**Gestor y organizador de Ventas y Compras de productos**

**Presentación**

Proyecto para facilitar la gestión de un mercado o una empresa que maneje pocos datos que persistan en el tiempo. La aplicación permite manipular, crear y filtrar datos, visualizar un grafico con los gastos de las compras y descargar el mismo, y exportar los datos a archivos csv o para excel.

**Funciones**

**Grafico**

Un grafico de flecha integrado que muestra los gastos a travéz de las ventas. Integrado con un algoritmo que permite que se actualize cada vez que se cambia alguno de los datos que tengan que ver con dinero. Sus funciones son **graph y update\_graph.** Puede descargarse una imagen que contenga el grafico actual de la aplicación, la imagén estará hubicada en la carpeta **Descargas** de su usuario.

**Logs**

Tiene la funcion de logs, así que despues de correr la app y mover un par de cosas, puede revisar el archivo logs.log en la raíz del proyecto. Se usa el modulo loggin de python y una función propia llamada **logs**

**Filtro**

Puede que sean cargados muchos datos y sea un poco molesto visualizarlos o buscar alguno en especifico, así que sobre el arbol de proyectos hay un input donde por defecto se puede buscar por ID del producto, si necesita buscar por nombre, fecha o costo tiene un menu de opciones a la derecha del input para cambiar de tipo de filtración. Para que funcionen se consutruyeron las funciones **validate\_is\_numeric, validate\_is\_text, validate\_input, filter\_treeview y update\_treeview.**

**Exportaciones**

Debajo del arbol de productos y al lado del botón de vender **"sell"**, se encuentran los dos botones con diferentes opciones de exportación: Exportación a csv y exportación a excel. Cuando presione en cualquiera de los dos, el archivo se descargará en su carpeta **Descargas** de su usuario. Para dar esta funcionalidad se usó la librería pandas dentro de las funciones **export\_to\_excel y export\_to\_csv.**

**Validaciones**

La app viene con validaciones para ciertas acciones:

* Al momento de registrar un producto, se valida el contenido de los campos con regex para revisar si son validos. Puede ver los patrones dentro de las funciones **validate\_name y validate\_cost.**
* Cuando se estén filtrando los productos, el menú y el input tendrán validaciones para saber exactamente a que tipo de filtro está estableciendo, validando también si el valor es numérico o texto. Puede ver su funcionamiento dentro de las funciones **validate\_is\_numeric**, **validate\_is\_text, validate\_input** y **filter\_treeview**.

**Manejo de datos**

Para manipular los datos que quiera, tiene diferentes funciones:

* Comprar un producto: De acá salen las validaciones anteriormente nombradas, la fecha de compra se almacena sola mediante el uso de datetime y el ID es autoincremental, así que no hace falta especificarlo en el formulario. Dentro de la app solo bastará con colocar el "Nombre" del producto y el "Costo" del mismo. El funcionamiento puede encontrarlo en las funciones **create\_element** y **buy.**
* Vender un producto: Se puede vender un producto haciendo click sobre el que quiera y presionando el boton "sell" que está por debajo del arbol de productos. La idea es que se guarden los datos de la ventas además de los de compra, pero por el momento se está usando una sola tabla. El funcionamiento se encuentra en la función **delete\_element.**
* Edición de productos: Edite cada producto haciendo doble click sobre el mismo, le apareceran unos campós pequeños con la información del nombre y del costo del producto, puede retirar la acción o terminarla haciendo click sobre cualquiera de los campos y presionando la tecla "enter". Sus funciones son **edit\_element y save\_edit.**

**Empezar con el proyecto**

Para correr la aplicación por primera vez, necesita crear su entorno virtual, ya sea con anaconda, virutualenvwrapper o los entornos virtuales de python.

> virtualenvwrapper:

***mkvirtualenv mi\_env***

> *Anaconda:*

para anaconda puede usar anaconda navigator o la terminal si lo tiene en las variables de entorno, con la segunda opción sería:

***conda create --name mi\_env python=3.11.10***

para activarlo:

***conda activate mi\_env***

> entornos virtuales de python (venv):

***python -m venv mi\_env***

para activarloen windows:  
***mi\_env/Source/activate***

para activarlo en linux o Mac:

***source mi\_env/bin/activate***

Seguido de haber creado y activado su entorno, debe correr el comando *`pip install -r requirements.txt*` haciendo referencia al archivo que contiene las dependencias del proyecto. Después de eso puede correr main.py y la app va a funcionar perfecto!