**Project Problem Statement: E-Commerce Management Portal**

**1. Background**

E-commerce has transformed the way customers shop and businesses sell products. However, developing a scalable and reliable portal that manages **products, users, orders, payments, and inventory** is still a challenge.

Small retailers often lack affordable, lightweight solutions and end up with either:

* Overly complex platforms that require heavy infrastructure
* Simple systems without real-time inventory, order tracking, or automated testing

The **E-Commerce Management Portal** project will address these issues by building a **console-based system in Core Java**, backed by **PostgreSQL for persistent data storage**, and **Selenium** to automate functional testing of key workflows.

**2. Problem Statement**

The objective is to **design and implement an E-Commerce Management Portal** that:

* Allows **Customers** to browse products, add to cart, checkout, and manage orders
* Allows **Admins** to add/manage products, track inventory, and view sales reports
* Uses **PostgreSQL** for reliable and scalable data storage
* Implements **automated testing using Selenium** for login, product search, cart, and checkout

**3. Objectives**

1. Implement backend logic using **Core Java** (OOP, JDBC, exception handling, multithreading where needed).
2. Use **PostgreSQL** to persist and manage data (products, users, orders, payments, inventory).
3. Provide **role-based access control** (Customer vs. Admin).
4. Handle **real-time inventory updates** during purchases.
5. Automate functional flows using **Selenium WebDriver** (UI regression testing).
6. Ensure modular, maintainable code for scalability.

**4. System Requirements**

**Functional Requirements**

* **Customer Module**
  + Register & Login
  + Browse/Search Products (by category, name, price range)
  + Add to Cart / Remove from Cart
  + Checkout & Make Payment
  + Track Order Status (Placed, Shipped, Delivered, Cancelled)
  + View Order History
* **Admin Module**
  + Admin Login
  + Add/Update/Delete Products
  + Manage Categories & Inventory
  + View All Orders & Customer Details
  + Generate Reports (e.g., total sales, best-selling products)
* **Order/Cart Module**
  + Real-time inventory deduction at checkout
  + Prevent overselling if multiple customers buy simultaneously
  + Generate unique Order ID + invoice details
* **Testing Module (Selenium)**
  + Automated test cases for:
    - Login (Customer/Admin)
    - Product Search
    - Add to Cart & Checkout
    - Admin product management

**Non-Functional Requirements**

* **Performance:** Support multiple concurrent customer sessions
* **Reliability:** Ensure accurate inventory tracking
* **Scalability:** Extendable to new categories and product types
* **Usability:** Simple **console-based menu system** for easy navigation

**5. Database Schema (PostgreSQL)**

**Tables**

**1. Users (Customers/Admins)**

CREATE TABLE users (

user\_id SERIAL PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

password VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15),

role VARCHAR(10) CHECK (role IN ('CUSTOMER','ADMIN')) DEFAULT 'CUSTOMER'

);

**2. Categories**

CREATE TABLE categories (

category\_id SERIAL PRIMARY KEY,

name VARCHAR(50) NOT NULL UNIQUE

);

**3. Products**

CREATE TABLE products (

product\_id SERIAL PRIMARY KEY,

name VARCHAR(100) NOT NULL,

category\_id INT REFERENCES categories(category\_id),

price DECIMAL(10,2) NOT NULL,

stock INT NOT NULL,

description TEXT

);

**4. Cart**

CREATE TABLE cart (

cart\_id SERIAL PRIMARY KEY,

user\_id INT REFERENCES users(user\_id) ON DELETE CASCADE,

product\_id INT REFERENCES products(product\_id),

quantity INT NOT NULL

);

**5. Orders**

CREATE TABLE orders (

order\_id SERIAL PRIMARY KEY,

user\_id INT REFERENCES users(user\_id) ON DELETE CASCADE,

order\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status VARCHAR(20) CHECK (status IN ('PLACED','SHIPPED','DELIVERED','CANCELLED')) DEFAULT 'PLACED',

total\_amount DECIMAL(10,2) NOT NULL

);

**6. Order Items**

CREATE TABLE order\_items (

order\_item\_id SERIAL PRIMARY KEY,

order\_id INT REFERENCES orders(order\_id) ON DELETE CASCADE,

product\_id INT REFERENCES products(product\_id),

quantity INT NOT NULL,

price DECIMAL(10,2) NOT NULL);