecto. Su método didáctico en la enseñanza de Compiladores, en el que los ejercicios propuestos en clase están meticulosamente elaborados para tratar la complejidad intrínseca del curso de forma gradual y comprensible, nos ha hecho capaces de adquirir las habilidades requeridas para aplicar estos conceptos a situaciones reales. Su dominio de ciencias de computación y sistemas se hace evidente en la claridad con que explica conceptos abstractos, como el análisis semántico, sintáctico y léxico.

Además, queremos expresar nuestro agradecimiento a Nancy's Collection por darnos la oportunidad de examinar sus procesos operativos y de sugerir soluciones tecnológicas para su empresa.

Tabla de contenidos

Introducción	2
Descripción del Caso de Estudio	3
Objetivos	7
Objetivo general.....	7
Objetivos específicos	7
Desarrollo.....	8
1. Identificación de requerimientos	8
1.1. Requerimientos de Usuario. (Negocio).....	8
1.2. Requerimientos Funcionales (Sistema).....	10
1.3 Requerimientos No Funcionales	14
2. Solución del Caso	17
2.1. Lenguaje de Producción.....	17
2.2. Lenguaje de Inventario.....	23
2.3. Lenguaje de Ventas	25
• Cliente	25
• Ventas	25
• Gastos.....	25
• Pagos	25
• Salarios.....	25
• Generación de reporte en CSV (Excel).	25
Bibliografía	29

Introducción

Durante los últimos años, las tiendas de ropa siempre han representado un medio para que las personas puedan compartir su autoexpresión personal y cultural, además de desempeñar un papel económico tanto global como localmente. Dentro de la sociedad, la industria textil constituye una fuente importante de empleo en diversas regiones del mundo, brindando oportunidades de trabajo para una amplia variedad de personas. Dentro del ámbito ambiental, muchas empresas del sector optan por tecnologías y procesos más sostenibles, como el uso de materiales sostenibles y la implementación de procesos de producción más eficientes en términos de recursos (AUDACES, 2020).

No obstante, en el Perú la industria textil enfrenta múltiples desafíos, entre ellos la competencia desleal derivada por la importación masiva de prendas, precios subvaluados, contrabando, dumping, pobres condiciones laborales, baja productividad y la falta de innovación e investigación. Asimismo, la pandemia de COVID-19 a inicios de la década de 2020 provocó que el sector caiga en un 31%, evidenciado por su vulnerabilidad ante crisis externas. A estos problemas se suman los constantes problemas de gestión de inventario, ventas y seguridad dentro de las operaciones (La Cámara, 2023).

A pesar de los desafíos ya mencionados, la industria textil continúa avanzando mediante la mejora de condiciones de trabajo, fortalecimiento de la investigación y la incorporación de nuevas tecnologías, como la impresión 3D, con el objetivo de mejorar e incrementar la eficiencia y la flexibilidad de la producción de prendas (AUDACES, 2023). Sin embargo, aún persisten desafíos dentro de 3 rubros fundamentales: Ventas, producción y manufactura y control de corte y confección. En el caso de los emprendimientos textiles, estos procesos pueden resultar lentos debido a la falta de personal o de herramientas tecnológicas adecuadas para la automatización y digitalización. En este sentido, el desarrollo e implementación de sistemas automatizados presenta una alternativa eficiente y viable para la optimización de tareas operativas, reducción de errores humanos y mejora de la eficiencia laboral en la industria textil.

En ese contexto, nuestro presente proyecto tiene como objetivo analizar la situación de la tienda textil llamada Nancy's Collection para diseñar e implementar un lenguaje específico y su respectivo intérprete en C++, orientado a automatizar tareas clave como producción, inventario y ventas. Con ello, buscamos contribuir a la evolución tecnológica continua y a la mejora de la competitividad de la industria textil peruana.

Descripción del Caso de Estudio

1. PRESENTACIÓN GENERAL

Nancy's es una empresa peruana del rubro Confección y Venta Minorista de Ropa, con presencia en cuatro tiendas físicas ubicadas en Lima e Iquitos y una tienda virtual. Se especializa en el diseño, fabricación y comercialización de prendas de vestir femeninas, destacando por su estilo moderno y la atención personalizada hacia sus clientes. Su modelo de negocio combina la producción propia con la venta directa a través de puntos físicos y plataformas digitales.

A pesar de contar con herramientas tecnológicas como TumiPos (para facturación), TumiSoft (para inventarios) y Excel (para reportes administrativos), la empresa enfrenta problemas de registro manual, duplicidad de datos y falta de automatización en áreas críticas de su operación. En particular, los procesos de producción, control de inventario y ventas diarias dependen de registros físicos y cálculos manuales, lo que genera errores, pérdida de trazabilidad y demoras en la toma de decisiones.

2. DESCRIPCIÓN DE ÁREAS INVOLUCRADAS

A continuación, se describen las principales áreas involucradas:

- Área de Producción y Manufactura:

Esta área convierte los diseños autorizados en artículos de vestuario acabados, gestionando las fases de corte, elaboración y acabado hasta llegar al empaquetado definitivo para que se distribuyan a las tiendas. El proceso comprende cerca de 13 pasos que se llevan a cabo en secuencia y que requieren la participación de diversos roles especializados (diseñadora, patronista, cortador, costurero). La información clave sobre las especificaciones técnicas, el número por talla y color, y el consumo de materiales se anota manualmente en cuadernos físicos individuales. Esta fragmentación complica el seguimiento del estado de cada orden de producción, produce que se realicen repetidas consultas entre trabajadores y carece de validaciones automáticas para identificar incongruencias en las cantidades o en las especificaciones.

- Área de Gestión de Inventario:

Encargada de garantizar que cada tienda cuente con la cantidad adecuada de prendas según la demanda y que los movimientos de mercadería estén correctamente registrados y controlados, también tiene un rol importante en la

reposición de productos. Aunque utilizan el sistema TumiSoft, también mantienen registros manuales en cuadernos para los movimientos de ingreso y salida de mercadería. Esta duplicidad provoca inconsistencias y errores de registro. La falta de automatización también dificulta la reposición automática de productos más vendidos, requiriendo verificaciones diarias y manuales.

