

Proyecto-FDP

Integrantes:

- Ian Enrique Frausto Cervantes (Senior)
  - Leonardo Pérez Molina (Junior)
  - Nallely Maldonado Ramírez (Junior)
  - Bryan Alexis Molina Loya (Mid)
  - Oscar Gabriel Reyes Delgado (Mid)

Profesor: Edgar Iván Patricio Aizpuru

## Contexto y Problema

### Escenario:

El escenario planteado para nuestro proyecto es dentro de una empresa para la realización de un programa con el fin de que facilitemos las actividades relacionadas con la toma de un inventario, como facilitar los datos ingresados dentro de una lista para el conocimiento de los materiales disponibles, su cantidad actual, el costo de cada uno y el presupuesto total de todos y cada uno de ellos, con el fin de hacer todo de una manera fácil de entender, concisa y práctica.

### Problemas a resolver:

Nuestras problemáticas a resolver son las siguientes:

- Captura de datos de manera fácil y optima para registrarlos dentro de una lista para un conocimiento de los mismos.
- Un costo total dentro de todos los materiales listados para una cotización y control de los mismos.
- Mostrar los datos capturados de forma clara para el mantenimiento de los mismos en el caso de ingresar más material a los ya existentes o una reducción de ellos.

### Criterios de exito

- El programa debe permitir la captura de datos de manera sencilla y sin errores.
- La lista de inventario debe mostrarse de forma clara y organizada.
- El cálculo del costo total de los materiales debe ser preciso.
- El sistema debe ser fácil de usar para cualquier usuario sin necesidad de conocimientos avanzados de programación.
- El programa debe permitir actualizar, añadir o reducir materiales en el inventario sin afectar los registros previos.

## Reglas del negocio

### Reglas de identificación de productos (Unicidad)

- Código Único: El código de un producto no debe existir previamente en el inventario al integrar registrar uno nuevo. (try\_codigo)
- Nombre Único (ignorando mayúsculas): El nombre de un producto (sin importar mayúsculas/minúsculas o espacios iniciales/finales) no debe existir previamente en el inventario al registrar uno nuevo. (try\_nombre(), normalizar())

### Reglas de Validacion de Datos de Entrada

- Cantidad No Negativa: La cantidad de cualquier producto no puede ser un número negativo (debe ser  $\geq 0$ ). (try\_int())
- Costo No Negativo: El costo individual de un producto no puede ser un número negativo (debe ser  $\geq 0$ ). (try\_float())
- Campos Requeridos (Código y Nombre): El código y el nombre de un producto no pueden estar vacíos al ingresarse. (try\_codigo(), try\_nombre())
- Formato de inventario: Cada registro de un producto debe contener exactamente 4 campos (código, nombre, cantidad, costo). (cargar\_inventario())

### Reglas de Nomenclatura y Restricciones de Archivos

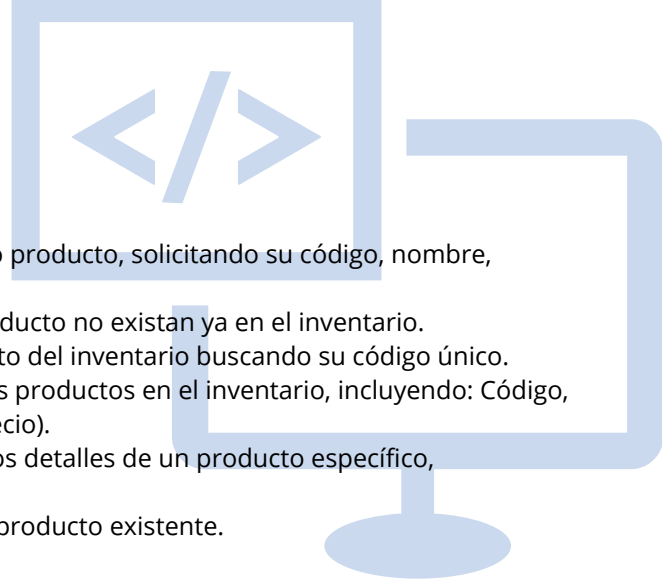
- Formato de Archivo de Inventario: El nombre de un archivo de inventario de seguir el formato. "[nombre\_input]-[día]-[mes]-[año].txt". (Lógica principal del while True)

