

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Curso: Lab LFP
Sección: B+
Ing. David Morales
Aux. Francisco Magdiel Asicono Mateo
Segundo Semestre 2023



Manual Técnico

Andres Alejandro Quezada Cabrera

202200174



Guatemala, 22 de Agosto del 2023

INTRODUCCIÓN

Bienvenido al Manual Técnico del Sistema de Gestión de Inventario. Este documento está diseñado para proporcionar una comprensión detallada de la estructura, funcionalidad y procesos involucrados en nuestro sistema de gestión de inventario. El sistema ha sido desarrollado para optimizar el proceso de seguimiento, control y gestión de inventario de productos en tu organización. A través de una interfaz intuitiva y herramientas eficientes, este sistema facilita la tarea crucial de administrar el flujo de productos y mantener registros precisos.

El sistema cuenta con varias características esenciales para un manejo efectivo del inventario. Algunas de estas características incluyen:

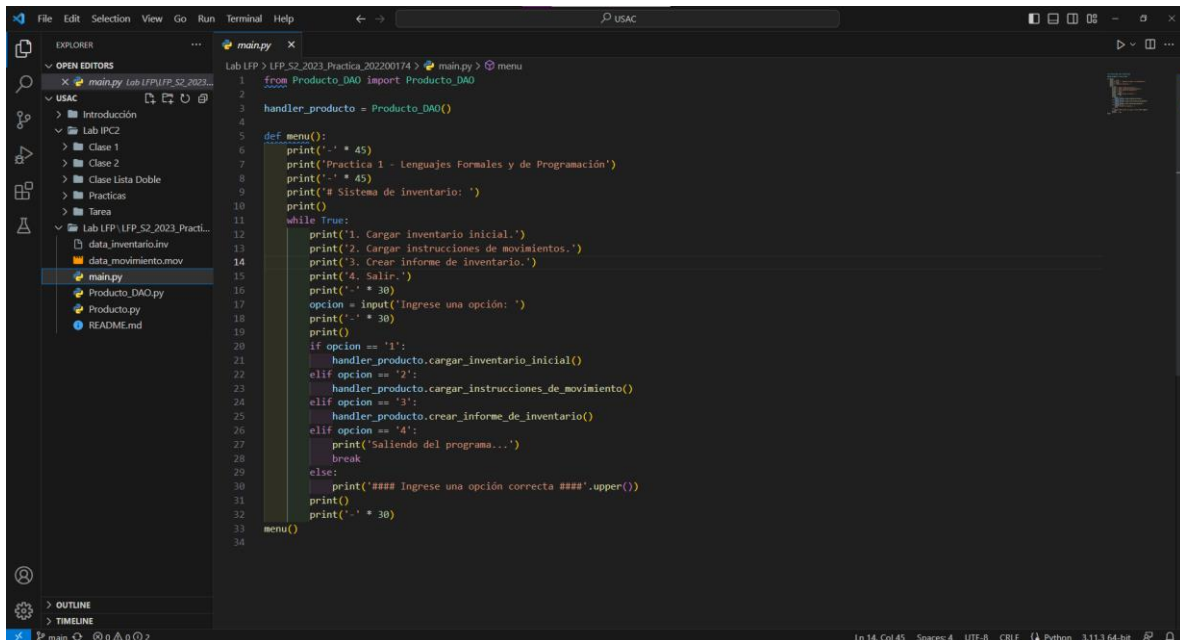
- **Menú Inicial:** Al iniciar el programa, se presenta un menú intuitivo que proporciona acceso rápido a las diversas funciones disponibles en el sistema.
- **Carga de Inventario Inicial:** Una funcionalidad que permite cargar información sobre el inventario inicial de productos. Esto se realiza a través de un archivo en formato .inv, facilitando la carga masiva de datos y evitando errores manuales.
- **Instrucciones de Movimiento:** El sistema permite cargar instrucciones de movimiento que afectan al inventario, como agregar stock o vender productos. Estas instrucciones se pueden cargar a través de un archivo en formato .mov, lo que garantiza un registro preciso y eficiente de todas las transacciones.
- **Generación de Informe de Inventario:** Se proporciona la opción de crear un informe de inventario que captura todos los detalles del inventario actual. Este informe se guarda en un archivo en formato .txt para su posterior referencia y análisis.