- Área de Ventas:

Esta área gestiona las transacciones de venta, la facturación electrónica (mediante TumiPos) y el registro y consolidación de datos financieros clave como ventas, pagos, gastos y salarios. Sin embargo, cada vendedor de cada tienda registra manualmente los reportes y los datos clave de las ventas al final del día. El problema que se presenta es que este registro puede requerir más tiempo y posibles incongruencias en caso se cometa un error. El principal desafío es implementar una nueva forma de registrar automáticamente todos los detalles de las ventas directamente al sistema (evitando el registro manual), lo que permitirá garantizar la integridad de los datos, reducir los tiempos de reporte y facilitar la toma de decisiones gerenciales.

3. MÉTODO ACTUAL Y LIMITACIONES

Actualmente, Nancy's combina el uso de sistemas digitales (TumiPos y TumiSoft) con registros manuales en cuadernos y hojas de cálculo. Esta metodología híbrida permite cierta operatividad, pero introduce múltiples limitaciones:

- Duplicidad de información, ya que los mismos datos deben ingresarse tanto en el sistema como en formatos físicos.
- Errores humanos en cálculos de consumo, costeo e inventario.
- Falta de trazabilidad, al no contar con un flujo digital unificado que conecte corte, inventario y ventas.
- Demoras en la generación de reportes administrativos, por depender de consolidaciones manuales en Excel.
- Dificultad para escalar los procesos hacia una gestión más automatizada o digitalizada.

Esta metodología limita la capacidad de Nancy's para tomar decisiones rápidas y basadas en datos confiables, afectando su control de costos, su planeamiento de producción y su eficiencia en la reposición de stock entre tiendas.

4. DELIMITACIÓN DEL ALCANCE

En este estudio, se ha elegido como foco principal los procesos de Control de corte y confección, control de ingreso y salida de mercadería, y registro de datos de ventas, porque representan las áreas más críticas y con mayor potencial de automatización mediante un lenguaje específico.

5. DETALLE DE PROCESOS CRÍTICOS:

- Control de Corte y Confección**

Nancy's Collection gestiona colecciones de diseños propios, elaborados dentro de la empresa. Cada colección comprende entre 15 y 30 diseños únicos, escalonados en cerca de cinco tallas y fabricados en unas dos a cuatro variantes cromáticas. En la actualidad, después de que un diseño ha sido aprobado, el cortador recibe instrucciones verbales o escritas en papel para trazar y calcular manualmente el uso de tela. Posteriormente, esta información se anota en cuadernos físicos sin una estructura definida. La falta de un sistema para producir y rastrear órdenes provoca que se pierda tiempo al tener que comunicar repetidamente las especificaciones, que se detecten errores en las cantidades después de invertir tiempo en el corte y que no sea posible calcular automáticamente los requerimientos totales de tela para varias órdenes a la vez. Se calcula que entre el 10 y el 15% del tiempo del equipo se dedica a aclaraciones y correcciones innecesarias; por otro lado, una falta de visibilidad sobre las necesidades de materiales puede provocar interrupciones en la producción debido a escasez de tela o compras excesivas por estimaciones inexactas, lo cual también impacta negativamente en la exactitud del cálculo de costos de producir artículos de vestuario nuevos y la habilidad para proporcionar información precisa sobre la disponibilidad de productos para las tiendas.

- Control de Ingreso y Salida de Mercadería**

Una vez terminadas las prendas, se envían desde el almacén central a las tiendas. Nancy's Collection, cada día, se estima que se despacha entre 15 y 20 prendas de diferentes modelos y tallas.

El sistema actual consiste en anotar el movimiento de mercaderías tanto en el sistema TumiSoft como en un cuaderno de control físico. Las vendedoras firman la recepción y confirman por WhatsApp.

Se reportaron diversas dificultades derivadas de la coexistencia de registros manuales y digitales. Con frecuencia se producen registros duplicados o inconsistentes entre el sistema TumiSoft y los cuadernos físicos utilizados por las tiendas para anotar los ingresos y salidas de mercadería. Esta duplicidad complica el control y genera discrepancias en las cifras de stock. Asimismo, existe una limitada trazabilidad en la distribución, ya que no siempre se puede identificar con precisión qué lote o paquete fue enviado a cada tienda, lo que retrasa la verificación de entregas y devoluciones. A esto se suma la ausencia de automatización para generar manifiestos o reportes de envío, que hoy se elaboran manualmente en hojas de cálculo. Como consecuencia, se presentan errores frecuentes en los inventarios, diferencias de stock entre el sistema y la realidad física y una pérdida considerable de tiempo en las verificaciones y conciliaciones manuales que deben realizar las vendedoras y la administradora cada semana.

- **Registro de datos de ventas**

Nancy's Collection gestiona el registro de ventas, pagos, gastos y salarios mediante el uso del sistema TumiPos. El proceso consiste en que al final del día, se debe hacer un reporte a través de libros físicos donde debe estar registrado los datos de las ventas realizadas, los gastos realizados por la empresa, los pagos realizados y los salarios ofrecidos al personal, en caso del almacenamiento de datos de clientes y ventas, se cuenta con Excel en cada tienda donde se separa por mes, al finalizar, se transfiere todo al sistema TumiPos. Sin embargo, El problema está en que al realizar el proceso, puede generar múltiples problemas como la inestabilidad de datos, aumento de costos operativos, los riesgos de seguridad del sistema y duplicidad de datos. Estas limitaciones afectan directamente la eficiencia operativa de la empresa, provocando retrasos en las entregas de reportes y dificultad al tomar decisiones gerenciales.

6. JUSTIFICACIÓN FINAL

Estos 3 procesos se consideran críticos porque están interconectados.

El análisis detallado permitirá diseñar un lenguaje específico que unifique la descripción y automatización de estos procesos. Este permitirá que los empleados registren y consulten información mediante instrucciones simples, como generar órdenes de producción, transferir mercadería o crear reportes de ventas.

Objetivos

Objetivo general

Crear un compilador que implemente un lenguaje específico de dominio y su respectivo intérprete en C++, con el fin de reducir el tiempo de tareas administrativas y optimizar los procesos de gestión de producción, inventario y ventas de Nancy's Collection, mediante la integración del análisis léxico, semántico y sintáctico para automatizar la gestión de procesos operativos.