## Requerimientos Funcionales

### Requisitos funcionales del Sistema de Inventario

#### Gestión de Archivos y Configuración Inicial

1. Carga/Selección de Archivo: El sistema debe permitir al usuario ingresar el nombre de un archivo de inventario y, opcionalmente, una fecha para buscar un archivo de inventario histórico.
2. Creación de Archivo: Si el archivo solicitado para cargar no existe, el sistema debe crear un nuevo archivo de inventario para guardar los cambios, utilizando el nombre de archivo base provisto por el usuario y la fecha actual.
3. Detección de Archivo Prohibido: El sistema debe impedir que el usuario use el nombre del reporte final del día actual (Reporte Final-d-m-a.txt) como el nombre base para el archivo de inventario.
4. Carga de Datos: El sistema debe poder leer y parsear los datos de un archivo de inventario (separados por comas) y cargarlos en la memoria para su procesamiento.



## Funcionalidades del CRUD (Crear, Leer, Actualizar, Borrar)

1. Agregar Producto: El sistema debe permitir al usuario ingresar un nuevo producto, solicitando su código, nombre, cantidad y costo unitario.
- Validación de Unicidad: Debe verificar que el código y el nombre del producto no existan ya en el inventario.
2. Quitar Producto: El sistema debe permitir al usuario eliminar un producto del inventario buscando su código único.
3. Ver Inventario: El sistema debe mostrar un listado completo de todos los productos en el inventario, incluyendo: Código, Producto, Cantidad, Precio Unitario y el Total de su valor (Cantidad × Precio).
4. Buscar Producto: El sistema debe permitir al usuario buscar y mostrar los detalles de un producto específico, identificándolo por su código.
- Actualizar Producto: El sistema debe permitir modificar los datos de un producto existente.

## Procesamiento y Reportes

1. Cálculo del Valor Total: El sistema debe calcular y mostrar la suma del valor de todo el inventario (la sumatoria de Cantidad × Costo para todos los productos).
2. Generación de Reporte Final: El sistema debe ser capaz de generar un archivo de texto de reporte (Reporte Final-d-m-a.txt) que contenga:
  - Número total de productos únicos.
  - Valor total del inventario.
  - Una tabla detallada con el Código, Nombre, Cantidad, Costo y Sub-Total por cada producto.

## Interacción con el Usuario

1. Menú de Opciones: El sistema debe presentar un menú con las opciones para acceder a cada una de las funcionalidades descritas.
2. Manejo de Errores en Entrada: El sistema debe manejar errores de ValueError (como ingresar texto donde se espera un número) y solicitar la reintroducción de datos.
3. Validación de Rangos: El sistema debe rechazar entradas inválidas como cantidades o costos negativos.
4. Retroalimentación: El sistema debe mostrar mensajes de carga y éxito (cargando()) para indicar al usuario que las operaciones se están llevando a cabo.

## Diseño de entradas y salidas

### Diseño de Entradas (Inputs)

Las entradas son los datos que el sistema consume, ya sea provistos por el usuario o cargados desde un archivo.

