**TRABAJO PRACTICO:**

-mATERIA:

* Laboratorio de Programacion III

-Profesor:

* Gonzalo Benoffi.

-Integrantes:

* Corasaniti, Marcos.
* Lambrech, Damian.
* Ramirez, Valentina.

**Repositorio: https://github.com/marcosCora/TP-Final-Laboratorio-3**

# ○Introduccion.

Nosotros hemos decidido realizar como trabajo final una aplicación del tipo E-Commerce. El programa está desarrollado en base a nuestro objetivo principal, que ofrezca la venta de tres tipos de productos electrónicos: celulares, televisores y computadoras. Por esta misma razón, el menú se encuentra enfocado en visualizar y comprar estos artículos.

El sistema cuenta con dos tipos de ingreso: iniciar sesión como cliente o como administrador. Según el ingreso, el programa les permite realizar determinadas acciones:

-El administrador puede manejar el catálogo, visualizar los productos que tiene a la venta y modificarlos si así lo desee. También puede acceder a los productos que fueron vendidos con anterioridad.

-El cliente puede visualizar el catálogo, con sus filtros de búsqueda, y comprar productos. También cuenta con un carrito, donde el usuario agrega los productos desea comprar y eliminar los que ya no.

○ Informe Técnico de la aplicación

La toma de decisiones sobre el diseño de clases y el funcionamiento del sistema se baso en el objetivo que el programa iba a tener: vender. Nuestra primera idea fue vender únicamente productos informáticos, pero cuando comenzó el diseño de clases nos dimos cuenta de que debía tener la posibilidad de vender otros.

# ►Paquete ModelsProducto

Por la razon mencionada anteriormente, en el diseño de clases se decidió crear una clase **Producto** y otra **Producto Informático** (que hereda de Producto), para que se le puedan agregar otros artículos sin necesidad de modificar los productos cargados. De Producto Informatico heredan **Computadora**, **Celular** y **Televisor**.

En **Producto** se encuentra los atributos que compartirian cualquier articulo que se encuentre a la venta, como nombre, marca, precio; y en **Producto Informatico**, los que tendrian en comun estos tipos de productos, como el sistema operativo, el almacenamiento, los accesorios, procesador, el tamaño de la pantalla.

Ademas,  **Producto** cuenta con otros dos atributos para manejar la entrada y salida de mercaderia: stock, que maneja la cantidad de articulos que se encuentran de este tipo en el inventario, y vendidos, que maneja la cantidad de articulos que se vendieron.

# ►Paquete ModelsUsuario

Para crear a los clientes y el administrador, se diseño la clase **Persona**, que cuenta con los atributos nombre, apellido y código postal. De esta clase hereda **Usuario**, que cuenta con los datos que van a identificar al cliente.

El menú que puede utilizar Usuario cuenta con las siguientes funciones:

* Para que el registro de un usuario sea exitoso, el nombre de usuario tiene que ser único.
* Para que el inicio de sesión sea exitoso, el usuario tiene que ingresar su nombre de usuario, que en caso de no existir lanzara una excepción, y luego de verificarlo, se le permite ingresar la contraseña.
* Menú del Carrito, que permite visualizarlo, eliminar y comprar productos.
* Menú del Catálogo, que permite visualizar todos los productos o por tipo de producto.

# ►Paquete ModelsFactura

Para el registro de productos comprados, se diseño la clase **Factura.** Esta clase tiene un arrayList de productos que se van a comprar y los datos del usuario que los compro. El usuario recibe esta factura, con el monto total que va a tener que pagar, y se guardan en un archivo que será visible solo para el admin.

# ►Paquete ModelsGestoras

Creamos tres clases que envuelven las descriptas anteriormente:

-**Gestora de Productos**, que maneja un HashMap de productos, con la clave siento el tipo de producto a guardan (por ejemplo, “celular”) y el valor un arrayList de productos. En esta clase están las funciones que manejan el archivo con productos. También cuenta con métodos que se utilizan a la hora de modificar los productos, como incrementar stock, cambiar precio y eliminar producto.

**-Gestora de Usuarios**, que maneja un HashSet de usuarios, que no permite que el nombre de usuario se repita. En esta clase están las funciones que manejan el archivo con usuarios.

-**Gestora de Factura**, que maneja un TreeSet de facturas, que están ordenadas por fecha de compra. En esta clase están las funciones que manejan el archivo con facturas.

# ►Paquete ModelsExcepcion

Para las excepciones, creamos una para cada situación que requiera:

-**Mi Excepción Usuario Repetido**, que se lanza en caso de que, a la hora de registrarse, el usuario ingresado ya exista.

- **Mi Excepción Nombre de Usuario**, que se lanza al momento de que el nombre de usuario sea incorrecto al momento de ingresar.

- **Mi Excepción Contrasenia Incorrecta**, que se lanza al momento de ingresar la contraseña del usuario y el código de admin, en caso de que sea incorrecta.

- **Mi Excepción Stock insuficiente**, que se lanza al momento de comprar un producto, por si no hay suficiente stock de ese producto para realizar la compra.

# ► Paquete ModelsEnum

-**Tipo Usuario**: cliente, administrador. Definen el menú que podrán ver a la hora de verificar que tipo de usuario son.

