# Use case:

|  |  |  |
| --- | --- | --- |
| create order due to lack | create circulating order | Use case name |
| Creating an order that will be sent when there is a lack of particular product | Create an order that will be sent at a regular time | Textual Description |
| worker | worker | List of Actors |
| 1. The amount of the specific products reached the minimum 2. The chosen supplier exist in the system | 1. The chosen supplier exist in the system | Pre-conditions |
| 1. A Order instance o will be created 2. A Product instances will be created when the amount reached the minimum 3. Each product p was associated with the current order 4. The order o gets an amount for each product p 5. The current order was associated with the Store(to add it to the historical orders) | 1. A Order instance o was created 2. A Product instances were created by demand 3. Each product p was associated with the current order 4. The order o gets an amount for each product p 5. The current order was associated with the Store (to add it to the historical orders) | Post-conditions |
| 1. Worker chooses a supplier 2. The worker create an order 3. The worker add the products and specify amount to each product 4. Worker send/save the order | 1. Worker chooses supplier 2. Worker creates a circulating order 3. Worker adds product and amounts to the order 4. The worker choose delivery days 5. Worker send/save the order | Main success scenario |
|  |  | Alternatives/Extensions |

Use case e:

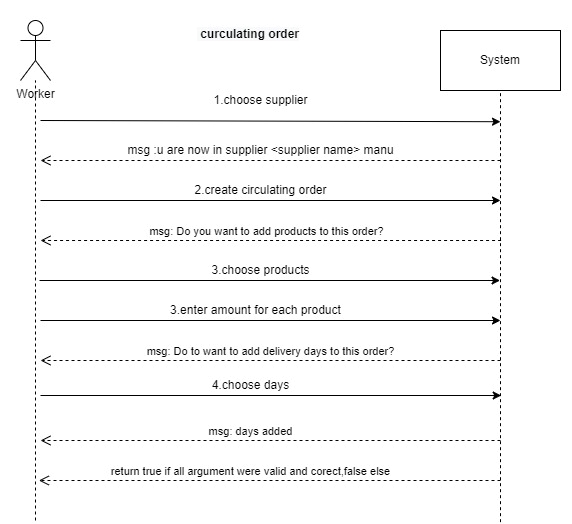
* name: create circulating order
* Textual Description: Create an order that will be sent in a fixed days
* List of Actors: Worker
* Pre-conditions: The chosen supplier exist in the system
* Post-conditions:
  1. A required supplier is chosen
  2. An Order instance o was created
  3. A Product instances were created by demand
  4. Each product p was associated with the current order
  5. The order o gets amount for each product p
  6. The current order o was associated with the Store (to add it to the historical orders)
* Main success scenario:
  1. Worker chooses supplier
  2. Worker creates a circulating order
  3. Worker adds product and amounts to the order
  4. Worker send/save the order
* Alternatives/Extensions:

A\* in any point the: system fails- need to recovery the system.

1. the worker choose supplier that isn't exist: the system signals the error to the worker

2. the worker choose the same product multiple times: the system signals the error to the worker

3. the worker choose the same day twice or invalid day (i.e. 8,9, -1): the system signals the error to the worker



Use case f:

* Name: create order due to lack
* Textual Description: Creating an order that will be sent when there is a lack of particular product
* List of Actors: worker, cashier
* Pre-conditions:
  1. The worker specific min amount to product
  2. The worker specify amount to refill
  3. The amount of the specific products reached to the minimum
  4. The chosen supplier exists in the system
* Post-conditions: (after the trigger is triggered)

1. An Order instance o will be created
2. A Product instances will be created
3. Each product p was associated with the current order o
4. The order o gets an amount product p
5. The current order was associated with the Store (to add it to the historical orders)

* Main success scenario: (When there is a lack of this product)

1. The system selects the supplier with the cheapest‏ price on the product
2. The system creates a new order to this supplier
3. The systemaddstheproduct and amount to the order
4. The system send\save the order

* Alternatives/Extensions:

A\* in any point the: system fails- need to recovery the system.

1. the worker choose product that isn't exist: the system signals the error to the worker

2. the worker specify invalid amount to refill: the system signals the error to the worker

3. there is no supplier which supply the specific product: the system signals the error to the worker

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי