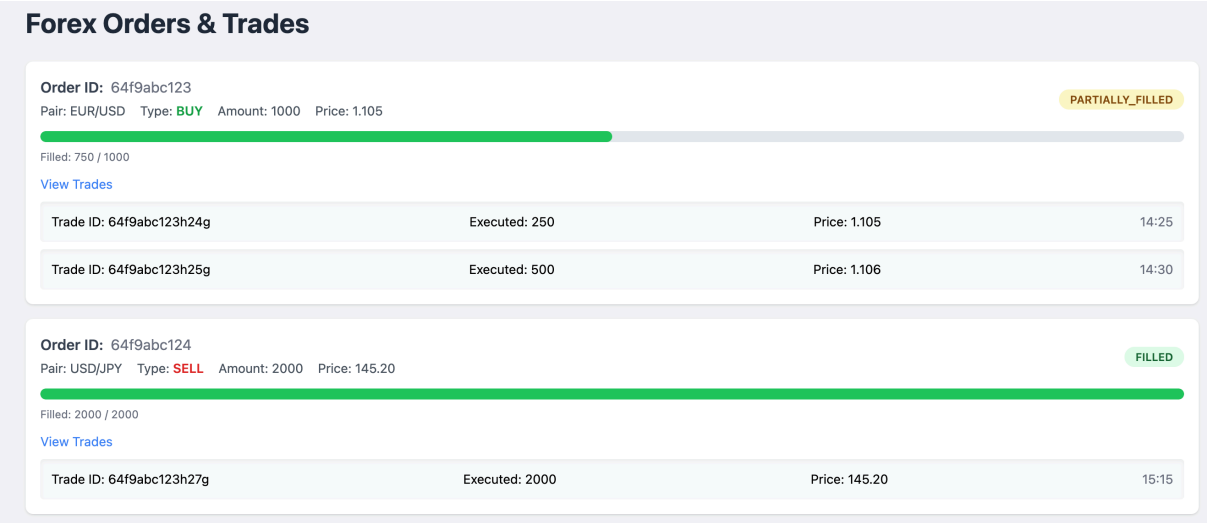
Server Interview – Orders & Trades

This module handles **orders with nested trades** and **real-time updates** for a Forex trading system.

The **screenshot** illustrates how orders and trades should appear use it as the basis for your schema and structure design.



Requirements

Orders

- Contain details such as **pair, type, amount, price, status**.
- Have a list of **trades** beneath them.
- Support status progression: `PENDING → PARTIALLY_FILLED → FILLED`

Trades

- Represent **individual executions** (amount, price, timestamp).
- Adding trades updates the order's **filled progress** and **status** automatically.

Business Rules

- The sum of all executed trades must not exceed the order's amount.
- If total filled `< amount` → `PARTIALLY_FILLED`.
- If total filled `= amount` → `FILLED`.
- If adding a trade would overshoot → reject (400).

API Endpoints (JWT required)

Method	Path	Purpose
POST	`/orders`	Create a new order
POST	`/orders/:id/trades`	Add a trade (updates order status automatically)

Simulation (Swagger/Postman)

- **Create an order** (e.g., BUY EUR/USD 1000).
- **Add trades** (e.g., 250 + 500 executed).
- Verify that the order status updates automatically (`PARTIALLY_FILLED`, then `FILLED`).

Deliverables

- Implement Orders module.
- Add API endpoints + validation.
- Test simulation with Swagger/Postman.