Objetivos específicos

1. Examinar los actuales procedimientos operativos de producción, distribución de inventario y cierre de ventas en Nancy's Collection con el fin de detectar variables, normas comerciales y flujos de información importantes que necesitan ser automatizados.
2. Establecer la gramática formal (lexical, sintáctica y semántica) de tres lenguajes específicos de dominio para el control de inventarios, la gestión de órdenes de producción y el registro diario de ventas, definiendo tokens, reglas de validación y producciones para cada uno.
3. Para cada uno de los tres lenguajes, desarrollar los analizadores léxicos y sintácticos utilizando métodos de reconocimiento de patrones y análisis recursivo descendente, garantizando que la tokenización sea adecuada y que las instrucciones sean validadas en términos estructurales.
4. Crear el módulo de análisis semántico con verificación de tipos, validación de normas comerciales concretas y producción de mensajes de error detallados para cada idioma.
5. Implementar el sistema de traducción dirigida por sintaxis que posibilite la interpretación de las instrucciones del lenguaje y la producción de resultados valiosos para los procesos operativos de Nancy's Collection.
6. Comprobar la funcionalidad de cada lenguaje a través de la ejecución de, por lo menos, dos programas demostrativos por proceso que evidencien situaciones reales en las que se usa Nancy's Collection, asegurando así una interpretación y producción apropiada de resultados.

Desarrollo

1. Identificación de requerimientos

1.1. Requerimientos de Usuario. (Negocio)

1.1.1. Producción. (Alejandra)

ID:	PROD-RU-01
Actor:	Patronista
Descripción:	La patronista DEBE poder registrar una nueva orden de producción especificando código de diseño, tallas, colores y cantidades por combinación.
Criterio de verificación:	La orden queda registrada digitalmente y puede consultarse por el cortador sin usar cuadernos físicos.
Prioridad:	Alta
ID:	PROD-RU-02
Actor:	Cortador
Descripción:	El cortador DEBE poder consultar órdenes de producción asignadas y registrar el consumo de tela utilizado en metros.
Criterio de verificación:	El sistema muestra las especificaciones de corte y almacena el consumo de tela registrado.
Prioridad:	Alta
ID:	PROD-RU-03
Actor:	Costurero
Descripción:	El costurero DEBE poder consultar órdenes cortadas para iniciar confección y registrar cantidades completadas.
Criterio de verificación:	El costurero visualiza órdenes en estado "Cortado" y actualiza progreso de confección.
Prioridad:	Alta
ID:	PROD-RU-04
Actor:	Supervisor de producción
Descripción:	El supervisor DEBE poder verificar que cantidades confeccionadas coincidan con cantidades cortadas, detectando discrepancias automáticamente.
Criterio de verificación:	El sistema compara cantidades y genera reporte de diferencias cuando existen.

Prioridad:	Alta
ID:	PROD-RU-05
Actor:	Administradora
Descripción:	La administradora DEBE poder calcular el costo de producción basándose en consumo de tela y precio por metro.
Criterio de verificación:	El sistema calcula: costo = consumo_tela × precio_por_metro.
Prioridad:	Media

1.1.2. Inventario. (Sofía)

ID:	INV-RU-01
Actor:	Administradora
Descripción:	La administradora debe poder registrar los movimientos de ingreso y salida de mercadería entre el almacén central y las tiendas físicas.
Criterio de verificación:	El sistema acepta las instrucciones y genera un registro visible con los datos de producto, talla, cantidad, origen, destino y fecha.
Prioridad:	Alta
ID:	INV-RU-02
Actor:	Vendedora
Descripción:	La vendedora debe poder consultar en cualquier momento el stock disponible en su tienda y verificar la fecha del último ingreso o transferencia.
Criterio de verificación:	Al ejecutar una instrucción de consulta, el sistema muestra el listado de productos con su cantidad y ubicación actual.
Prioridad:	Alta
ID:	INV-RU-03
Actor:	Administradora
Descripción:	La administradora debe poder generar órdenes de transferencia entre tiendas en función del nivel de stock y la demanda de productos.
Criterio de verificación:	El compilador ejecuta correctamente la instrucción de transferencia y actualiza el inventario de origen y destino.
Prioridad:	Alta

1.1.3. Ventas. (Hugo)

ID:	[VENTAS]-[RU]-[01]
Actor:	Usuario
Descripción:	Un usuario en específico debe generar las listas solicitantes según las instrucciones establecidas en el algoritmo
Criterio de verificación:	El sistema valida si la solicitud está escrita según la instrucción
Prioridad:	Alta
ID:	[VENTAS]-[RU]-[02]
Actor:	Administrador
Descripción:	Un usuario en específico debe generar las listas solicitantes
Criterio de verificación:	El sistema detecta que tipo de usuario está autorizado para la gestión de datos al iniciar sesión
Prioridad:	Alta
ID:	[VENTA]-[RU]-[03]
Actor:	Usuario
Descripción:	Debe visualizar, editar y confirmar las listas generadas por el sistema antes de su envío
Criterio de verificación:	El sistema registra las operaciones realizadas por el usuario
Prioridad:	Media

1.2. Requerimientos Funcionales (Sistema)

1.2.1. Generales.

ID:	GEN-RF-01
Actor:	El compilador
Descripción:	El compilador DEBE implementar un analizador léxico capaz de tokenizar palabras reservadas, identificadores, números y operadores básicos.
Criterio de verificación:	Dado un programa válido, genera secuencia completa de tokens correctamente clasificados.
Prioridad:	Alta
ID:	GEN-RF-02
Actor:	El compilador

Descripción:	El compilador DEBE implementar un analizador sintáctico que valide la estructura gramatical usando análisis descendente recursivo.
Criterio de verificación:	Acepta programas sintácticamente correctos y rechaza inválidos con mensaje de error indicando posición del problema.
Prioridad:	Alta
<hr/>	<hr/>
ID:	GEN-RF-03
Actor:	El sistema
Descripción:	El sistema DEBE implementar análisis semántico que verifique tipos de datos, valores válidos y reglas de negocio.
Criterio de verificación:	Detecta errores como cantidades negativas, valores fuera de rango, e informa el tipo de error específico.
Prioridad:	Alta
<hr/>	<hr/>
ID:	GEN-RF-04
Actor:	El compilador
Descripción:	El compilador DEBE implementar traducción dirigida por sintaxis para interpretar o ejecutar las instrucciones del lenguaje.
Criterio de verificación:	Las construcciones sintácticas válidas producen las acciones esperadas al ejecutarse.
Prioridad:	Alta
<hr/>	<hr/>
1.2.2. Producción. (Alejandra)	
<hr/>	<hr/>
ID:	PROD-RF-01
Actor:	El compilador
Descripción:	El compilador DEBE tokenizar palabras reservadas del lenguaje de producción.
Criterio de verificación:	Identifica correctamente cada palabra reservada y la distingue de identificadores de usuario.
Prioridad:	Alta
<hr/>	<hr/>
ID:	PROD-RF-02
Actor:	El sistema
Descripción:	El compilador DEBE validar sintaxis de declaración de orden.
Criterio de verificación:	Acepta declaraciones bien formadas y rechaza sintaxis incorrecta indicando el problema.
Prioridad:	Alta
<hr/>	<hr/>

