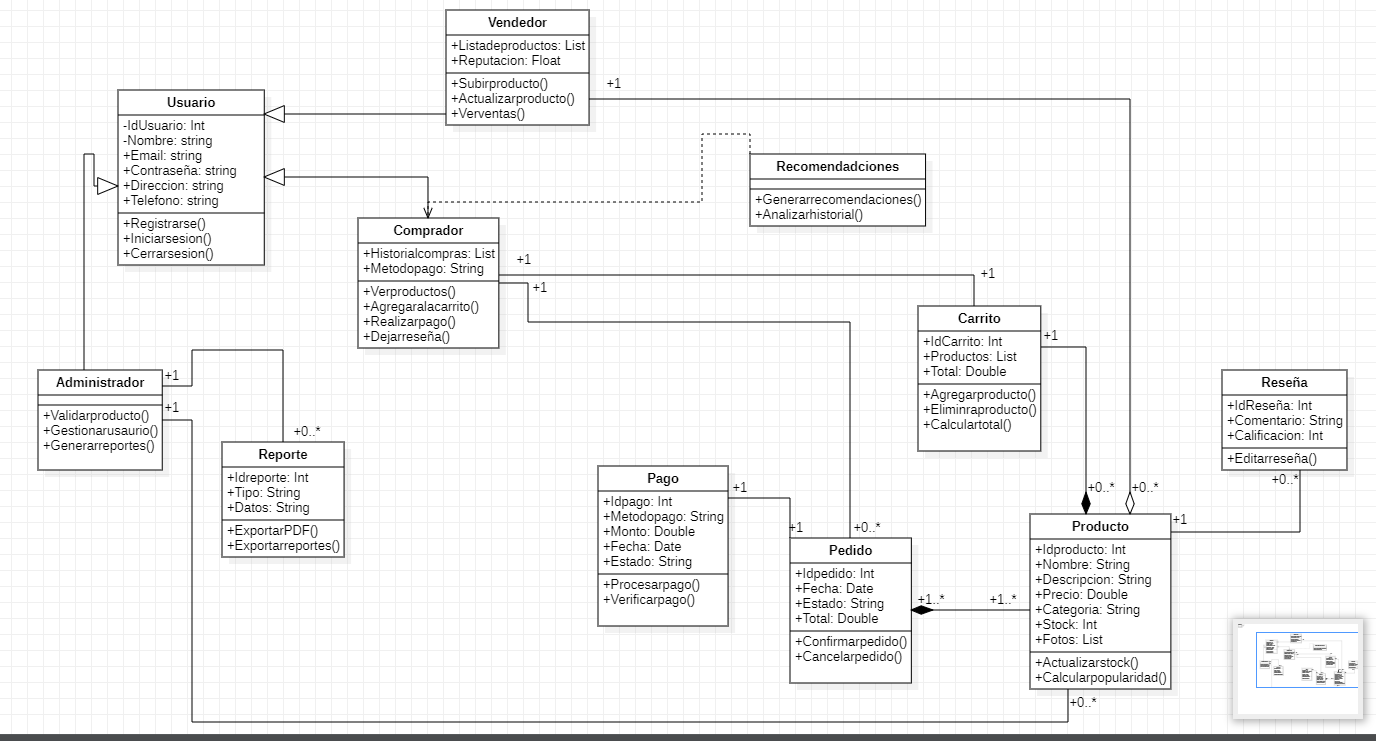
Informe Diagrama de clases



Relaciones

. Usuario – Comprador – Vendedor – Administrador

* Relación: Herencia (generalización).
* Justificación:  
  + La clase Usuario es la clase padre que contiene atributos y métodos comunes (id, nombre, email, dirección, teléfono, registro, inicio de sesión, cierre de sesión).
  + Comprador, Vendedor y Administrador heredan de Usuario, porque todos son un tipo de usuario pero con funcionalidades especializadas.
  + Esto sigue el principio de reutilización de código: los métodos comunes se definen en la superclase y las subclases implementan funciones específicas.

2. Comprador – Carrito

* Relación: Asociación 1 a 1.
* Justificación:  
  + Cada comprador debe tener un único carrito activo para gestionar sus productos antes de comprar.
  + La multiplicidad 1..1 asegura que el carrito siempre está asociado a un único comprador y no puede compartirse entre usuarios, lo cual es lógico en un entorno de e-commerce.

3. Carrito – Producto

* Relación: Asociación de composición (*0..* productos en el carrito\*).
* Justificación:  
  + Un carrito se compone de una lista de productos.
  + Si el carrito se elimina, los productos en esa instancia del carrito dejan de estar asociados (aunque los productos sigan existiendo en el catálogo).
  + La multiplicidad 0..\* permite agregar y eliminar productos dinámicamente.

4. Comprador – Pedido

* Relación: Asociación 1 a muchos.
* Justificación:  
  + Un comprador puede generar múltiples pedidos a lo largo del tiempo.
  + La multiplicidad 0..\* en el lado de pedido representa el historial de compras.
  + Mantener esta relación es importante para funcionalidades como seguimiento de pedidos, devoluciones y generación de reportes.

5. Pedido – Producto

* Relación: Asociación de agregación (0..\* productos en un pedido).
* Justificación:  
  + Cada pedido está compuesto por una lista de productos seleccionados del carrito en el momento de la compra.
  + La agregación indica que los productos existen independientemente del pedido (los productos no se borran si se elimina el pedido).

6. Pago – Pedido

* Relación: Asociación 1 a 1.
* Justificación:  
  + Cada pedido debe tener exactamente un pago asociado que valide la transacción.
  + Esto asegura trazabilidad en el sistema contable y previene duplicidad de pagos.

7. Producto – Reseña

* Relación: Asociación 0..\*.
* Justificación:  
  + Un producto puede tener muchas reseñas de distintos compradores.
  + Permite analizar la satisfacción del cliente y generar confianza en otros compradores.

8. Comprador – Reseña

* Relación: Asociación 0..\*.
* Justificación:  
  + Un comprador puede dejar varias reseñas, una por cada producto adquirido.
  + Se puede restringir para evitar reseñas de productos que no fueron comprados.

9. Administrador – Reporte

* Relación: Asociación 0..\*.
* Justificación:  
  + El administrador genera reportes de ventas, usuarios y productos.
  + La multiplicidad 0..\* permite que se generen múltiples reportes a lo largo del tiempo.

10. Recomendaciones – Producto / Comprador

* Relación: Asociación.
* Justificación:  
  + La clase Recomendaciones genera sugerencias personalizadas con base en el historial de compra del comprador y en los productos disponibles.
  + Esta relación es importante para mejorar la experiencia de usuario mediante recomendaciones automáticas.

Conclusiones

El diseño sigue principios de POO (herencia, agregación, composición) y está alineado con la lógica de un sistema de comercio electrónico.  
 Las relaciones permiten:

* Reutilización de código (gracias a la herencia de Usuario).
* Integridad en el flujo de compra (carrito → pedido → pago).
* Escalabilidad (se pueden agregar nuevos métodos y clases sin romper el modelo).
* Experiencia de usuario mejorada (reseñas, recomendaciones).
* Gestión administrativa eficiente (reportes, validación de productos).