Use case description

Use case name: Update inventory and notice on deficiencies.

Textual description: a storage worker adds or removes items of specific product from store and storage, and the system updates accordingly. If there are deficiencies of this product, the system notices on it.

Actors: storage worker.

Pre-conditions: the product is in store.

Post-conditions: the product's amount in store and storage is updated, and a deficiency notice is sent to the worker if there is a deficiency in this product.

Main success scenario:

1. Storage worker selects a catalog number.
2. Storage worker selects if to add items or remove items.
3. Storage worker types expiration date of items he want to add/remove.
4. Storage worker types the number of items he wants to add/remove from storage.
5. Storage worker types the number of items he wants to add/remove from store.
6. Storage worker repeats steps 3-5 until indicates done.

Extensions:

a. Product reaches its minimal quantity or below.
    a. The system sends a deficiency notice to the worker.
b. Worker types catalog number which does not exist.
    a. The system sends "wrong catalog number" notice to the worker.
    b. The worker starts again.
c. Worker wants to remove more items from store/storage from specific expiration dates than there are.
    a. System sends "not enough quantity" message to worker.
    b. Worker starts again.
d. Worker wants to remove items from an expiration date which does not exist for this product.
    a. System sends "expiration date does not exist" message to worker.
    b. Worker starts again.
e. Worker types wrong data (for example, types the catalog number as "abc"...).
    a. System sends "give us good info please!" message to worker.
    b. Worker starts again.

Use case name: Update discount from products in categories.

Textual description: Storage worker types the new discount, selects the categories he wants the discount to be on, and the system updates accordingly.

Actors: Storage worker.

Pre-conditions: Valid discount, categories exist.

Post-conditions: All the products in these categories update their discount.

Main success scenarios:

1. Worker types the discount.
2. Worker selects the categories from main categories, sub-categories, and sizes.

Extensions:

a. Worke types invalid discount.
    a. System sends "discount should be between 0-100" message to worker.
    b. Worker starts again.
b. Worker selects categories which do not exist.
    a. System sends "category does not exist" message to worker.
    b. Worker starts again.
c. Worker types wrong data.
    a. System sends "give us good info please!" message to worker.
    b. Worker starts again.

Contracts

Contract CO1: Update inventory.

Operation: Cross References:	updateInventory() Use cases: update inventory
Pre-conditions:	The product is in store Valid amount Valid expiration dates
Post-condition:	Product's amounts are updated

Contract CO2: update discount for categories.

Operation: Cross References:	updateDiscountForCategories() Use cases: update discount for categories
Pre-conditions:	Valid discount Categories exist
Post-condition:	Product's discount was updated