

# Runtime Terror

## **Project Vision:**

This software will benefit the user because it will allow for easy management of an online store's inventory, allowing the store to add/remove items, keep track of the total inventory, keep track of orders. The system will manage the store's inventory allowing the employees to focus on other tasks. The main stakeholders for this project are the customers of the store and the store's employees. The customers will only be able to purchase the items if they are in stock, and the employees will need to know if stock is getting low so that they can order more. The system will have a GUI that allows for the user to look up an item by an ID number and see its stock, a general description, and make a purchase. The software will also need to keep track of the items that the customer has added to their cart, to enable the purchase of multiple items.

## **Website Link w/ Link to GitHub and Issue Tracking:**

<https://runtimeerrorbu.github.io/InventoryManagement/>

## **Group Members:**

Ashley Bickham: 13 hours worked

Joshua Hunter (Project Leader): 12 hours worked

Austin Lehman: 22 hours worked

Tyler Ross: 11 hours worked

CSI 3471  
Ashley Bickham, Joshua Hunter, Austin Lehman, Tyler Ross

**SIMPLE GANTT CHART** by Vertex42.com  
<https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html>

Mon, 2/1/2021	
1	

[illegible]

CSI 3471  
Ashley Bickham, Joshuas Hunter, Austin Lehman, Tyler Ross

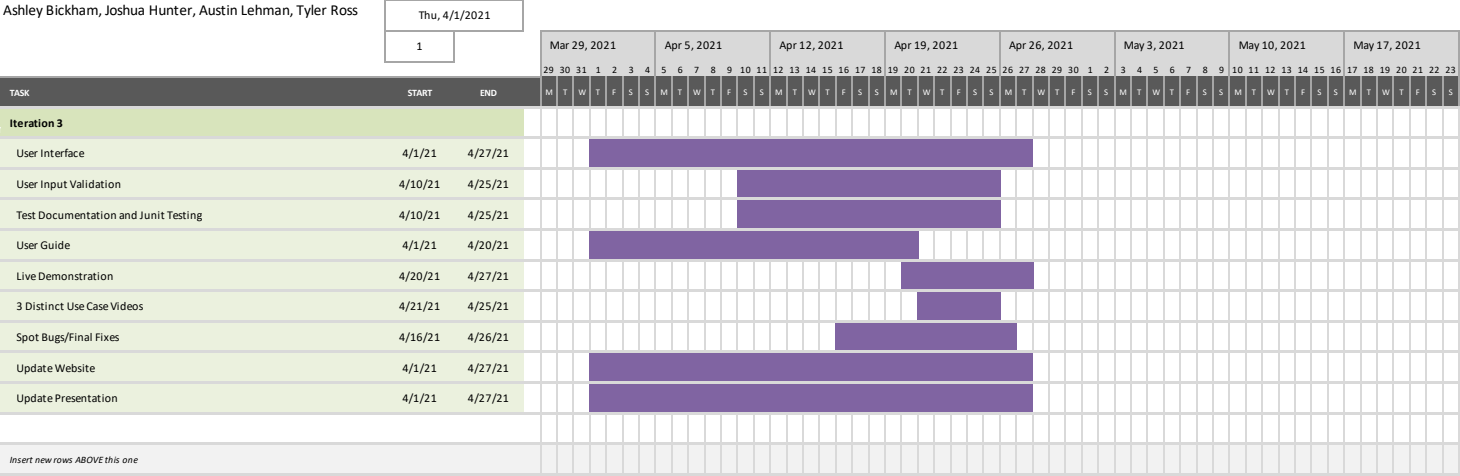
SIMPLE GANTT CHART by Vertex42.com  
<https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html>

Ashley Bickham, Joshua Hunter, Austin Lehman, Tyler Ross			Wed, 3/3/2021																																																							
1			Mar 1, 2021							Mar 8, 2021							Mar 15, 2021							Mar 22, 2021							Mar 29, 2021							Apr 5, 2021							Apr 12, 2021							Apr 19, 2021						
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
TASK			START		END																																																					
Iteration 2																																																										
Demo of User Interface			3/3/21		4/1/21																																																					
Design Class Diagram			3/3/21		3/15/21																																																					
Design Sequence/Communication Diagrams			3/10/21		3/28/21																																																					
GRASP			3/15/21		3/25/21																																																					
Update Website			3/5/21		4/1/21																																																					
Update Powerpoint			3/5/21		4/1/21																																																					
Insert new rows ABOVE this one																																																										

Runtime Terror Gantt Chart

CSI 3471  
Ashley Bickham, Joshua Hunter, Austin Lehman, Tyler Ross

SIMPLE GANTT CHART by Vertex42.com  
<https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html>