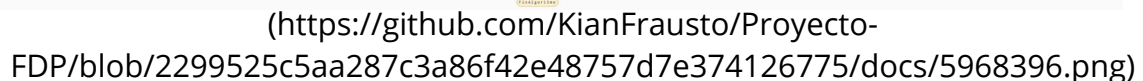
1. Entrada de Archivo (Persistencia de Datos) El sistema espera que los archivos de inventario (\*.txt) sigan un formato CSV plano (valores separados por comas), donde cada línea reopresenta un producto.
  - Estructura de Línea: "[Código],[Nombre],[Cantidad],[Costo]". (Debe tener exactamente 4 campos por línea)
  - Separador: Coma(.). El sistema usa `linea.split(",")` para parsear
  - Ejemplo: A123,Laptop,5,12500.50
2. Entrada de Usuario (Consola) Las Entradas del usuario son dinámicos y se capturan a través de la función `input()`. El sistema aplica validaciones estrictas a cada una.
  - Nombre de Archivo Base: Cadena de texto. (Se usa para construir el nombre del archivo de inventario)
  - Fecha de Búsqueda: Tres cadenas(YYYY.MM.DD). (Se convierten a `int` y a un objeto `datetime.date`. Acepta estar vacío para usar la fecha actual)
  - Código del Producto: Cadena de texto (identificador unico). (`try_codigo()`). No puede ser un espacio vacío)
  - Nombre del Producto: Cadena de texto. (`try_nombre()`). No puede ser un espacio vacío)
  - Cantidad: Número entero ( $\geq 0$ ). (`try_int()`). No puede ser negativo.)
  - Costo: Número decimal ( $> 0$ ). (`try_float()`). No puede ser negativo ni igual a cero.)
  - Opciones del Menu: Un dígito del 1 al 8. (`menu()`)

**Las salidas son los datos que el sistema produce, mostrados al usuario (consola) o escritos en archivos (reportes).**

- **Nombre de Archivo:** "Reporte Final-[día]-[mes]-[año\_actual].txt"
- **Encabezado:** Texto con el total de productos y valor total. (Valor total del inventario: \$155000.00)
- **Tabla de Productos:** Líneas con formato fijo para alineación. ("{: ^25} {: ^25} {: ^20} \$: ^9.2f} \$: ^11.2f}\n")

- **Menus:** Texto simple con opciones numeradas. (menu())
- **Mensajes de Carga:** Animacion de puntos(...) con \r para dar la sencasion de procesamiento. (cargando())
- **Errores/Validaciones:** Mensajes de texto directo que informan por que fallo la entrada o la operacion. (try...except en funcion de captura)

- ## Diagrama de flujo



## PLAN DE TRABAJO

### Semana 1:

- [✓] Planteamiento del problema a realizar. (Todos)
- [✓] Distribución de los roles. (Senior)
- [✓] Realización del GitHub (main) para uso del equipo. (Senior)
- [✓] Realización de los commits dentro del repositorio de GitHub (Todos)

### Semana 2:

- [✓] Realización del pseudo-código para integración completa del proyecto a realizar. (Senior)
- [✓] Actualización a una versión principal del código con funciones completas. (Mids)
- [✓] Verificación del código base en busca de posibles errores y optimización de entradas y salidas. (Juniors)
- [✓] Validación de las ramas creadas para su integración dentro del código principal. (Senior)

### Semana 3:

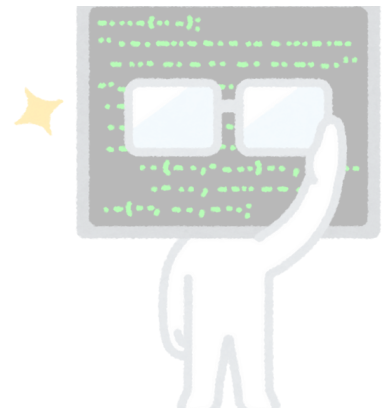
- [✓] Realización de actualizaciones dentro del programa para futuras versiones. (Mids)
- [✓] Ajustar variables y validación de las funciones nuevas del programa. (Juniors)
- [✓] Ajuste de ramas nuevas para validación de las funciones dentro de la rama principal y validar un funcionamiento correcto. (Senior)

### Semana 4:

- [✓] Reajuste de nuevos comandos para parches dentro de las versiones anteriores. (Mids)
- [✓] Simplificar funciones dentro del código para un funcionamiento más optimo. (Junior)
- [✓] Integración para una versión final del código. (Senior)

### Semana 5:

- [✓] Ultimas actualizaciones dentro del programa. (Mids)
- [✓] Revisión a detalle del código completo por posibles errores en el código. (Juniors)
- [✓] Entrega del código en una fase completa lista para su ejecución sin errores. (Senior)



