Business Case Explanation

In this case, we are managing an inventory system for a store that sells various types of products, such as electronics, clothing, food, furniture, and appliances. The goal is to identify and group products based on their availability in stock and to organize them by their categories.

Objectives:

1. Identify Products Out of Stock: Filter out the products that have no stock left.
2. List Categories with No Stock: Determine which product categories have at least one product that is out of stock.
3. Group Products by Category: Organize the products into groups based on their category, for easier access and analysis.

Solution Implementation

Step 1: Use of Inheritance

To make the code more maintainable and modular, inheritance was introduced. Instead of treating all products the same, we created a base class Product that contains common attributes and methods for all product types. Then, we created derived classes for specific product categories like Electronics, Clothing, Food, etc.

- Base Class (Product): Contains common attributes like name, category, and quantityOfStock.
- Derived Classes (Electronics, Clothing, Food, etc.): Inherit from the base class Product and represent specific product categories.

This approach allows us to easily add new product categories in the future without modifying the existing code structure.

Step 2: Refactoring the Main Logic

With the new class structure, we refactored the main logic to handle the following tasks:

1. Creating a List of Products: A list of products was created using the specific derived classes, making it clear which category each product belongs to.
2. Filtering Products Out of Stock:
    - The products were filtered using a stream operation to identify those with a quantityOfStock of 0.
    - The filter() method was used to find these products, and the result was collected into a list using collect(Collectors.toList()).
3. Listing Categories with No Stock:
    - The filtered list was further processed to extract the categories of the out-of-stock products.
    - The map() method was used to extract the category from each product, and distinct() was applied to remove duplicates.
4. Grouping Products by Category:
    - The products were grouped by category using the Collectors.groupingBy() method.
    - This grouping allows us to easily access and analyze products based on their category.