Brief technical challenge Mercado libre

# abstracción de los elementos

Despues de analizar la UI de referncia, se empezara a modelar la solucion bottom up desde la persistencia (modelo de datos) hasta lo que se mostrara como interfaz grafica, para el reto tecnico se tendra en consideracion tres entidades a nivel de datos

1. Vendedor
2. Producto
3. Publicación

## Vendedor

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Texto

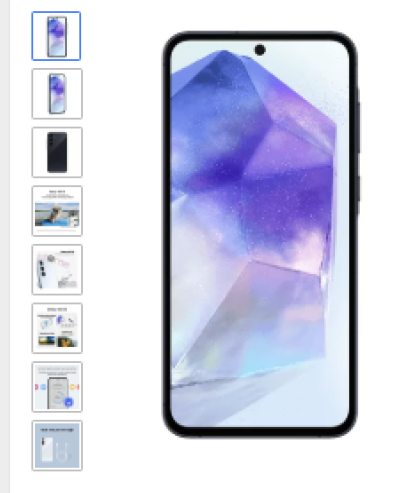
El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

* El vendedor tiene información básica como
  + Nombre
  + Numero de ventas
  + Calificación de atención
  + Calificación de tiempo de entrega
* Esta relacionado con los productos que ofrece
* Esta relacionado a sus publicaciones

## Producto

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

* El producto tiene información básica como
  + Nombre
  + Imagen
  + Arreglo de características
  + Numero de disponibles
  + Precio
  + Descuento
  + Atributo de selección de modelo (color)
* El producto está relacionado consigo mismo bajo un atributo que diferencia el modelo
* El producto esta relacionado con su vendedor
* El producto esta relacionado con una publicación

## Publicación

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

* Cobro por envio
* Detalles post venta:
  + Cobro por devolución
  + Protección de compra
  + Tiempo de garantia
* métodos para compra:
  + Tarjetas de crédito
  + Tarjetas debito
  + Efectivo
* Productos del mismo vendedor (relacionados)
* Productos relacionados (de otros vendedores)
* Descripcion de la publicación
* Esta relacionado con el producto que vende
* Esta relacionado con el vendedor de la publicación

# Modelo de datos

Diagrama

El contenido generado por IA puede ser incorrecto.

* Los vendedores pueden tener múltiples productos, por lo cual productos tiene un FK que identifica su vendedor
* Los Vendedores pueden tener múltiples publicaciones por lo cual la publicación tiene una FK que corresponde al vendedor
* Los productos están relacionados consigo mismos (relación reflexiva) ya que pueden existir varios modelos del mismo productos este esta asociado con un parent id si este parent id no existe significa que el producto es padre
* Los productos están relacionados a una publicación por lo cual la publicación tiene una FK que lo relaciona al producto padre

# Retos durante el desarrollo

La restricción de usar archivos csv y json para persistir aumento un poco la dificultad durante el desarrollo, para el caso del proyecto use archivos csv por su similtud a una tabla SQL

* Se uso opencsv como librería para leer los archivos csv y me apoye en IA para las operaciones principales de repositorio ya que no es la persistencia usual
* Con IA también genere los datos base con los que se probo el servicio
* Gracias al clean arcitecture se dejo toda la relación entre el repositorio y los servicios por medio de interfacese en caso de que se pueda tomar ese ticket como strech

Al encontrarme con el descuento por producto decidí separarlo desde el modelo de datos, ya que el descuento puede ser por producto especifico(hijo) todos los productos en general o a nivel de vendedor, por lo cual me pareció mas lógico separar esta modelo

Decidí devolver la información en formato JSON con toda la información necesaria para crear el front end

La relación recursiva entre producto me permitió generar el modelo de forma mas sencilla, sin embargo falta validar la lógica del descuento en el precio base

Si se solicita un listing (publicación) con fecha el va a calcular todas las ofertas activas, ya que estas pueden vencer

Se presupone que las imágenes tienen otra fuente (bucket S3 por ejemplo) las cuales se extraerán con otro micro

Se incluye Swagger para que quede mejor documentado y quede mas fácil a los desarrolladores front end implementar lo necesario para consumir el API

# Futuras mejoras

Para próximos sprints se incluirán los siguientes requisitos:

## Funcionales

* Modelo Front End para el consumo por medio de una UI
* Informacion de productos relacionados extraída a partir de un id de listing buscando pro categoría y precio
* Micro para el manejo de calificaciones del vendedor (tiempo de respuesta, tiempo de envio)
* Micro para el manejo de las imágenes del producto

## No funcionales

* Implementación de base de datos Postgre, con una imagen de Docker contenerizado para ejecución de pruebas locales
* Inclusión de Jacoco para exigir 80% de code coverage