INSATACIÓN DEL PROGRAMA Y USO DEL PROGRAMA

Ir al siguiente enlace para obtener el programa

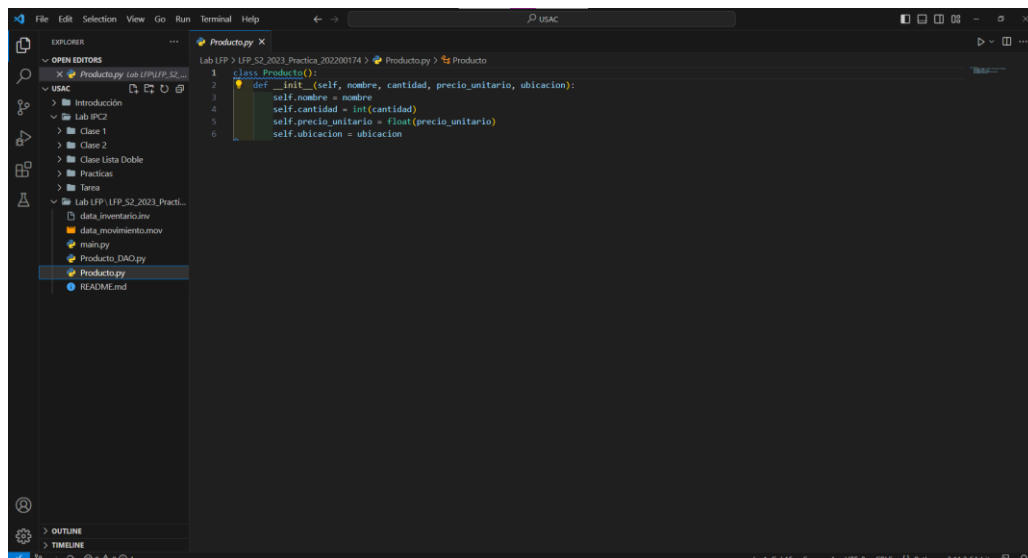
https://github.com/MrQS94/LFP_S2_2023_Practica_202200174

En el archivo main.py, podremos encontrar que existe un menú, en el cual nos servirá para acceder a las funcionalidades del programa.



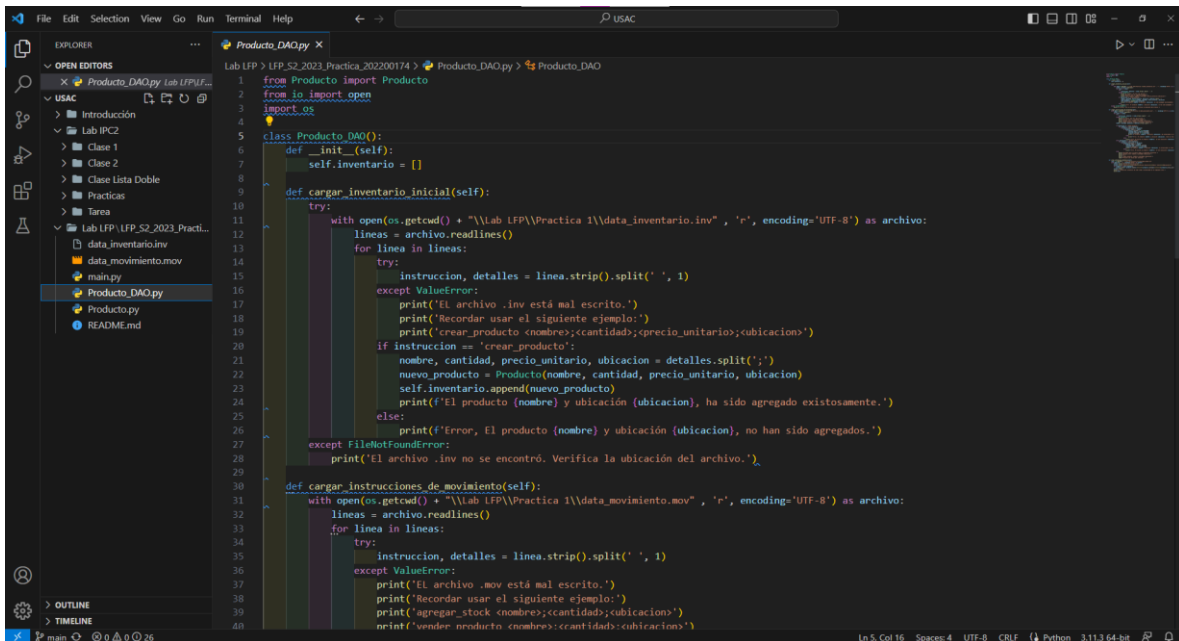
```
1 from Producto.DAO import Producto.DAO
2
3 handler_producto = Producto.DAO()
4
5 def menu():
6     print('\n * 45')
7     print('Practica 1 - Lenguajes Formales y de Programación')
8     print('\n * 45')
9     print('# Sistema de inventario: ')
10    print()
11    while True:
12        print('1. Cargar inventario inicial.')
13        print('2. Cargar instrucciones de movimientos.')
14        print('3. Crear informe de inventario.')
15        print('4. Salir.')
16        print('\n * 30')
17        opcion = input('Ingrese una opción: ')
18        print('\n * 30')
19        print()
20        if opcion == '1':
21            handler_producto.cargar_inventario_inicial()
22        elif opcion == '2':
23            handler_producto.cargar_instrucciones_de_movimiento()
24        elif opcion == '3':
25            handler_producto.crear_informe_de_inventario()
26        elif opcion == '4':
27            print('Saliendo del programa...')
28            break
29        else:
30            print('### Ingrese una opción correcta ###'.upper())
31        print()
32    print('\n * 30')
33 menu()
34
```