ID:	PROD-RF-03
Actor:	El sistema
Descripción:	El sistema DEBE validar que cada orden declare al menos una talla, un color, y que cantidades sean números enteros positivos.
Criterio de verificación:	Rechaza órdenes con cantidades ≤ 0 o con listas vacías, mostrando mensaje específico.
Prioridad:	Alta
ID:	PROD-RF-04
Actor:	El sistema
Descripción:	El sistema DEBE interpretar instrucciones para actualizar el registro de producción.
Criterio de verificación:	Actualiza la orden con consumo de tela registrado y permite consultas posteriores.
Prioridad:	Alta
ID:	PROD-RF-05
Actor:	El sistema
Descripción:	El sistema DEBE calcular costo de materiales aplicando fórmula: consumo \times precio.
Criterio de verificación:	Retorna valor numérico correcto con dos decimales de precisión.
Prioridad:	Alta
ID:	PROD-RF-06
Actor:	El sistema
Descripción:	El sistema DEBE soportar consultas con filtros.
Criterio de verificación:	Retorna únicamente las órdenes que cumplen el criterio especificado.
Prioridad:	Media

1.2.3. Inventario. (Sofía)

ID:	INV-RF-01
Actor:	El compilador
Descripción:	El compilador debe tokenizar correctamente los verbos clave del lenguaje de inventario.

Criterio de verificación:	El analizador léxico identifica todos los tokens válidos y rechaza los símbolos no definidos.
Prioridad:	Alta
ID:	INV-RF-02
Actor:	El compilador
Descripción:	El compilador debe parsear la estructura de la instrucción de transferencia, validando su sintaxis.
Criterio de verificación:	El analizador sintáctico acepta estructuras válidas y produce un error estructurado ante estructuras inválidas.
Prioridad:	Alta
ID:	INV-RF-03
Actor:	El compilador
Descripción:	El compilador debe validar semánticamente que la tienda de origen en una transferencia tenga stock suficiente (mayor que cero) del producto solicitado.
Criterio de verificación:	Si el stock de origen es insuficiente, el análisis semántico produce un mensaje de error detallado, evitando inconsistencias.
Prioridad:	Alta
ID:	INV-RF-04
Actor:	El sistema
Descripción:	El sistema debe generar un manifiesto de envío o código intermedio tras una transferencia exitosa que elimine la necesidad de registros manuales en cuadernos.
Criterio de verificación:	Tras el análisis completo (léxico, sintáctico y semántico), el sistema produce una salida legible y estructurada que refleja los datos de la orden de transferencia, validando la integridad de la instrucción.
Prioridad:	Alta

1.2.4. Ventas. (Hugo)

ID:	[VENTAS]-[RF]-[01]
Actor:	El sistema

Descripción:	El sistema debe generar y mostrar un reporte de todo lo solicitado por el usuario
Criterio de verificación:	El compilador debe reconocer cada valor asignado y solicitado por el usuario
Prioridad:	Alta
ID:	[VENTAS]-[RF]-[02]
Actor:	El sistema
Descripción:	Debe permitir a un usuario en específico la inserción y eliminación de datos de ventas y clientes
Criterio de verificación:	El algoritmo debe dar libertad al usuario al gestionar los datos
Prioridad:	Alta
ID:	[VENTAS]-[RF]-[03]
Actor:	El sistema
Descripción:	Debe enviar una notificación al realizar una acción
Criterio de verificación:	El algoritmo primero debe detectar la acción del usuario antes de mandar la notificación
Prioridad:	Media
ID:	[VENTAS]-[RF]-[04]
Actor:	El sistema
Descripción:	Debe permitir a un usuario en específico la inserción, actualización y eliminación de datos de ventas y clientes
Criterio de verificación:	El sistema debe realizar las operaciones en tiempo real y muestra la versión más reciente de los registros
Prioridad:	Alta
1.3 Requerimientos No Funcionales	
1.3.1 Generales del sistema.	
1.3.1.1 Rendimiento.	
ID:	GEN-RNF-01
Actor:	El sistema
Descripción:	El sistema DEBE compilar y ejecutar programas en menos de 3 segundos.
Criterio de verificación:	Programas de hasta 200 líneas completan en < 3 segundos en hardware estándar (i5, 8GB RAM).

Prioridad:	Media
-------------------	-------

1.3.1.2 Fiabilidad.

ID:	GEN-RNF-02
Actor:	El sistema
Descripción:	El sistema DEBE reportar errores con mensajes descriptivos que indiquen tipo de error, ubicación y sugerencia de corrección.
Criterio de verificación:	Cada error (léxico, sintáctico, semántico) genera mensaje comprensible con línea/columna del problema.
Prioridad:	Alta

1.3.1.3 Usabilidad.

ID:	GEN-RNF-03
Actor:	El sistema
Descripción:	La sintaxis del lenguaje DEBE usar palabras en español y estructura similar a pseudocódigo.
Criterio de verificación:	Usuarios con conocimientos básicos pueden escribir programas simples después de ver ejemplos.
Prioridad:	Media

1.3.1.4 Mantenibilidad.

ID:	GEN-RNF-04
Actor:	El sistema
Descripción:	El código DEBE estar modularizado en componentes: analizador léxico, sintáctico, semántico, con interfaces claras.
Criterio de verificación:	Cada módulo puede modificarse sin afectar otros módulos (bajo acoplamiento).
Prioridad:	Media

1.3.1.5 Portabilidad.

ID:	GEN-RNF-05
Actor:	El sistema

Descripción:	El compilador DEBE ejecutarse en Windows, Linux y macOS sin modificaciones.
Criterio de verificación:	El mismo código compila exitosamente en los tres sistemas operativos.
Prioridad:	Media

