1. Tema: Retail - Análisis del Flujo de Efectivo de tiendas para el pago a proveedores
2. Problemática Para la gran mayoría de tiendas, el flujo de caja de chica diaria resulta de gran importancia para el sostenimiento de sus negocios. Estas tiendas al subsistir de transacciones minoristas, consecuentemente, no cuentan con una base fija de ingresos diariamente. Esto posiciona a las mismas en una situación económicamente vulnerable ante las volatilidades que puedan existir en el ejercicio de la compra-venta. Si los tenderos no disponen de la liquidez suficiente al momento que los proveedores los visitan, estos no podrían continuar con el abastecimiento de nuevas mercaderías para el negocio.
3. Solución Desarrollar un sistema en Python que permita registrar las entradas a la caja chica (efectivo, transferencias, qr), calcular automáticamente el saldo disponible y generar reportes que faciliten la previsión de pagos a proveedores. De esta manera, los tenderos podrán anticipar faltantes de efectivo, tomar decisiones financieras más informadas y asegurar la continuidad del abastecimiento de mercaderías para su negocio.
4. Fuentes de Datos Cuatro bases de datos en formato CSV de la tienda aurelion: Tabla de Clientes: ruta(Base\_de\_datos/clientes.csv) Tabla de Detalles de Venta: ruta(Base\_de\_datos/detalle\_ventas.csv) Tabla de Productos: ruta(Base\_de\_datos/productos.csv) Tabla de Ventas: ruta(Base\_de\_datos/ventas.csv)
5. Estructura Base de Datos **Definicion** Cada base de datos contiene registros organizados en filas y columnas: Cliente: Cada registro representa un cliente, con su ID, nombre, correo electrónico, ciudad y fecha de alta. Detalle\_ventas: Cada registro representa un producto vendido en una venta específica, con cantidad, precio unitario e importe total. Productos: Cada registro corresponde a un producto disponible, indicando su ID, nombre, categoría y precio unitario. Venta: Cada registro representa una venta realizada, incluyendo fecha, nombre y correo del cliente, y el medio de pago utilizado. **Relación** Los datos permiten relacionar clientes, ventas y productos para análisis de transacciones y comportamiento de ventas. **Estructura, tipos y escala** Tabla: cliente id\_cliente: N(3) Nombre\_cliente: AN(50) mail: AN(30) ciudad: AN(30) fecha\_alta = registro año: N(4) mes: N(2) dia: N(2) Fin registro Fin tabla

Tabla: Detalle\_ventas id\_venta: N(3) id\_producto: N(2) nombre de producto: AN(30) cantidad: N(1) precio\_unitario: N(4) importe: N(5) Fin tabla

Tabla: Productos id\_producto: N(3) nombre\_producto: AN(30) categoria: AN(20) precio\_unitario: N(4) Fin tabla

Tabla: Ventas id\_venta: N(3) fecha = registro año: N(4) mes: N(2) dia: N(2) fin registro nombre\_cliente: AN(50) mail: AN(30) medio\_pago: AN(“tarjeta”, “qr”, “transferencia”) Fin registro

1. Información Programa y Pasos **Información** El programa trabaja con tres archivos principales, cada uno con una estructura de registros diferente: 1- Archivo de Ventas (arch\_ventas): Este archivo se usa para filtrar las ventas por fecha o medio de pago. 2- Archivo de Detalle de Ventas (arch\_detalle): Este archivo permite calcular los importes totales y determinar cuáles productos se vendieron más. 3- Archivo de Productos (arch\_productos): Este se usa para inicializar el arreglo de productos, que luego acumula la cantidad e importe vendido de cada uno. **Estructura de trabajo interna** El programa crea un arreglo llamado productos (de tamaño N) que guarda temporalmente: nombre del producto, cantidad vendida acumulada, importe total acumulado Este arreglo se actualiza a medida que se leen los registros y al final se ordena de mayor a menor según el importe, para mostrar el ranking de los más vendidos. **Pasos** *Ambiente* 1- Se realiza la inicialización de Archivos y Registros 2- Se definen los registros Ventas, Detalle\_ventas, Productos y Producto\_acum 3- Se crean los archivos correspondientes (arch\_ventas, arch\_detalle, arch\_productos) y sus registros de lectura.4- Se define un arreglo productos para acumular la cantidad y el importe de cada producto. 5- Se inicializan variables de totales (tot\_diario, tot\_mensual, tot\_total) y de medios de pago (tot\_tarjeta, tot\_efectivo, tot\_qr, tot\_transferencia). *Apertura de Archivos* 1- Se abren los archivos de ventas, detalle de ventas y productos. 2- Se leen los primeros registros de arch\_ventas y arch\_detalle. 3- Ingreso del Período de Resumen 4- El usuario ingresa el plazo de resumen que desea conocer: diarias, mensuales o total. 5- Se inicializan los acumuladores de totales y se asignan valores iniciales a las variables de mes, día y pago *Inicialización del Arreglo de Productos* 1- Se recorre el archivo arch\_productos y se cargan los nombres de productos en el arreglo productos. 2- Se inicializan cantidad e importe en 0 para cada producto. *Cálculo de Ventas por Período* Ventas Diarias: 1- Se solicita al usuario el mes y día deseados. 2- Se recorre el archivo de ventas, filtrando por la fecha ingresada. 3- Para cada venta, se recorre el archivo de detalle y se acumulan importes: 4- Total diario (tot\_diario) y total por medio de pago. 5- Se actualiza el arreglo de productos con cantidad e importe. Ventas Mensuales: 1- Se solicita al usuario el mes deseado. 2- Se recorre el archivo de ventas, filtrando por el mes ingresado. 3- Para cada venta, se acumulan importes de manera similar a la opción diaria. Ventas Totales: 1- Se recorren todos los registros de ventas y detalle de ventas. 2- Se acumulan los importes totales y por medio de pago. 3- Se actualiza el arreglo de productos con cantidad e importe. *Ordenamiento de Productos* 1- Se ordena el arreglo productos de manera descendente según el importe total de cada producto. 2- Esto permite identificar rápidamente los productos más vendidos en términos de ingresos. *Salida de Resultados* 1- Se muestra un resumen del total de ventas según el período seleccionado. 2- Se imprime el total por cada medio de pago: tarjeta, efectivo, QR y transferencia. 3- Se muestra la lista de productos más vendidos, indicando: Nombre del producto. Cantidad vendida. Importe total generado. *Cierre de Archivos* 1- Se cierran todos los archivos (arch\_ventas, arch\_detalle y arch\_productos).
2. Pseudocódigo Pseudocódigo: ruta(Pseudocódigo/pseudocódigo.md)
3. Diagrama Diagrama de flujo: ruta(Diagrama/Diagrama.png)
4. Sugerencias de Copilot

