Báo cáo: E-commerce Product Management System

1. Phân tích OOA (Object-Oriented Analysis)

Bước 1: Xác định đối tượng (Objects)

- Product (sản phẩm)
- Electronics (sản phẩm điện tử, kế thừa từ Product)
- ShoppingCart (giỏ hàng)
- Order (đơn hàng – có thể mở rộng sau)
- InventoryList (danh sách kho hàng – dùng template)
- Discountable (giao diện giảm giá)

Bước 2: Xác định thuộc tính (Attributes)

- Product: id, name, price, stock
- Electronics: kế thừa Product, thêm warrantyMonths
- ShoppingCart: danh sách sản phẩm + số lượng
- InventoryList<T>: vector<T> items
- Order: danh sách sản phẩm, tổng tiền (mở rộng)

Bước 3: Xác định phương thức (Methods)

- Product: display(), applyDiscount(), get/set cho các thuộc tính
- Electronics: display() override
- ShoppingCart: addToCart(), showCart(), operator+=
- InventoryList<T>: addItem(), showAll(), getItems()
- Discountable: applyDiscount() (pure virtual)

Bước 4: Xác định quan hệ kế thừa

- Electronics **generalization** từ Product
- Product **realization** của interface Discountable
- ShoppingCart sử dụng (composition) Product
- InventoryList là template quản lý Product

2. Thiết kế lớp và giải thích

- **Kế thừa (Inheritance):**
  Lớp Electronics được xây dựng dựa trên lớp Product, mở rộng thêm thuộc tính về thời gian bảo hành (warrantyMonths). Đây là mối quan hệ "is-a", nghĩa là Electronics cũng là một loại Product.
- **Nạp chồng toán tử (Operator overloading):**
  Chương trình định nghĩa toán tử so sánh bằng (==) trong lớp Product để so sánh hai sản

phẩm dựa trên ID. Ngoài ra, lớp ShoppingCart cũng nạp chồng toán tử cộng bằng (+=) để thêm sản phẩm vào giỏ hàng. Nhờ đó, thao tác thêm sản phẩm trở nên tự nhiên và gần gũi hơn với cách viết toán học.

- **Interface (Abstract class):**
  Lớp Discountable định nghĩa phương thức thuần ảo applyDiscount(rate). Tất cả các lớp kế thừa từ Product đều buộc phải cài đặt phương thức này, nhờ đó đảm bảo được tính đa hình khi áp dụng giảm giá cho nhiều loại sản phẩm khác nhau.
- **Template class:**
  Lớp InventoryList<T> được cài đặt dưới dạng template để có thể lưu trữ nhiều kiểu đối tượng khác nhau, chẳng hạn Product, Electronics hoặc các loại khác trong tương lai. Việc sử dụng template làm cho mã nguồn trở nên linh hoạt và có khả năng tái sử dụng cao hơn.

3. Walkthrough mã nguồn

- **Lớp cơ sở Product:**
  Lưu trữ thông tin cơ bản của sản phẩm gồm mã sản phẩm (id), tên (name), giá (price) và số lượng trong kho (stock). Đồng thời, lớp này cài đặt phương thức applyDiscount để điều chỉnh giá bán khi áp dụng giảm giá.
- **Lớp Electronics:**
  Kế thừa từ Product và ghi đè phương thức display để hiển thị thêm thông tin về thời gian bảo hành.
- **Nạp chồng toán tử:**
  - Trong lớp Product, toán tử so sánh bằng (==) được sử dụng để so sánh hai sản phẩm dựa trên ID.
  - Trong lớp ShoppingCart, toán tử cộng bằng (+=) được sử dụng để thêm sản phẩm vào giỏ hàng nếu còn hàng.
- **Template InventoryList<T>:**
  Được cài đặt với vector để quản lý danh sách sản phẩm. Các phương thức chính gồm có addItem, showAll và getItems.
- **Lớp ShoppingCart:**
  Thay vì chỉ lưu trữ danh sách con trỏ Product, chương trình cài đặt thêm một cấu trúc CartItem với hai thành phần: con trỏ Product và số lượng (quantity). Nhờ đó, giỏ hàng có thể quản lý số lượng sản phẩm và tự động trừ tồn kho khi thêm sản phẩm.

4. Kết quả kiểm thử

Case 1: Thêm sản phẩm vào kho

- Input: Product id = P01, name = Book, price = 50, stock = 10
- Output: Inventory hiển thị sản phẩm vừa thêm.

Case 2: Thêm sản phẩm điện tử

- Input: Electronics id = E01, name = Laptop, price = 1500, stock = 5, warranty = 24
- Output: Inventory hiển thị Laptop với bảo hành 24 tháng.

Case 3: Thêm vào giỏ hàng

- Input: Chọn P01, quantity = 3
- Output: Giỏ hàng hiển thị 3 Book, tồn kho giảm còn 7.

Case 4: Xem giỏ hàng nhiều lần

- Output: Vẫn hiển thị đúng số lượng, không crash.

5. UML Diagrams

Class Diagram (mô tả):

- Product: id, name, price, stock, +display(), +applyDiscount(), +operator==
- Electronics: +warrantyMonths, override display()
- Discountable: +applyDiscount() (abstract)
- InventoryList<T>: +addItem(), +showAll(), +getItems()
- ShoppingCart: +addToCart(), +showCart(), operator+=

Sequence Diagram (thao tác "Add to cart"):

1. User nhập productID + quantity
2. Hệ thống tìm sản phẩm trong InventoryList
3. Nếu tìm thấy, gọi ShoppingCart.addToCart()
4. ShoppingCart kiểm tra tồn kho, trừ số lượng, thêm vào cart
5. Hiển thị thông báo "Added successfully"

6. Sử dụng LLM (AI Tool)

Trong quá trình làm bài, em có sử dụng ChatGPT để:

- Brainstorm ý tưởng về cách cài đặt template InventoryList.
- Tham khảo cách viết operator overloading cho giỏ hàng.

Em đã hỏi "Suggest a template class for inventory in C++" và sau đó em điều chỉnh lại để phù hợp với yêu cầu đề tài.