En el archivo Producto.py vamos a encontrar el modelo que vamos a utilizar para guardar los productos, en este caso, sería el nombre, cantidad, precio_unitario, ubicación, este último será uno de los más importantes del programa.



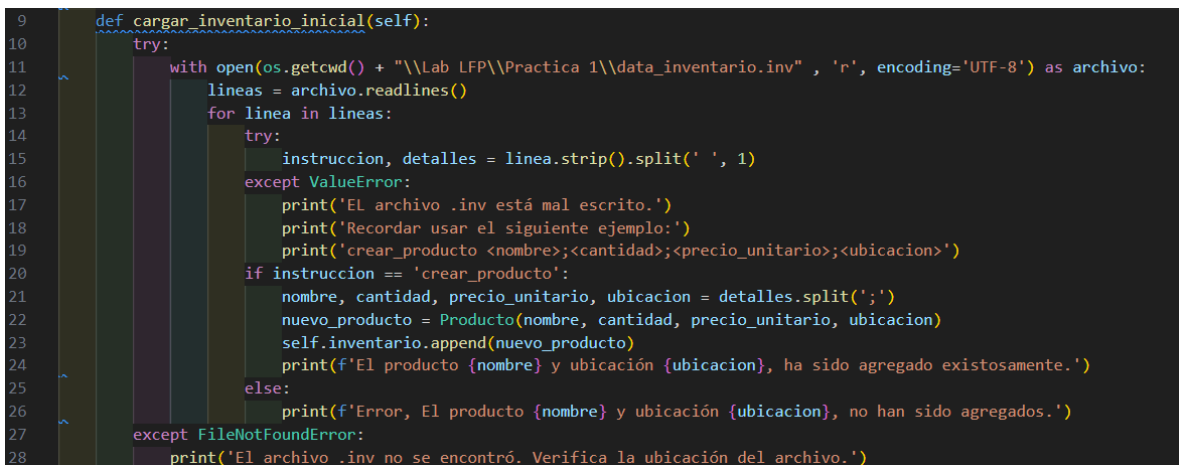
```
1 class Producto():
2     def __init__(self, nombre, cantidad, precio_unitario, ubicacion):
3         self.nombre = nombre
4         self.cantidad = int(cantidad)
5         self.precio_unitario = float(precio_unitario)
6         self.ubicacion = ubicacion
```