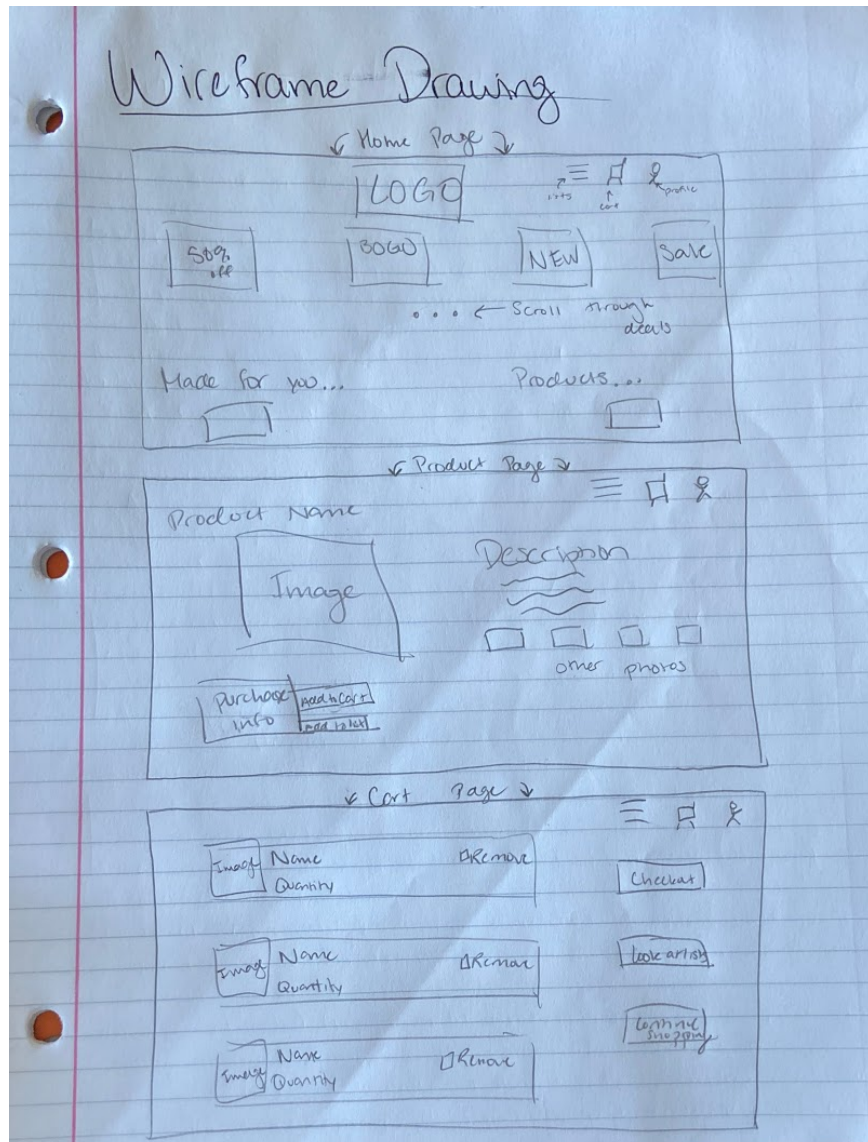
## **Requirements List:**

- R1:** The system will keep a set of items that are sold by the store
- R2:** The store will be able to add and remove from the set of items
- R3:** Items will be associated with a unique ID to allow for easy lookup and management
- R4:** The system will keep track of the amount of each item that is available for sale
- R5:** Customers will be able to add and remove items they wish to purchase into a cart
- R6:** The system will be able to keep track of each order by its unique ID
- R7:** Each order will be stored in a database to allow for easy lookup
- R8:** Customers must be able to make accounts easily
- R9:** When stock is low, the system must notify an employee or manager
- R10:** Customers can rent items that allow it.
- R11:** Customers can return items that have been rented.
- R12:** If a customer's rental is late a fee should be sent to them.
- R13:** The customer should be able to return an item within 30 days.
- R14:** The customer should be able to search for items.
- R15:** The customer should be able to add items to a wishlist.
- R16:** The system should notify the customer when an item in their wishlist is in stock.
- R17:** The system should notify the customer when an item in their wishlist is on sale.