1.3.2 Específicos por módulo.

1.3.2.1 Producción.

ID:	PROD-RNF-01
Actor:	El sistema
Descripción:	Los cálculos de costo DEBEN tener precisión de dos decimales para exactitud contable.
Criterio de verificación:	Todos los cálculos monetarios se redondean a 2 decimales.
Prioridad:	Alta
ID:	PROD-RNF-02
Actor:	El sistema
Descripción:	El sistema DEBE validar que consumo de tela sea número decimal positivo en metros.
Criterio de verificación:	Rechaza valores negativos o no numéricos con mensaje de formato esperado.
Prioridad:	Media
ID:	PROD-RNF-03
Actor:	El sistema
Descripción:	Las operaciones de actualización DEBEN ser atómicas para evitar inconsistencias.
Criterio de verificación:	Si una actualización falla, la orden mantiene su estado anterior sin corrupción parcial.
Prioridad:	Alta

1.3.2.2 Inventario.

ID:	INV-RNF-01
Tipo:	Fiabilidad

Descripción:	La interpretación del lenguaje de inventario deberá mantener la consistencia entre el stock físico y el registrado, eliminando la duplicidad de datos en cuadernos físicos.
Criterio de verificación:	La discrepancia entre el stock reportado en TumiSoft y el stock físico debe ser inferior al 1% en el plazo de una semana.
Prioridad:	Alta
ID:	INV-RNF-02
Tipo:	Usabilidad
Descripción:	El lenguaje deberá usar comandos intuitivos que reduzcan la curva de aprendizaje de la administradora y las vendedoras al usar el sistema.
Criterio de verificación:	El tiempo promedio para ingresar y validar una transferencia de stock se reduce en un 50% con respecto al método manual actual.
Prioridad:	Media

2. Solución del Caso

2.1. Lenguaje de Producción

2.1.1. Diseño del lenguaje. (Alejandra)

El lenguaje de producción está diseñado para automatizar el registro y consulta de órdenes de producción en Nancy's Collection, específicamente desde el paso 6 (escalado por tallas) hasta el paso 11 (verificación de cantidad confeccionada) del proceso documentado en la entrevista. El objetivo es reemplazar los cuadernos físicos actuales donde se anotan las especificaciones técnicas, el número por talla y color, y el consumo de materiales, eliminando la fragmentación de información y las validaciones manuales propensas a error.

Características del Lenguaje

1. Paradigma: Imperativo, orientado a instrucciones secuenciales que reflejan el flujo de trabajo real de Nancy's Collection (Nystrom, 2021, Cap. 8: Statements and State).
2. Tipado: Estático fuerte para cantidades (enteros) y consumo de tela (decimales), evitando errores de cálculo identificados en el caso de estudio.
3. Dominio: Específico para operaciones de producción textil: creación de órdenes, registro de corte con consumo de tela, consulta de órdenes por estado, y cálculo de costos de materiales.

Elementos Léxicos

1. **Palabras reservadas:** ORDEN, DISEÑO, TALLAS, COLORES, CANTIDADES, CORTE, CONSUMO, CONSULTAR, DONDE, CALCULAR_COSTO, PRECIO_TELA.
2. **Literales:** Identificadores alfanuméricos (códigos de diseño), enteros positivos (cantidades), decimales positivos (metros de tela, precios).
3. **Operadores:** == (comparación para filtros), : (asignación de atributos), , (separador).
4. **Delimitadores:** [] (listas de tallas/colores), { } (cuerpo de instrucciones).

Sintaxis Principal

Declaración de Orden de Producción

```
ORDEN <identificador> {
    DISEÑO: <codigo>
    TALLAS: [<talla1>, <talla2>, ...]
    COLORES: [<color1>, <color2>, ...]
    CANTIDADES: [<cant_T1C1>, <cant_T1C2>, ..., <cant_T2C1>, ...]
}
```

Esta estructura captura la información que actualmente "se anota manualmente en cuadernos físicos sin una estructura definida" (Avance 1, p.7), proporcionando validación automática de consistencia entre dimensiones.

Registro de Corte

```
CORTE <id_orden> {
    CONSUMO: <metros_tela>
}
```

Digitaliza el paso 8 del proceso: "El cortador al realizar el trazo indica cuanta tela entra en el modelo", permitiendo cálculos automáticos de costeo.

Consulta de Órdenes

```
CONSULTAR ORDENES DONDE ESTADO == <valor>
```

Soluciona el problema documentado de "repetidas consultas entre trabajadores" al centralizar la información en formato consultable.

Cálculo de Costo

```
CALCULAR_COSTO <id_orden> PRECIO_TELA: <valor>
```

Automatiza el costeo mencionado en el paso 8, aplicando la fórmula: costo = consumo_tela × precio_por_metro.

Semántica y Reglas de Validación

- **Coherencia dimensional:** La cantidad de elementos en CANTIDADES debe ser igual a |TALLAS| × |COLORES|.
- **Positividad:** Todas las cantidades y medidas deben ser > 0.

- **Tipos:** CANTIDADES admite solo enteros; CONSUMO y PRECIO_TELA admiten decimales con máximo 2 decimales.
- **Secuencialidad:** No se puede registrar CORTE para una orden inexistente.

Ejemplo Funcional

```
ORDEN ORD-2025-001 {
    DISEÑO: BLUSA-VER-023
    TALLAS: [S, M, L, XL]
    COLORES: [Azul, Blanco]
    CANTIDADES: [10, 12, 8, 15, 5, 10, 6, 8]
}

CORTE ORD-2025-001 {
    CONSUMO: 45.50
}

CALCULAR_COSTO ORD-2025-001 PRECIO_TELA: 22.00

CONSULTAR ORDENES DONDE ESTADO == Cortado
```

Este programa crea una orden con $4 \text{ tallas} \times 2 \text{ colores} = 8$ combinaciones (validación dimensional), registra 45.50 metros de tela consumidos, calcula el costo ($45.50 \times 22.00 = 1,001.00$), y consulta órdenes en estado "Cortado".

2.1.2. Análisis léxico. (Alejandra)

Categorías de Tokens

1. Palabras Reservadas (Keywords)

Identificadores con significado especial en el lenguaje que no pueden usarse como nombres de variables.