## Evidencia de pruebas (capturas de pantalla de la ejecución en Python).

Inicio del programa y creación de un nuevo inventario.

```
Ingrese el nombre del archivo para el inventario: prueba
Ingrese la fecha del archivo (deje en blanco para usar la fecha actual):)
Año (YYYY):
Mes (MM):
Día (DD):
El archivo: prueba-26-9-2025.txt no existe. Se creará un nuevo inventario para hoy con el nombre: prueba-26-9-2025.txt.
```

Menu.

```
=== Menú ===
1. Agregar producto a inventario.
2. Quitar producto de inventario.
3. Ver los productos registrados.
4. Buscar producto.
5. Calcular total de los productos.
6. Actualizar cantidad o precio de producto.
7. Generar reporte final.
8. Salir.
Seleccione una opcion: █
```

Agregar un nuevo producto.

```
Seleccione una opcion: 1
Accediendo al archivo: prueba-26-9-2025.txt...

===== Registro de producto =====
¿Cual es el codigo del producto?
Codigo a ingresar: MAN-1

¿Cual es el nombre del producto?
Nombre a ingresar: Manzana


¿Cual sera la cantidad?
Cantidad: 25

¿Cual sera el costo individual del producto?
Costo: $9.99
Actualizando inventario...

Producto agregado correctamente.

=== Menú ===
```

Así se ve el archivo de texto



```
prueba-26-9-2025.txt U x
prueba-26-9-2025.txt
1 MAN-1,manzana,25,9.99
2 |
```

Ver inventario.

```
Seleccione una opcion: 3
Cargando inventario...
```

```
=====
Codigo          Producto          Cantidad          Precio          Total
=====
MAN-1           Manzana           25               $ 9.99          $ 249.75
=====

=== Menú ===
```

## Intentar agregar un producto con código duplicado.

```
Seleccione una opcion: 1
Accediendo al archivo: prueba-26-9-2025.txt...
```

```
===== Registro de producto =====
```

```
¿Cual es el codigo del producto?
Codigo a ingresar: MAN-1
```

```
El codigo ya existe en el inventario
Escribe un nuevo codigo
```

No permite el mismo código dos veces  
en el inventario

```
¿Cual es el codigo del producto?
Codigo a ingresar: MAN-2
```

```
¿Cual es el nombre del producto?
Nombre a ingresar: Manzana Roja
```

```
¿Cual sera la cantidad?
Cantidad: 15
```

```
¿Cual sera el costo individual del producto?
Costo: $14.99
Actualizando inventario...
```

```
Producto agregado correctamente.
```

```
=== Menú ===
```

## Actualizar la cantidad de un producto.

```
Seleccione una opcion: 6
Cargando inventario...
```

| Codigo | Producto     | Cantidad | Precio   | Total     |
|--------|--------------|----------|----------|-----------|
| MAN-1  | Manzana      | 25       | \$ 9.99  | \$ 249.75 |
| MAN-2  | Manzana Roja | 15       | \$ 14.99 | \$ 224.85 |

```
Validando opciones...
```

```
¿Que desea actualizar?
1. Actualizar cantidad.
2. Actualizar precio.
Seleccione una opcion (1 o 2): 1
Escribe el codigo del producto a modificar: MAN-2
```

```
Actualizando cantidad del producto: Manzana Roja (Actual: 15)
```

```
¿Cual sera la cantidad?
Cantidad: 25
Actualizando inventario...
```

```
Producto 'Manzana Roja' actualizado exitosamente.
El nuevo cantidad es: 25
```

```
=== Menú ===
```

```
prueba-26-9-2025.txt U x
prueba-26-9-2025.txt
1 MAN-1,manzana,25,9.99
2 MAN-2,manzana roja,25,14.99
3
```

Se actualizó la cantidad

## Eliminar un producto.

```
Seleccione una opcion: 2
Cargando inventario...
```

| Codigo | Producto     | Cantidad | Precio   | Total     |
|--------|--------------|----------|----------|-----------|
| MAN-1  | Manzana      | 25       | \$ 9.99  | \$ 249.75 |
| MAN-2  | Manzana Roja | 25       | \$ 14.99 | \$ 374.75 |

```
Accediendo al archivo: prueba-26-9-2025.txt...
```

```
Escribe el codigo del producto a borrar: MAN-1
Actualizando inventario...
```

```
El producto 'Manzana' con codigo 'MAN-1' ha sido eliminado con exito.
```

```
=== Menú ===
```

```
prueba-26-9-2025.txt U x
prueba-26-9-2025.txt
1 MAN-2,manzana roja,25,14.99
2
```