-**Tipo Producto**: computadora, celular, televisor. Define el tipo de producto que se va a guardar en cada arreglo del mapa de productos.

-**Tipo PC**: notebook, netbook, pc de escritorio. Características del producto computadora.

-**Tipo Pantalla**: led, lcd, oled, amoled. Características de las pantallas del producto televisor.

-**Tipo Opción**: son el tipo de opciones del menú,



○ Manual del Usuario:

Usuarios para navegar:

**usuario admin:**

nombre de usuario: admin123

contraseña: admin123

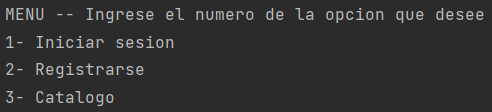
codigo admin: holajava

**usuario normal:**

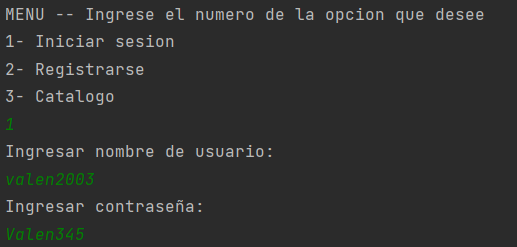
nombre de usuario: gonza

contraseña: gonza123

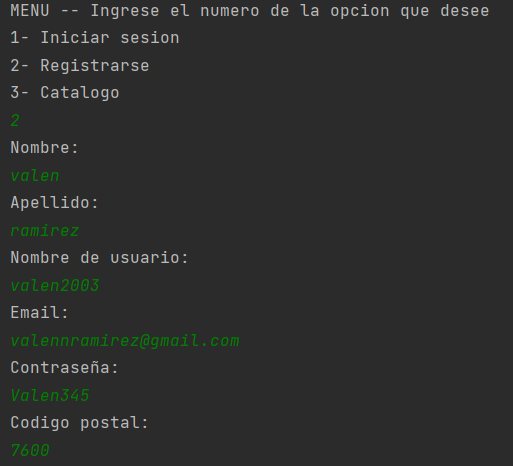
**Si tiene problema con los archivos, dejamos en un bloc de notas dentro de la carpeta para que copie y pegue en el main lo que indicamos. De esta manera se cargaran los usuarios y producctos en los archivos**.



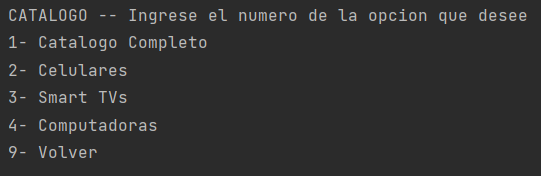
-Opcion 1:



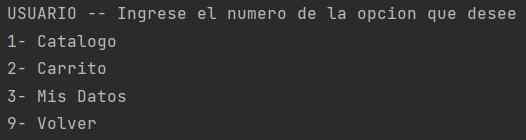
-Opcion 2:



-Opcion 3:



Una vez el usuario ya ingresado,



○ Diario de trabajo.

**Semana 1:**

Ideamos un sistema de concecionaria, que luego modificamos a un sistema de ventas en general para poder abarcar los temas vistos en clase, comenzamos diagramando a grandes rasgos las Clases base como Producto o Usuario, pero a medida que requeriamos mas abstraccion se nos dificulto llevar al papel las especificaciones necesarias (Por ejemplo el funcionamiento de algunas clases genericas y las colecciones a usar).

**Semana 2:**

En la segunda semana diseñamos el diagrama de clases mas detallado con funcionalidades mas especificas. Decidimos separar ProductoInformatico de Producto para dar la posibilidad de (en un futuro) agregar distintos tipos de productos, creamos las clases gestoras y, a medida que ibamos desarrollando los metodos, creamos las excepciones necesarias. Decidimos usar el HashSet para manejar los usuarios, ya que no permite duplicados. Usamos el HashMap <TipoProducto,Arraylist<Producto>>, compuesto de una clave TipoProducto ( CELULAR-PC-TV) y un valor de ArrayList <Producto>, el cual nos permite acceder inmediatamente a la lista de productos deseada. Utilizamos ArrayList para varias colecciones de las clases, como el Carrito o las Facturas. Los enum se fueron dando tambien a medida que se desarrollo el programa.

Para llevar a cabo el proyecto, intentamos dividir las tareas y trabajar a travez de git., actualizando el proyecto a medida que lo requeriamos. Nos coordinamos a travez de WhatsApp y haciendo llamadas diarias por Discord.

**Semana 3:**

Para la tercer semana se decidio agregar una clase Factura, que guarde la fecha, el producto y el nombre del usuario que compro ese determinado articulo, diseñamos su propia gestora, que se encarga de almacenar el listado de facturas en un archivo local JSON al cual puede acceder el administrador. Tambien se desarrollo el menu y el sistema, que compone la interfaz grafica.

Ademas, se decidio utilizar la genericidad para la clase controladora de archivos. Esta clase implementa una interfaz .

○ Fuentes de información consultadas.

* Chat gpt
* StackOverflow
* Baeldung