## **Use Cases:**

1. Make Purchase – Ashley
2. Add Item to Cart – Austin
3. Add Item to Wishlist – Tyler
4. Restock Item – Joshua
5. Return Item – Ashley
6. Make Account – Austin
7. Send/Ship Item – Tyler
8. Add New Product – Tyler
9. Delete Product – Joshua
10. View Product Availability – Joshua
11. Generate Report – Ashley
12. Remove Item from Cart – Austin

	REQ 1	REQ 2	REQ 3	REQ 4	REQ 5	REQ 6	REQ 7	REQ 8	REQ 9	REQ 10	REQ 11	REQ 12	REQ 13	REQ 14	REQ 15	REQ 16	REQ 17
UC - 1 Make Purchase	✓		✓			✓	✓			✓							
UC - 2 Add Item to Cart	✓		✓	✓	✓		✓			✓				✓		✓	
UC - 3 Add Item to Wishlist	✓		✓	✓			✓			✓				✓	✓	✓	✓
UC - 4 Restock Item	✓		✓	✓			✓		✓								
UC - 5 Return Item	✓		✓			✓	✓				✓	✓	✓				
UC - 6 Make Account								✓									
UC - 7 Ship/Send Item	✓		✓				✓			✓	✓						
UC - 8 Add New Product	✓	✓	✓							✓							
UC - 9 Delete Product	✓	✓	✓														
UC - 10 View Product Availability	✓		✓	✓					✓	✓	✓			✓		✓	✓
UC - 11 Generate Report						✓	✓		✓							✓	
UC - 12 Remove Item from Cart	✓		✓		✓		✓										



**Use Case Name:** Add Item to Cart

**Scope:** Store

**Level:** user goal

**Primary Actor:** customer

**Stakeholders and Interests:**

- Customer: wants to be able to purchase multiple different items
- Company: wants to be able to keep track of the items the customer purchases so that they can get their items in a timely manner

**Preconditions:** Customer wants to add an item to their cart

**Success Guarantee (or Postconditions):**

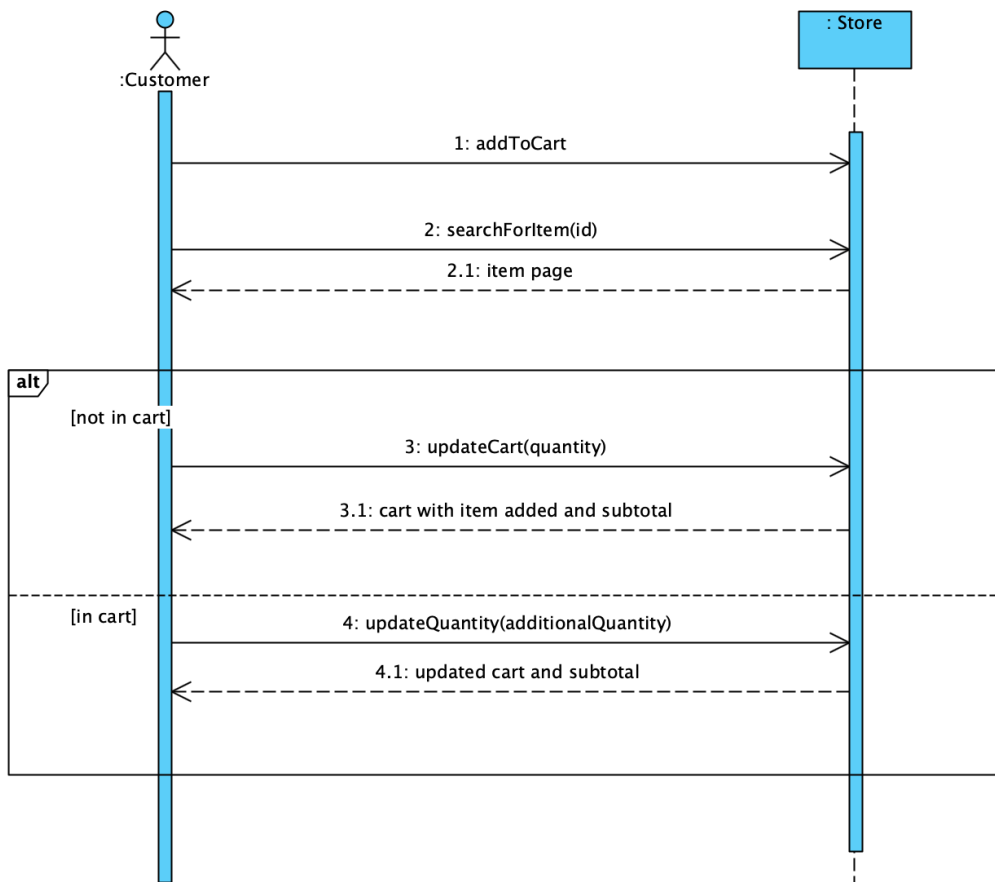
The item is added to the cart, and the subtotal is updated

