Documentation

**Project Overview**

You are tasked with developing a comprehensive Inventory Management System for a small store. This system should manage various types of items, handle payments, and process orders. Implement interfaces, abstract classes, and multiple concrete classes.

**Part 1: Inventory Management System**

**Interfaces and Abstract Classes**

**1. Item Interface**

**Description:** Represents items in the inventory.

**Methods:**

* getItemDetails(): Returns the details of the item.
* calculateValue(): Calculates the value of the item.
* displayDescription(): Displays the item's description.

**2. Categorizable Interface**

**Description:** Represents items that can be categorized.

**Methods:**

* setCategory(String category): Sets the category of the item.
* getCategory(): Gets the category of the item.

**3. Breakable Interface**

**Description:** Indicates items that can break.

**Methods:**

* isBreakable(): Checks if an item is breakable.
* handleBreakage(): Handles the item breakage.

**4. Perishable Interface**

**Description:** Represents items that can perish.

**Methods:**

* isPerishable(): Checks if an item is perishable.
* handleExpiration(): Handles the item expiration.

**5. Sellable Interface**

**Description:** Represents items that can be sold.

**Methods:**

* setPrice(double price): Sets the price of the item.
* getPrice(): Gets the price of the item.

**6. Abstract Item Class**

**Description:** Implements the Item, Categorizable, Breakable, Perishable, and Sellable interfaces.

**Attributes:**

* String name
* double price
* String category
* boolean breakable
* boolean perishable

**Methods:**

* Implements common functionality such as getting item details.
* Provides default implementations for category, breakable, perishable, and sellable attributes.

**Superclasses and Inheritance**

**7. InventoryItem class**

**Description:** Extends AbstractItem.

**Attributes:**

* int itemID
* int quantity

**Methods:**

* Getters and setters for ID and quantity.

**8. Item Types**

**Description:** Create subclasses for specific item types.

**Subclasses:**

* ElectronicsItem: Inherits from InventoryItem, with additional attributes specific to electronics.
* GroceryItem: Inherits from InventoryItem, with additional attributes specific to groceries.
* FragileItem: Inherits from InventoryItem, with additional attributes specific to fragile items.

**Methods:**

* Constructors to set specific attributes.
* Override relevant methods to calculate item values differently.

**File I/O, User Interface, Payments, and Orders**

**9. InventorySystem class**

**Description:** The Inventory class is responsible for managing a collection of InventoryItem objects and Order objects. It provides methods to add, remove, retrieve items by ID, and handle file I/O operations for saving and loading the inventory.

**Attributes**

* private List<InventoryItem> items: A list to store the inventory items.
* private List<Order> ordes: A list to store the orders.

**Methods**

* addItem(InventoryItem item): Adds an item to the inventory.
* removeItemById(int itemId): Removes an item from the inventory by its ID.
* getItemById(int itemId): Retrieves an item from the inventory by its ID.
* getItems(): Retrieves all items in the inventory.
* getOrdes(): Retrieves all orders.
* addOrder(Order order): Adds an order to the list of orders.

**File I/O**

**Description:** Save and load inventory data to/from text files.

**Methods:**

* saveInventory(List<InventoryItem> inventory, String fileName): Saves the inventory data to a file.
* loadInventory(String filename): Loads the inventory data from a file.

**10. User Interface**

**Description:** Create a command-line interface (CLI) – InventorySystem class

**Features:**

* Add items, remove items by ID, display a list of items, categorize items, and place orders.
* Display a menu for user choices and handle user input gracefully.

**11. Payments and Orders (20 points)**

**Description:** Implement classes for Payment and Order.

**Classes:**

* Payment: Handles the payment processing.
* Order: Represents an order with details such as order ID, items, quantities, total cost, and payment method.

**Methods:**

* Calculate order totals and process payments.
* Update inventory quantities after orders are placed.

**Part 2: Payment Processing**

**1. Payment Processor**

**Description:** Handle payments.

**Class:** PaymentProcessor

**Methods:**

* processPayment(PaymentMethod method): Processes payments using various payment methods.
* Validation for payment methods and simulate payment authorization.

**2. Payment Methods**

**Description:** Different payment methods.

**Interfaces/Abstract Classes:**

* CreditCardPayment
* PayPalPayment

**Attributes:**

* Appropriate attributes such as card number, PayPal account.
* Validation for payment methods.

**Part 3: User Interface Enhancement**

Update InventorySystem class

**Features:**

* Select and purchase items.
* Shopping cart functionality to add items to the cart, view the cart, and place orders.
* Integrate payment processing into the ordering process.

**Part 4: Order Processing**

Update Order class

**Attributes:**

* Order ID, items, quantities, total cost, and payment method.

**Methods:**

* Calculate the order total.
* Process payments.
* Update inventory quantities.