Token	Lexema	Descripción
KW_ORDEN	ORDEN	Declara nueva orden de producción
KW_DISEÑO	DISEÑO	Especifica código de diseño
KW_TALLAS	TALLAS	Lista de tallas a producir
KW_COLORES	COLORES	Lista de colores disponibles
KW_CANTIDADES	CANTIDADES	Matriz de cantidades por talla-color
KW_CORTE	CORTE	Registra operación de corte
KW_CONSUMO	CONSUMO	Metros de tela consumidos
KW_CONSULTAR	CONSULTAR	Inicia consulta de órdenes
KW_ORDENES	ORDENES	Tipo de entidad a consultar
KW_DONDE	DONDE	Introduce condición de filtro
KW_ESTADO	ESTADO	Atributo de estado de orden
KW_CALCULAR_COSTO	CALCULAR_COSTO	Calcula costo de materiales
KW_PRECIO_TELA	PRECIO_TELA	Precio por metro de tela

Las palabras reservadas se identifican mediante tabla hash después de reconocer un identificador. Si el lexema coincide con entrada en la tabla, se clasifica como keyword; de lo contrario, es un identificador de usuario.

2. Identificadores (Identifiers)

Nombres definidos por el usuario para órdenes, diseños, tallas y colores.

Token	Patrón	Ejemplo
ID	[a-zA-Z][a-zA-Z0-9_-]*	ORD-2025-001, BLUSA-VER-023, Azul

Validación semántica posterior: Los identificadores de orden deben seguir formato ORD-YYYY-NNNN (validado en fase semántica, no léxica).

3. Literales Numéricos

Representan cantidades y medidas en el dominio textil.

Token	Patrón	Descripción	Ejemplo
NUM_ENTERO	[0-9]+	Cantidades de prendas (positivos)	10, 45
NUM_DECIMAL	[0-9]+\.[0-9]{1,2}	Metros de tela, precios (máx 2 decimales)	45.50, 22.00

Nota: La validación de positividad (valor > 0) se realiza en fase semántica, no léxica.

4. Operadores

Símbolos para comparación y asignación.

Token	Lexema	Uso
OP_IGUAL_IGUAL	==	Comparación en filtros (DONDE ESTADO == Cortado)
OP_DOS_PUNTOS	:	Asignación de atributos (DISEÑO: código)

5. Delimitadores

Agrupan elementos y separan componentes.

Token	Lexema	Uso
LBRACE	{	Inicio de bloque
RBRACE	}	Fin de bloque
LBRACKET	[Inicio de lista/array
RBRACKET]	Fin de lista/array
COMMA	,	Separador de elementos
EOF	(fin de archivo)	Marca final de entrada

6. Espacios y Comentarios (Ignorados)

Patrón	Descripción

[\t\r\n]+	Espacios en blanco, tabs, saltos de línea (ignorados)
//[^n]*	Comentario de línea (ignorado)

Autómata Finito Determinista (AFD)

El reconocimiento de tokens se implementa mediante un AFD implícito en el código del scanner. Estados principales:

```

INICIO → (letra) → IDENTIFICANDO_ID → (letra|dígito|_|-) →
IDENTIFICANDO_ID
                                         → (otro) → [retroceso, clasifica ID
o KW]

INICIO → (dígito) → IDENTIFICANDO_NUM → (dígito) → IDENTIFICANDO_NUM
                                         → (.) → IDENTIFICANDO_DECIMAL →
(dígito) → IDENTIFICANDO_DECIMAL
                                         → (otro) → [retroceso, emite
NUM_ENTERO]

INICIO → (:) → [emite OP_DOS_PUNTOS]
INICIO → (=) → ESPERANDO_SEGUNDO_IGUAL → (=) → [emite OP_IGUAL_IGUAL]
                                         → (otro) → [ERROR: '=' solo no
válido]
INICIO → ({) → [emite LBRACE]
INICIO → ([]) → [emite LBRACKET]

```

Retroceso (lookahead): Cuando se reconoce un identificador seguido de un carácter no alfanumérico, se retrocede un carácter en el stream de entrada para que el siguiente token lo procese correctamente (Nystrom, 2021, p. 63).

Tabla de Símbolos de Palabras Reservadas

```

unordered_map<string, TokenType> keywords;
keywords["ORDEN"] = KW_ORDEN;
keywords["DISEÑO"] = KW_DISEÑO;
keywords["TALLAS"] = KW_TALLAS;
keywords["COLORES"] = KW_COLORES;
keywords["CANTIDADES"] = KW_CANTIDADES;
keywords["CORTE"] = KW_CORTE;
keywords["CONSUMO"] = KW_CONSUMO;
keywords["CONSULTAR"] = KW_CONSULTAR;
keywords["ORDENES"] = KW_ORDERES;
keywords["DONDE"] = KW_DONDE;
keywords["ESTADO"] = KW_ESTADO;
keywords["CALCULAR_COSTO"] = KW_CALCULAR_COSTO;
keywords["PRECIO_TELA"] = KW_PRECIO_TELA;

```

2.1.3. Análisis sintáctico. (Alejandra)

El análisis sintáctico (parsing) toma la secuencia de tokens del analizador léxico y verifica que la estructura del programa sea válida según la gramática del lenguaje (Nystrom, 2021, Cap. 6: Parsing Expressions). Este proceso construye un árbol sintáctico abstracto (AST) que representa la estructura jerárquica del programa y sirve como entrada para el análisis semántico y la generación de código.

Nuestro parser utiliza análisis descendente recursivo (Recursive Descent Parsing), una técnica donde cada regla de producción de la gramática se implementa como una función recursiva que reconoce esa construcción sintáctica (Nystrom, 2021, p. 139).

Gramática Formal del Lenguaje de Producción

La gramática es LL(1) (Left-to-left, Leftmost derivation, 1 token lookahead), lo que permite implementar un parser descendente sin retroceso.

```

Programa      → Instrucción* EOF

Instrucción   → Declaracion_Orden
                | Registro_Corte
                | Consulta
                | Calculo_Costo

Declaracion_Orden → KW_ORDEN ID LBRACE
                    KW_DISEÑO OP_DOS_PUNTOS ID
                    KW_TALLAS OP_DOS_PUNTOS Lista_Valores
                    KW_COLORES OP_DOS_PUNTOS Lista_Valores
                    KW_CANTIDADES OP_DOS_PUNTOS Lista_Valores
                    RBRACE

Registro_Corte → KW_CORTE ID LBRACE
                  KW_CONSUMO OP_DOS_PUNTOS NUM_DECIMAL
                  RBRACE

Consulta       → KW_CONSULTAR KW_ORDENES KW_DONDE KW_ESTADO
OP_IGUAL_IGUAL ID

Calculo_Costo → KW_CALCULAR_COSTO ID KW_PRECIO_TELA OP_DOS_PUNTOS
NUM_DECIMAL

Lista_Valores → LBRACKET Valor (COMMA Valor)* RBRACKET

Valor          → ID
                | NUM_ENTERO
                | NUM_DECIMAL

```

Notación:

1. * : Cero o más repeticiones

2. | : Alternativa (o)
3. (...) : Agrupación
4. Mayúsculas: Tokens terminales del analizador léxico
5. Minúsculas: No terminales (reglas de producción)

Conjuntos FIRST y FOLLOW

Los conjuntos FIRST y FOLLOW son fundamentales para construir la tabla de análisis predictivo LL(1).