En el archivo `Producto_DAO.py` será nuestro “cerebro” el cual será utilizado para recolectar, actualizar e imprimir la información.



```
1 from Producto import Producto
2 from io import open
3 import os
4
5 class Producto_DAO():
6     def __init__(self):
7         self.inventario = []
8
9     def cargar_inventario_inicial(self):
10        try:
11            with open(os.getcwd() + "\\Lab LFP\\Practica 1\\data_inventario.inv", 'r', encoding='UTF-8') as archivo:
12                lineas = archivo.readlines()
13                for linea in lineas:
14                    try:
15                        instruccion, detalles = linea.strip().split(' ', 1)
16                    except ValueError:
17                        print('El archivo .inv está mal escrito.')
18                        print('Recordar usar el siguiente ejemplo:')
19                        print('crear_producto <nombre>;<cantidad>;<precio_unitario>;<ubicacion>')
20                    if instruccion == 'crear_producto':
21                        nombre, cantidad, precio_unitario, ubicacion = detalles.split(';')
22                        nuevo_producto = Producto(nombre, cantidad, precio_unitario, ubicacion)
23                        self.inventario.append(nuevo_producto)
24                        print(f'El producto {nombre} y ubicación {ubicacion}, ha sido agregado existosamente.')
25                    else:
26                        print(f'Error, El producto {nombre} y ubicación {ubicacion}, no han sido agregados.')
27        except FileNotFoundError:
28            print('El archivo .inv no se encontró. Verifica la ubicación del archivo.')
29
30    def cargar_instrucciones_de_movimiento(self):
31        with open(os.getcwd() + "\\Lab LFP\\Practica 1\\data_movimiento.mov", 'r', encoding='UTF-8') as archivo:
32            lineas = archivo.readlines()
33            for linea in lineas:
34                try:
35                    instruccion, detalles = linea.strip().split(' ', 1)
36                except ValueError:
37                    print('El archivo .mov está mal escrito.')
38                    print('Recordar usar el siguiente ejemplo:')
39                    print('agregar_stock <nombre>;<cantidad>;<ubicacion>')
40                    print('vender_producto <nombre>;<cantidad>;<ubicacion>')
```

En el archivo `Producto_DAO.py`, tenemos la función “`cargar_inventario_inicial`”, el cual se encarga por medio de `with open(ruta)` y `readLines()` de conocer donde se encuentra el archivo y leer lo que tiene el archivo, en este caso tenemos un `.inv`. En el momento que ya sido leído, la variable `instrucción` se encarga de conocer lo que vamos a realizar, en este caso será `crear_producto`, y la función `detalles` se encarga de conocer, el nombre, cantidad, `precio_unitario` y `ubicación`, muy importante porque con ello podremos saber cuales productos vamos a añadir. Luego de eso, lo insertamos dentro de un array llamado `inventario`, que anteriormente creamos y ahí es donde se guarda toda nuestra información.



```
9 def cargar_inventario_inicial(self):
10     try:
11         with open(os.getcwd() + "\\Lab LFP\\Practica 1\\data_inventario.inv", 'r', encoding='UTF-8') as archivo:
12             lineas = archivo.readlines()
13             for linea in lineas:
14                 try:
15                     instruccion, detalles = linea.strip().split(' ', 1)
16                 except ValueError:
17                     print('El archivo .inv está mal escrito.')
18                     print('Recordar usar el siguiente ejemplo:')
19                     print('crear_producto <nombre>;<cantidad>;<precio_unitario>;<ubicacion>')
20                 if instruccion == 'crear_producto':
21                     nombre, cantidad, precio_unitario, ubicacion = detalles.split(';')
22                     nuevo_producto = Producto(nombre, cantidad, precio_unitario, ubicacion)
23                     self.inventario.append(nuevo_producto)
24                     print(f'El producto {nombre} y ubicación {ubicacion}, ha sido agregado existosamente.')
25                 else:
26                     print(f'Error, El producto {nombre} y ubicación {ubicacion}, no han sido agregados.')
27     except FileNotFoundError:
28         print('El archivo .inv no se encontró. Verifica la ubicación del archivo.')
```

