1. Design class for "Product"

Table 1. Attribute design

#	Name	Data type	Default value	Description
1	description	string	""	Description of the product.
2	Barcode	String	“”	Unique identifier of the product
3.	warehouseEntryDate	Date	Null	Date when the product enterd the warehouse
4	Weight	Int	0	Product weight
5	Dimensions	Int	0	Available quantity
6	Quantity	Int	0	Available quantity
7	currentPrice	Float	0.0	Price per unit
8	Category	String	“”	Product category
9	Value	Float	0.0	Product value
10	Title	String	“”	Product title

Table 2: Operation design

#	Name	Return type	Description (purpose)
1	getProductInfo(searchAttributes, sortRequest)	String	Retrieve formatted product information based on search criteria and sorting preferences
2	getProductInfo()	void	Retrieve basic product information without filters

Method: getProductInfo

- **Parameters:**
  - searchAttributes: any -> Criteria to filter product information

- sortRequest: any -> Sorting preferences for the returned information

- **Exceptions:**
  - InvalidAttributeException if search attributes are not valid
  - SortingException if sorting request cannot be processed
- **How to use parameters/attributes:**

1. Validate input parameters for searchAttributes and sortRequest
2. Filter the product information based on provided searchAttributes
3. Apply sorting based on sortRequest if provided
4. Format the filtered and sorted information as a String
5. Return the formatted product information

Method: getProductInfo

- **Parameters**: None
- **Exceptions:** None
- **How to use parameters/attributes:**
  1. Retrieve product information
  2. Format the information as String
  3. Return the formatted basic product information

2. Design class for "ViewProductController"

Table 1: Attribute design

No attributes defined in the diagram.

Table 2: Operation design

#	Name	Return type	Description (purpose)
1	getProductDetail	void	Retrieve and display detailed information for a specific product

2	getProductList()	void	Retrieve and display the list of all products
3	getProductList(searchAttributes, sortRequest)	void	Retrieve and display filtered product list based on search criteria and sorting preferences

Method: getProductDetail

- **Parameters:**
  - product: int -> the product to retrieve details for
- **Exceptions:**
  - ProductNotFoundException if product ID does not exist
- **How to use parameters/attributes:**

1. Validate the product ID
2. Retrieve detailed product information using the ID
3. Send the product data to ProductDetailScreen for display

Method: getProductList

- **Parameters:** None
- **Exceptions:** None
- **How to use parameters/attributes:**
  1. Retrieve the list of all available products
  2. Format the list for display
  3. Send the list to ProductListScreen for rendering

Method: getProductList(searchAttributes, sortRequest)

- **Parameters:**
  - searchAttributes: any -> Criteria to filter products
  - sortRequest: any -> Sorting preferences for the product list
- **Exceptions:**
  - InvalidSearchException if search attributes are invalid

    - SortingException if sorting request cannot be processed

- How to use parameters/attributes:

1. Validate input parameters

2. Call Product.getProductInfo(searchAttributes, sortRequest) to get the product list satisfied conditions

3. Format the list for display

4. Send the formatted list to ProductListScreen

3. Design class for "HomeScreen"

Table 1: Attribute design

#	Name	Data type	Default value	Description
1	productListScreen	ProductListScreen	null	Reference to the product list screen component

Table 2: Operation design

#	Name	Return type	Description (purpose)
1	homePage()	void	Display the main home page
2	homePage(searchAttributes, sortRequest)	void	Display the home page with filtered products

Method: homePage()

- **Parameters:** None

- **Exceptions**: None

- **How to use parameters/attributes:**

    1. Load default home page layout

    2. Display featured products and categories

    3. Show navigation options

Method: homePage(searchAttributes, sortRequest)

- **Parameters:**
    - searchAttributes: any -> Search criteria for products to display on home page
    - sortRequest: any -> Sorting preferences for displayed products
- **Exceptions:**
    - InvalidSearchException if search attributes are not valid
- **How to use parameters/attributes:**

1. Validate search attributes and sort request
2. Request filtered product list for ProductListScreen that satisfied conditions
3. Maintain search criteria for pagination or refinement

4. Design class for "ProductListScreen"

Table 1: Attribute design

#	Name	Data type	Default value	Description
1	listOfProducts	array<Product>	[]	Array containing products to be displayed in the list

Table 2: Operation design

#	Name	Return type	Description (purpose)
1	ProductList(listOfProducts)	void	Display a list of products on the screen
2	ViewProductDetails(product)	void	Navigate to detailed view of a specific product

Method: ProductList(listOfProducts)

- **Parameters:**
    - listOfProducts: array<Product> -> Array of products to display in the list
- **Exceptions:**
    - EmptyListException if product list is empty
- **How to use parameters/attributes:**

1. Validate input parameter to ensure it contains valid product data
2. Format the product data for display
3. Render the product list with appropriate layout and styling
4. Implement pagination if the list is long

Method: ViewProductDetails(product)

- **Parameters:**
    - product: Product -> The product to view in detail
- **Exceptions:**
    - ProductNotFoundException if product data is invalid
- **How to use parameters/attributes:**

1. Validate the product parameter
2. Request detailed product information from ViewProductController
3. Navigate to ProductDetailScreen
4. Pass the product data to ProductDetailScreen for display

5. Design class for "ProductDetailScreen"

Table 1: Attribute design

#	Name	Data type	Default value	Description
1	product	Product	null	The product being displayed in detail

Table 2: Operation design

#	Name	Return type	Description (purpose)
1	ProductDetail(product)	void	Display detailed information for a specific product

Method: ProductDetail(product)

- **Parameters:**
    - product: int -> the product to display in detail
- **Exceptions:**

    - ProductNotFoundException if product is invalid

- **How to use parameters/attributes:**

1. Validate the product ID
2. Request detailed product information from ViewProductController
3. Format the product information for display
4. Render the detailed product view with all available information
5. Display options for adding to cart