FIRST (primeros tokens que pueden aparecer):

```

FIRST(Programa) = {KW_ORDEN, KW_CORTE, KW_CONSULTAR, KW_CALCULAR_COSTO,
EOF}
FIRST(Instrucción) = {KW_ORDEN, KW_CORTE, KW_CONSULTAR,
KW_CALCULAR_COSTO}
FIRST(Declaracion_Orden) = {KW_ORDEN}
FIRST(Registro_Corte) = {KW_CORTE}
FIRST(Consulta) = {KW_CONSULTAR}
FIRST(Calculo_Costo) = {KW_CALCULAR_COSTO}
FIRST(Lista_Valores) = {LBRACKET}
FIRST(Valor) = {ID, NUM_ENTERO, NUM_DECIMAL}

```

FOLLOW (tokens que pueden aparecer después):

```

FOLLOW(Programa) = {$}
FOLLOW(Instrucción) = {KW_ORDEN, KW_CORTE, KW_CONSULTAR,
KW_CALCULAR_COSTO, EOF}
FOLLOW(Lista_Valores) = {KW_DISEÑO, KW_TALLAS, KW_COLORES,
KW_CANTIDADES, RBRACE}
FOLLOW(Valor) = {COMMA, RBRACKET}

```

2.2. Lenguaje de Inventario

2.2.1. Diseño del lenguaje. (Sofia)

El lenguaje de inventario ha sido diseñado para optimizar la gestión digital de los movimientos de mercadería dentro del sistema de Nancy's Collection, abarcando las operaciones de ingreso, transferencia, consulta y exportación de datos de inventario entre el almacén central y las distintas tiendas físicas.

Este lenguaje propone reemplazar los registros manuales y cuadernos físicos mediante instrucciones estructuradas, claras y legibles por el compilador, con el fin de reducir errores humanos y mejorar la trazabilidad de cada transacción.

Su diseño se centra en la automatización de cuatro procesos principales:

- Transferencia de mercadería entre tiendas o entre almacén y tienda.
- Ingreso de nuevas prendas al inventario.
- Consulta del stock actual o histórico.
- Exportación de reportes o manifiestos en formatos digitales (por ejemplo, CSV).

Cada instrucción se define dentro de un bloque identificado por una palabra clave principal (TRANSFERIR, INGRESAR, CONSULTAR o EXPORTAR) y delimitado por llaves {}, con un conjunto de atributos y valores que describen los detalles específicos de la operación.

2.2.2. Análisis léxico. (Sofía)

El análisis léxico del lenguaje de inventario tiene como propósito identificar y clasificar los componentes básicos de las instrucciones escritas en formato estructurado. Cada bloque representa una operación de inventario, delimitada por llaves {} y conformada por pares atributo-valor separados por dos puntos (:).

Esta fase está implementada en el archivo lexer_inventario.cpp, donde el analizador recorre el archivo fuente carácter por carácter, transformando las palabras, números y símbolos en tokens que pueden ser interpretados posteriormente por el analizador sintáctico.

El analizador léxico reconoce los siguientes tipos de elementos:

- Palabras clave: TRANSFERIR, INGRESAR, CONSULTAR, EXPORTAR, PREnda, TALLA, CANTIDAD, DE, A, EN, FORMATO, FECHA, CODIGO, DESCRIPCION.
- Símbolos estructurales: {}, :, .
- Cadenas de texto (STRING): valores entre comillas dobles "...", utilizados para nombres de productos, tiendas, descripciones o formatos.
- Números (NUMBER): valores numéricos asociados a cantidades de prendas u otros datos cuantitativos.
- Tokens desconocidos (UNKNOWN): cualquier carácter o palabra que no pertenece al vocabulario definido del lenguaje.

2.2.3. Análisis sintáctico. (Sofía)

El análisis sintáctico, implementado en el archivo parser_inventario.cpp, tiene como propósito verificar que la secuencia de tokens generada por el analizador léxico cumpla con las reglas gramaticales definidas para el lenguaje de inventario.

Cada instrucción principal (TRANSFERIR, INGRESAR, CONSULTAR y EXPORTAR) representa una operación específica del sistema de control de inventario y debe seguir una estructura sintáctica bien definida basada en bloques delimitados por llaves {} y pares clave-valor.

El parser realiza una lectura secuencial de los tokens emitidos por el lexer, comprobando que:

- Cada bloque empieza con una palabra reservada válida.
- El contenido del bloque esté delimitado correctamente por {} y :.
- Cada atributo obligatorio (por ejemplo, PREnda, TALLA, CANTIDAD, DE, A, EN, FORMATO, FECHA, CODIGO,

DESCRIPCION) esté seguido del símbolo : y de un valor correspondiente, el cual puede ser una cadena de texto (STRING) o un número (NUMBER) según el tipo de dato esperado.

En caso de que la estructura no se cumpla, el parser genera mensajes de error específicos, indicando el tipo de problema detectado, como la falta de una llave de apertura o cierre, ausencia de valor, o uso incorrecto de una palabra reservada.

Además, el analizador está diseñado con un mecanismo de recuperación de errores, que le permite continuar el proceso de análisis incluso si se detectan inconsistencias dentro de un bloque, garantizando así una revisión completa del archivo fuente.

2.3. Lenguaje de Ventas

2.3.1 Diseño del lenguaje. (Hugo)

El lenguaje de Gestión de clientes y ventas ha sido diseñado para generar automáticamente un archivo Excel con los datos ingresados dentro del sistema de Nancy's Collection, abarcando las categorías de clientes, ventas, gastos, pagos y salarios.

