Requirements (E-Commerce Mini Backend)

Overview

Build a minimal e-commerce backend for two roles: - **Staff**: manage products and inventory. - **Customers**: browse products, save credit cards, and place purchases.

Core Use Cases

Staff 1. Add a product with name, description, price, and stock quantity. 2. Update product price and stock. 3. List/search products by name and price range. 4. View products low on stock (e.g., stock_qty < 5).

Customer 1. Register with name and email (unique). 2. Add one or more credit cards (brand, last4, exp month/year, token placeholder). 3. Browse/search products and view price/stock. 4. Place a purchase with one or more items (qty > 0). Stock is reduced accordingly.

Functional Requirements

- A purchase must contain at least one item.
- Total purchase amount equals the sum of `qty * unit_price` for all items.
- Prevent purchasing quantities that exceed available stock.
- Allow customers to store multiple cards.
- Products can be marked active/inactive; only active products are shown to customers.

Non-Functional Requirements

- Use **PostgreSQL**.
- Provide a **single SQL file** (`sql/db.sql`) with DDL and INSERT seed data.
- Provide **5–8 SQL queries** demonstrating multi-table JOINs (`sql/queries.sql`).
- Provide a small **application** that connects to the DB. Here: **FastAPI** + simple HTML frontend.
- Provide a **video** demo (schema, queries, app actions).

Data Validation & Constraints

- Emails must be unique.
- Price and quantities are non-negative.
- Credit card brand limited to common values.
- Referential integrity via foreign keys.
- CHECK constraints to enforce domain rules (e.g., qty > 0).

Out of Scope (for simplicity)

- Real payment processing (use a token placeholder).
- Authentication/authorization (assume trusted use).
- Shipping, returns, promotions (optional extensions).