En el archivo Producto_DAO.py, tenemos la función “cargar_instrucciones_de_movimiento”, el cuál es el encargado de agregar o vender stock de un producto, realiza una vez más la lectura del archivo, y en este caso tenemos 2 instrucciones “vender_producto” o ”agregar_stock”, entonces utilizamos un if anidado para conocer cual instrucción es la que vamos a utilizar, dentro de cada if existen otros if que se encargan de conocer la ubicación y el nombre del producto, y si el primero no se encuentra, significa que no existe ese producto en esa ubicación, en el caso de la instrucción “vender_producto”, tenemos que tener en cuenta que la cantidad a vender no puede ser mayor al stock del producto actual, permite conocer si lo anterior es negativo.

```

30 def cargar_instrucciones_de_movimiento(self):
31     with open(os.getcwd() + "\\Lab LFP\\Practica 1\\data_movimiento.mov", 'r', encoding='UTF-8') as archivo:
32         lineas = archivo.readlines()
33         for linea in lineas:
34             try:
35                 instruccion, detalles = linea.strip().split(' ', 1)
36             except ValueError:
37                 print('El archivo .mov está mal escrito.')
38                 print('Recordar usar el siguiente ejemplo:')
39                 print('agregar_stock <nombre>;<cantidad>;<ubicacion>')
40                 print('vender_producto <nombre>;<cantidad>;<ubicacion>')
41                 nombre, cantidad, ubicacion = detalles.strip().split(';')
42
43             if instruccion == 'vender_producto':
44                 for producto in self.inventario:
45                     if producto.ubicacion == ubicacion:
46                         if producto.nombre == nombre:
47                             if producto.cantidad >= int(cantidad):
48                                 cantidad_temp = producto.cantidad
49                                 producto.cantidad -= int(cantidad)
50                                 print(f'El producto {nombre} y ubicación {ubicacion}, ha actualizado su stock de {cantidad_temp} a {producto.cantidad}.')
51                             else:
52                                 print(f'Error, El producto {nombre} y ubicación {ubicacion}, tiene un stock menor al deseado.')
53                         else:
54                             print(f'Error, No existe un producto ({nombre}), en esta ubicación ({ubicacion})')
55             elif instruccion == 'agregar_stock':
56                 for producto in self.inventario:
57                     if producto.ubicacion == ubicacion:
58                         if producto.nombre == nombre:
59                             cantidad_temp = producto.cantidad
60                             producto.cantidad += int(cantidad)
61                             print(f'El producto {nombre} y ubicación {ubicacion}, ha actualizado su stock de {cantidad_temp} a {producto.cantidad}.')
62                         else:
63                             print(f'Error, No existe un producto ({nombre}), en esta ubicación ({ubicacion})')
64             else:
65                 print('Verifique que su archivo tenga la siguiente estructura.')
66                 print('agregar_stock <nombre>;<cantidad>;<ubicacion>')
67                 print('ó')
68                 print('vender_producto <nombre>;<cantidad>;<ubicacion>')
69                 input('Presione una tecla para continuar...')

```

Por último, tenemos la función “crear_informe_de_inventario”, esto se encarga reconocer que exista el archivo “resultado_123123.txt”, sino crea el archivo, recorre todo el inventario buscando cada nombre, cantidad, precio_unitario, total y ubicación del producto.

```

1 def crear_informe_de_inventario(self):
2     ruta = os.getcwd() + "\\Lab LFP\\Practica 1\\resultado_123123.txt"
3     with open(ruta, 'a', encoding='UTF-8') as archivo:
4         archivo.write("Informe de Inventario:\n")
5         archivo.write(
6             "Producto\t\t\tCantidad\t\t\tPrecio Unitario\t\t\tValor Total\t\t\tUbicación\n")
7         for producto in self.inventario:
8             archivo.write(
9                 f'{producto.nombre}\t\t\t{producto.cantidad}\t\t\t{producto.precio_unitario}\t\t\t{int(producto.cantidad) * float(producto.precio_unitario)}\t\t\t{producto.ubicacion}\n')
10        archivo.close()
11        print(
12            'El informe del inventario ha sido creado o actualizado en la siguiente ruta:')
13        print(ruta)

```