Este lenguaje propone un método más eficiente y rápida para reemplazar los métodos manuales de registros manuales mediante instrucciones estructuradas, claras y legibles por el compilador.

Su diseño se centra en la generación automática de datos de los siguientes rubros:

- Cliente
- Ventas
- Gastos
- Pagos
- Salarios
- Generación de reporte en CSV (Excel).

```
enum class TokenType {
    //Palabras clave
    REGISTRAR_CLIENTE, REGISTRAR_VENTA, REGISTRAR_GASTO, REGISTRAR_PAGO, REGISTRAR_SALARIO, GENERAR_Reporte,
    //Simbolos
    PARENTESIS_IZQ, PARENTESIS_DER, COMA, FIN_SETENCIA,
    //Literales
    CADENA, NUMERO,
    //Control
    FIN_ARCHIVO, ERROR
};
```

Cada instrucción está representada por una palabra clave principal que equivale a cada rubro identificado (REGISTRAR_CLIENTE, REGISTRAR_VENTA, REGISTRAR_GASTO, REGISTRAR_PAGO, REGISTRAR_SALARIO, GENERAR_Reporte) y delimitado por llaves {}, con un conjunto de símbolos, literales y el control del algoritmo.

2.3.2. Análisis léxico. (Hugo)

El análisis léxico del lenguaje de gestión de ventas y clientes tiene como objetivo identificar y clasificar cada componente de las instrucciones escritas estructuradamente, cada

una de ellas representada por cada rubro identificado para la generación del registro y delimitado por llaves {} y conformado por pares atributo-valor separados por puntos y comas (“;”).

El proceso esta implementado como lexer_ventas.cpp. Donde el analizador se encarga de recorrer el algortimo principal carácter por carácter, transformando todos los valores en tokens que pueden ser interpretados por el analizador sintáctico (parser)

- Palabras clave:
 - REGISTRAR_CLIENTE
 - REGISTRAR_VENTA
 - REGISTRAR_GASTO
 - REGISTRAR_PAGO
 - REGISTRAR_SALARIO
 - GENERAR_Reporte

```
//Palabras Clave
if (isalpha(c)) {
    string palabra;
    while (isalpha(peek()) || peek() == '_') {
        palabra += advance();
    }
    if (palabra == 'REGISTRAR_CLIENTE') {
        return { TokenType::REGISTRAR_CLIENTE, palabra, linea, columna };
    }
    if (palabra == 'REGISTRAR_VENTA') {
        return { TokenType::REGISTRAR_VENTA, palabra, linea, columna };
    }
    if (palabra == 'REGISTRAR_GASTO') {
        return { TokenType::REGISTRAR_GASTO, palabra, linea, columna };
    }
    if (palabra == 'REGISTRAR_PAGO') {
        return { TokenType::REGISTRAR_PAGO, palabra, linea, columna };
    }
    if (palabra == 'REGISTRAR_SALARIO') {
        return { TokenType::REGISTRAR_SALARIO, palabra, linea, columna };
    }
    if (palabra == 'GENERAR_Reporte') {
        return { TokenType::GENERAR_Reporte, palabra, linea, columna };
    }
    return { TokenType::ERROR, palabra, linea, columna };
}
```

- Símbolos estructurales:
 - (
 -)
 - ,
 - ;

```

Token Token_Next() {
    espacio();
    if (pos >= fuente.size()) {
        return { TokenType::FIN_ARCHIVO, "", linea, columna };
    }
    char c = peek();

    //Simbolos
    if (c == '(') {
        advance();
        return { TokenType::PARENTESIS_IZQ, "(", linea, columna };
    }
    if (c == ')') {
        advance();
        return { TokenType::PARENTESIS_DER, ")", linea, columna };
    }
    if (c == ',') {
        advance();
        return { TokenType::COMA, ",", linea, columna };
    }
    if (c == ';') {
        advance();
        return { TokenType::FIN_SENTENCIA, ":", linea, columna };
    }
}

```

- Números: valores numéricos asociados a los montos de ventas, gastos, pagos y salarios

```

//Numeros
if (isdigit(c)) {
    string num;
    while (isdigit(peek()) || peek() == '=') {
        num += advance();
    }
    return { TokenType::NUMERO, num, linea, columna };
}

```

- Cualquier carácter desconocido, emitirá un mensaje de error que indica que no existe tal carácter.

```

//No hay Coincidencia
string arr(1, c);
advance;
return { TokenType::ERROR, arr, linea, columna };

```

2.3.3. Análisis sintáctico. (Hugo)

El análisis sintáctico del algoritmo de gestión de clientes y ventas tiene como objetivo verificar que la secuencia de tokens generada por el algoritmo lexer cumpla con las

instrucciones definidas para el lenguaje de gestión. Siendo que cada palabra clave represente cada rubro a registrar de la gestión y debe seguir la estructura sintáctica bien definida basada en bloques delimitados por llaves {} y pares clave-valor.

El algoritmo parser se encarga de realizar una lectura secuencial de los tokens emitidos por el lexer, comprobando que:

- Cada bloque inicie con una palabra reservada (ID)
- El contenido debe estar delimitado por comas (,).
- Cada valor asignado para cada rubro debe estar seguido por una coma y de un valor correspondiente (Monto, detalle, nombre, fecha, estado) el cual puede ser un valor string o un numero según el tipo de datos esperado.

En caso de que no se cumpla, el parser generar mensajes de error que indica en que linea no existe tal valor o si hay algún problema detectado dentro de los datos solicitados, como valores equivocados, falta de llaves, etc.

Bibliografía

Audaces. (3 de diciembre de 2020). Industria textil: ¿cuál es su importancia en el mercado? <https://audaces.com/es/blog/industria-textil>

Audaces. (20 de septiembre de 2023). Sector textil: conoce sus avances tecnológicos en todo el mundo. <https://audaces.com/es/blog/seCTOR-textil>

Drew. (2023). Automatización en la industria textil: Revolución de la moda. Wearedrew.co. <https://blog.wearedrew.co/industria-textil/automatizacion-en-la-industria-textil-revolucion-de-la-moda>

Perú 21. (13 de abril de 2025). Industria textil peruana en alerta por nueva ola de importaciones tras aranceles de EE.UU. <https://peru21.pe/economia/industria-textil-en-alerta-por-nueva-ola-de-importaciones-tras-aranceles/>

Crafting Interpreters. (2015). Craftinginterpreters.com. <https://craftinginterpreters.com/>