Se borró correctamente



Calcular total.

```
Seleccione una opcion: 5
Calculando valores...
```

```
=====
El valor total de los productos en el inventario es: $374.75
=====
```

```
=== Menú ===
```

Buscar producto.

```
Seleccione una opcion: 4
Buscando codigos...
```

```
=====
Codigo Disponibles
```

```
=====
MAN-2
```

```
¿Cual es el codigo a buscar?
Codigo: MAN-2
Buscando codigo...
```

```
=====
Codigo      Producto      Cantidad      Precio      Total
-----
MAN-2      Manzana Roja      25      $ 14.99      $ 374.75
=====
```

```
=== Menú ===
```

Generar reporte final.

```
Seleccione una opcion: 7
```

```
=== Generando Reporte Final ===
```

```
Cargando...
```

```
Reporte generado exitosamente en el archivo: Reporte Final-26-9-2025.txt
```

```
=== Menú ===
```

Asi se ve el Reporte Final:

```
≡ Reporte Final-26-9-2025.txt
```

```
1  === INVENTARIO Y REPORTE FINAL ===
```

```
2  Numero total de productos unicos: 1
```

```
3  Valor total del inventario: $374.75
```

```
4
```

```
5  === DETALLES POR PRODUCTO ===
```

```
6  =====
```

```
7  |  |  | Codigo      Nombre      Cantidad      Costo      Total
```

```
8  |  |  | =====
```

```
9  |  |  | MAN-2      Manzana Roja      25      $ 14.99      $ 374.75
```

```
10 |  |  | =====
```

```
11
```

## Conclusiones:

La implementación del sistema de inventario no solo resultó en un programa operativo, sino que también representó una oportunidad para aplicar buenas prácticas de programación y solucionar un problema tangible de registro y gestión de materiales, utilizando los conocimientos adquiridos en el curso. Los principales aprendizajes fueron:

- Almacenar la información en archivos para su persistencia, utilizando un formato estructurado que incluye código, nombre, cantidad y costo.
- Validar las entradas del usuario para evitar campos vacíos o valores numéricos incorrectos, previniendo fallos y garantizando la integridad de los datos.
- Aceptar las reglas del negocio, como la asignación de un código y un nombre único para cada producto, mediante procesos de normalización que eliminan duplicados.
- Ejecutar operaciones completas de gestión (CRUD): añadir, consultar, buscar, modificar y eliminar productos; sumado al cálculo de totales y la generación de reportes.
- Estructurar el código en funciones modulares (como `try_codigo`, `try_nombre`, `try_int`, `try_float`) y administrarlas a través de un menú centralizado y fácil de usar.
- Ofrecer una presentación clara al usuario, con tablas correctamente alineadas y mensajes de error comprensibles.
- Implementar manejo de excepciones para que el programa pueda recuperarse ante fallos y mantener su funcionamiento.
- Fomentar el trabajo en equipo, donde cada integrante, desde su rol, contribuyó en diferentes etapas, mediante revisiones y pruebas iterativas hasta alcanzar el producto final.

**El sistema satisface los criterios de éxito y los requisitos funcionales establecidos: captura datos de forma sencilla, presenta el inventario de manera ordenada, calcula con precisión el costo total y permite realizar actualizaciones sin perder el historial.**

**En resumen, este proyecto no solo significó la creación de un software útil, sino también un aprendizaje integral que abarcó desde la organización del código y el manejo de errores, hasta la valoración del trabajo colaborativo. Asimismo, sienta las bases para futuras mejoras, como la incorporación de interfaces gráficas más amigables o el uso de bases de datos avanzadas, posicionándose como un cimiento sólido para iniciativas de mayor envergadura.**

