Sequence Diagram - Alur Pembaruan Info Pengiriman

Diagram ini memvisualisasikan urutan interaksi sistem saat User Pemilik atau User Pegawai menambahkan informasi pengiriman (kurir dan nomor resi) ke sebuah pesanan yang sudah ada.

Aktor dan Komponen yang Terlibat

1. **User (Pemilik/Pegawai):** Pengguna yang menginput data pengiriman di dasbor.
2. **main-app (PHP):** Aplikasi backend yang memproses logika.
3. **postgres-db (Database):** Tempat menyimpan dan memperbarui data.

Visualisasi Diagram (PlantUML)

Gunakan kode berikut di PlantUML Web Server untuk menghasilkan diagram visual dari alur ini.

Prompt untuk PlantUML:

```
@startuml
' Diagram Urutan untuk Alur Pembaruan Info Pengiriman

title Alur Proses Pembaruan Pesanan dengan Info Pengiriman

actor "User (Pemilik/Pegawai)" as User
participant "main-app (PHP)" as App
database "postgres-db" as DB

User -> App : 1. POST /orders/{id}/ship (nama_kurir, no_resi)
activate App

App -> App : 2. Validasi data input (resi tidak boleh kosong)

App -> DB : 3. START TRANSACTION
activate DB

App -> DB : 4. Buat Catatan Pengiriman (shipments)\nINSERT INTO shipments (order_id, courier_name, ...)
activate DB
DB --> App : 5. Konfirmasi shipment dibuat
deactivate DB

App -> DB : 6. Perbarui Status Pesanan (orders)\nUPDATE orders SET order_status = 'shipped'
```

```
WHERE order_id = {id}
activate DB
DB --> App : 7. Konfirmasi status diperbarui
deactivate DB

App -> DB : 8. COMMIT TRANSACTION
activate DB
DB --> App : 9. Transaksi Sukses
deactivate DB

App --> User : 10. Response (Redirect kembali ke detail pesanan)
deactivate App

@enduml
```

Penjelasan Langkah-demi-Langkah

1. **POST /orders/{id}/ship**
    - **Aksi:** Di halaman detail pesanan, pengguna mengklik tombol "Kirim Pesanan", lalu mengisi formulir dengan nama kurir dan nomor resi, kemudian menyimpannya.
    - **Aliran:** Peramban mengirimkan data tersebut ke aplikasi PHP, menargetkan pesanan dengan ID spesifik.
2. **Validasi data input**
    - **Aksi:** Aplikasi memastikan data yang diterima valid, terutama nomor resi tidak boleh kosong.
3. **START TRANSACTION**
    - **Aksi:** Aplikasi memulai transaksi database. Ini sangat penting untuk memastikan kedua operasi (membuat catatan pengiriman dan memperbarui status pesanan) berhasil bersamaan.
4. **Buat Catatan Pengiriman (shipments)**
    - **Aksi:** Aplikasi membuat baris data baru di tabel shipments. Karena ada hubungan *one-to-one* antara orders dan shipments, catatan ini dibuat dengan order_id yang sesuai.
    - **Aliran:** PHP mengirim perintah INSERT ke tabel shipments.
5. **Konfirmasi shipment dibuat**
    - **Aliran:** Database mengkonfirmasi bahwa catatan pengiriman berhasil dibuat.
6. **Perbarui Status Pesanan (orders)**
    - **Aksi:** Ini adalah langkah logis berikutnya. Setelah data pengiriman disimpan, status pesanan di tabel orders diperbarui menjadi 'shipped'.
    - **Aliran:** PHP mengirim perintah UPDATE ke tabel orders.
7. **Konfirmasi status diperbarui**
    - **Aliran:** Database mengkonfirmasi bahwa status pesanan berhasil diperbarui.
8. **COMMIT TRANSACTION**

    - **Aksi:** Karena kedua operasi database berhasil, aplikasi menyimpan semua perubahan secara permanen. Jika salah satu dari langkah #4 atau #6 gagal, transaksi akan dibatalkan (ROLLBACK) dan tidak ada perubahan yang disimpan, sehingga data tetap konsisten.
9. **Response (Redirect ...)**
    - **Aksi:** Aplikasi mengirimkan respons sukses kembali ke pengguna.
    - **Aliran:** Biasanya berupa *redirect* kembali ke halaman detail pesanan, yang kini akan menampilkan status "Dikirim" beserta informasi kurir dan nomor resi.

Ini adalah contoh bagus bagaimana Sequence Diagram membantu memastikan integritas data dalam alur kerja operasional. Apakah ada alur lain yang ingin kita petakan?