* Se inicia el proyecto preguntandole a Copilot como conectar Git con Visual Studio con el fin de desarrollar un repositorio virtual en el que todos los participantes pudieran poner cambios en tiempo real en el código, a lo cual nos dio paso por paso como realizar la conexión, finalmente esta idea fue descartada en esta actividad pero será retomada por el grupo mas adelante
* Con una idea clara del proyecto que queríamos desarrollar se le pregunta a Copilot como estructurar la documentación en la cual nos basamos para el desarrollo del proyecto, dicha documentación son los 10 ítems desarrollados en el código
* Teníamos pensado que el código arrojará la información referente a: según el dinero recaudado por una tienda si o no alcanzaba para pagar a los proveedores, pero como no contábamos con esos valores en la base de datos, la idea fue descartada
* Le preguntamos como conectar la Documentacion.md al Programa.py, es decir, como conectar el programa main a diferentes secciones de la documentación, paso que seria mas adelante importante al momento de conectar el codigo como las bases de datos
* Sumando a lo anterior se le pregunta cómo acceder a los diferentes ítems en formato lista(uno por uno), lo cual agregado al código
* Conforme íbamos avanzando se le fueron preguntando los errores que iba arrojando el programa desarrollado, por ejemplo nos indicaba que Python no puede encontrar el archivo Documentacion.md en la misma carpeta donde se ejecuta el programa. y nos daba la ruta correcta en el código
* Una vez terminado el código preguntamos por errores de sintaxis
* Se le pregunta si la base de datos se puede recorrer de inicio a fin como un archivo, esto con el fin de entender cómo trabajar con un archivo csv en python mientras desarrollamos el pseudocódigo, nos indico como poner la ruta de enlace con el tipo de archivo, tanto en pseudocódigo como en el diagrama desarrollado
* Al momento de desarrollar el programa, como ya hemos dicho anteriormente, seria desarrollado por ítems o secciones, en un punto del desarrollo, para seleccionar un ítem, nos enviaba a todos y no discriminaba, por lo tanto le mandamos el código para que nos ayudará en el proceso de selección del ítem
* Al momento de desarrollar los totales: por día, mensuales y el total general, le pedimos ayuda con la sintaxis
* Le ingresamos el pseudocódigo con el fin de detectar errores en la sintaxis, pero nos nos realizó cambios
* A la hora de hacer el resultado de los productos más vendidos de manera descendente (los que tuvieran más importe) por diario, mensual y total; nos sugirió usar la estructura de arreglos
* Se le preguntó qué tipo de ordenamiento seria el mejor para organizar el arreglo, y nos recomendó el arreglo de burbuja: Para i := 1 hasta N-1 hacer Para j := 1 hasta N-i hacer Si productos[j].cantidad\_vendida < productos[j+1].cantidad\_vendida entonces aux := productos[j] productos[j] := productos[j+1] productos[j+1] := aux Fin si Fin para Fin para
* Se le preguntó por la estructura de los bucles a lo cual nos reescribió el código, cambiando sintaxis, lo cual no fue tomado, ya que confiábamos en nuestra estructura
* Le preguntamos si debíamos definir el tamaño del arreglo, nos dice que sí, ya que el arreglo es una estructura de tamaño
* Se le preguntó cómo integrar la imagen del diagrama de flujo el pseudocódigo, y el nos establece la ruta
* Le pedimos que nos ayudará con los comentarios