Coding Challenge - Order Management System

Instructions

- Project submissions should be done through the partcipants' Github repository, and the link should be shared with trainers and Hexavarsity.
- Each section builds upon the previous one, and by the end, you will have a comprehensive **Order Management System** implemented with a strong focus on **SQL**, **control flow statements, loops, arrays, collections, exception handling**, **database interaction**.
- Follow **object-oriented principles** throughout the project. Use classes and objects to model real-world entities, **encapsulate data and behavior**, and **ensure code reusability**.
- Throw **user defined exceptions** from corresponding methods and handled**.**
- The following **Directory structure** is to be followed in the application.
    - **entity/model**
        - Create entity classes in this package. All entity class should not have any business logic.
    - **dao**
        - Create Service Provider interface to showcase functionalities.
        - Create the implementation class for the above interface with db interaction.
    - **exception**
        - Create user defined exceptions in this package and handle exceptions whenever needed.
    - **util**
        - Create a **DBPropertyUtil** class with a static function which takes property file name as parameter and returns connection string.
        - Create a **DBConnUtil** class which holds **static method** which takes connection string as parameter file and returns **connection object(Use method defined in DBPropertyUtil class to get the connection String)**.
- **main**
    - Create a class MainModule and demonstrate the functionalities in a menu driven application.

Problem Statement:

Create SQL Schema from the product and user class, use the class attributes for table column names.

1. Create a base class called **Product** with the following attributes:
    - **productId** (int)
    - **productName** (String)
    - **description** (String)
    - **price** (double)
    - **quantityInStock** (int)
    - **type** (String) [Electronics/Clothing]
2. Implement constructors, getters, and setters for the **Product** class.
3. Create a subclass **Electronics** that inherits from **Product**. Add attributes specific to electronics products, such as:

- **brand** (String)
- **warrantyPeriod** (int)

4. Create a subclass **Clothing** that also inherits from **Product**. Add attributes specific to clothing products, such as:
    - **size** (String)
    - **color** (String)
5. Create a **User** class with attributes:
    - **userId** (int)
    - **username** (String)
    - **password** (String)
    - **role** (String) // "Admin" or "User"
6. Define an interface/abstract class named **IOrderManagementRepository** with methods for:
    - **createOrder(User user, list of products):** check the user as already present in database to create order or create user (store in database) and create order.
    - **cancelOrder(int userId, int orderId):** check the userid and orderId already present in database and cancel the order. if any userId or orderId not present in database throw exception corresponding **UserNotFound or OrderNotFound exception**
    - **createProduct(User user, Product product):** check the admin user as already present in database and create product and store in database.
    - **createUser(User user):** create user and store in database for further development.
    - **getAllProducts():** return all product list from the database.
    - **getOrderByUser(User user):** return all product ordered by specific user from database.
7. Implement the **IOrderManagementRepository** interface/abstractclass in a class called **OrderProcessor**. This class will be responsible for managing orders.
8. Create **DBUtil** class and add the following method.
    - **static getDBConn():Connection** Establish a connection to the database and return database Connection
9. Create **OrderManagement** main class and perform following operation:
    - main method to simulate the loan management system. Allow the user to interact with the system by entering choice from menu such as "createUser", "createProduct", "cancelOrder", "getAllProducts", "getOrderbyUser", "exit".