Informe Técnico: Sistema de Gestión de Stock

1. Diseño de modelos y relaciones

El sistema fue desarrollado utilizando **Django** y **Django ORM** para la persistencia de datos. El diseño de los modelos busca representar los elementos principales de una bodega o inventario y sus relaciones.

Modelos principales

- **Producto**
  - Campos: nombre, código, descripción, stock_actual, precio_unitario.
  - Representa cada artículo disponible en la bodega.
- **Movimiento**
  - Campos: producto (FK a Producto), tipo (entrada o salida), cantidad, fecha, usuario.
  - Registra cada acción sobre el stock, asegurando trazabilidad.
- **Proveedor** (opcional, según necesidades)
  - Campos: nombre, rut, contacto.
  - Se relaciona con **Movimiento** para entradas de stock.
- **Cliente** (opcional)
  - Campos: nombre, rut, email.
  - Se relaciona con **Movimiento** para salidas de stock.

Relaciones

- **Un Producto puede tener muchos Movimientos.**
- **Cada Movimiento pertenece a un Producto.**
- Los Movimientos se diferencian por tipo (entrada o salida) y modifican el stock del producto asociado.

El diagrama lógico es sencillo y escalable:

Producto (1) --- (N) Movimiento

2. Lógica de los movimientos y reglas de stock

Funcionalidades implementadas

1. **Listar movimientos (listar_movimientos)**
    a. Recupera todos los movimientos ordenados por fecha descendente.
    b. Se renderiza en la vista "movimiento/listar.html".
2. **Crear movimiento (crear_movimiento)**
    a. Usa un formulario (MovimientoForm).
    b. Si es **ENTRADA**, suma la cantidad al stock del producto.
    c. Si es **SALIDA** o **MERMA**, valida que el stock no quede negativo.
    d. Se actualiza el stock del producto y se guarda el movimiento.
    e. Usa messages.error para avisar cuando no hay stock suficiente.
3. **Histórico de un producto (historial_producto)**
    a. Filtra los movimientos de un producto en particular.
    b. Ideal para trazabilidad.
4. **Editar movimiento (editar_movimiento)**
    a. Permite modificar un movimiento ya creado.
    b. Ojo 👀: actualmente solo actualiza el movimiento, **pero no recalcula el stock** en caso de cambio.
    c. Podrías mejorar esto recalculando stock si cambia el tipo/cantidad.
5. **Eliminar movimiento (eliminar_movimiento)**
    a. Antes de eliminar, **revierte el efecto del movimiento en el stock**.
    b. Ejemplo: si eliminas una ENTRADA, se resta stock; si eliminas una SALIDA, se vuelve a sumar.

Conclusiones

El sistema cumple con:

- Mantener la integridad del stock en cada operación.
- Garantizar trazabilidad de los movimientos.
- Evitar inconsistencias mediante reglas de negocio simples pero efectivas.

El diseño con modelos independientes y relaciones bien definidas permite **escalabilidad**, pudiendo agregar módulos de proveedores, clientes o reportes en el futuro sin modificar la lógica principal.