**Main Success Scenario (or Basic Flow):**

1. Customer navigates to the item
2. Customer clicks add to cart button
3. Customer enters the quantity they wish to purchase
4. Items are added to cart
5. Subtotal is updated
6. The customer is informed that the items have been successfully added to the cart

**Extensions (or Alternative Flows):**

- 2a. Item is already in cart:
  1. Customer selects how many they wish to add to the cart
  2. The customer's selection is added to their previous selection





**Use Case Name:** Remove Item from Cart

**Scope:** Store

**Level:** user goal

**Primary Actor:** customer

**Stakeholders and Interests:**

- Customer: wants to be able to keep track of the items to be purchased, and make adjustments to them (remove, alter number of) if needed.
- Company: wants to be able to properly track an order, including when items are removed, so that purchases are accurate.

**Preconditions:** Customer wants to remove an item from their cart

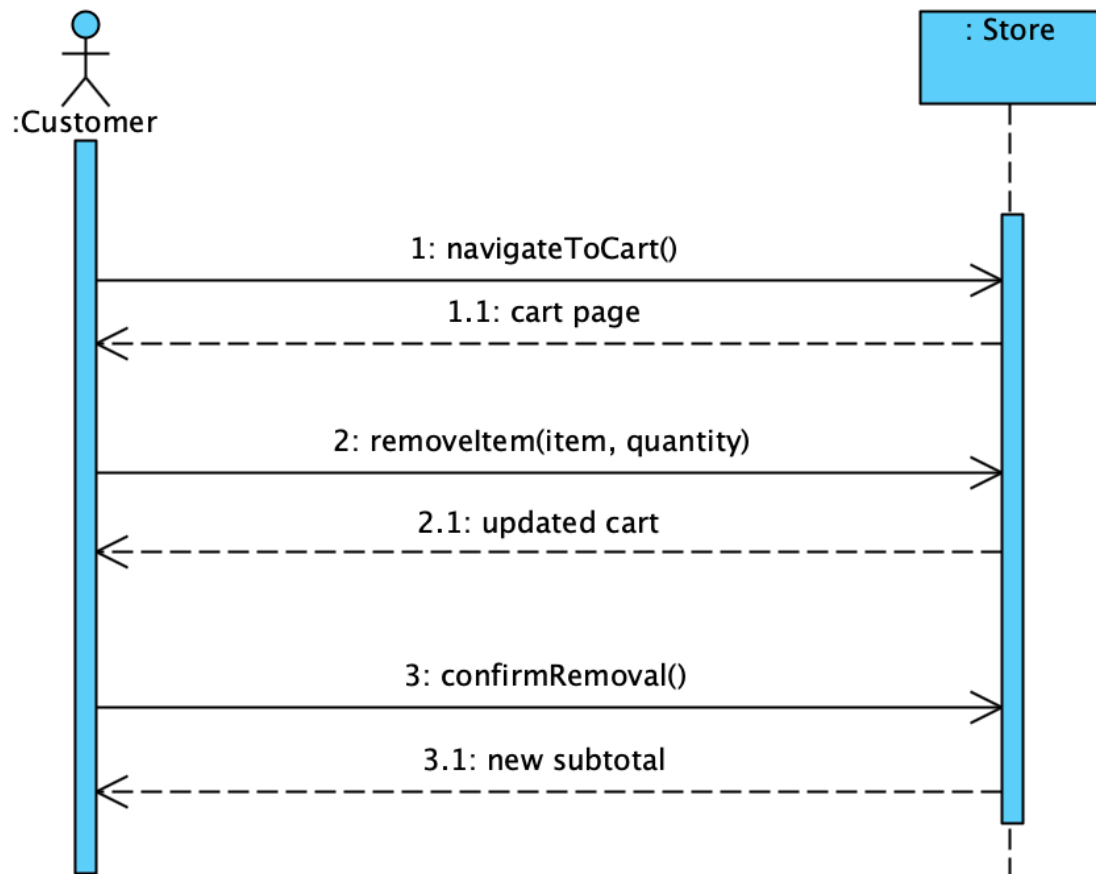
**Success Guarantee (or Postconditions):** The item the customer wanted removed from the cart will be removed and the subtotal calculated.

**Main Success Scenario (or Basic Flow):**

1. Customer navigates to their current shopping cart instance
2. The customer clicks on the item to be removed from the cart
3. The customer clicks on the UPDATE ORDER button
4. The customer clicks remove item
5. The machine shows a verification that the user wants to remove the item
6. The customer selects yes
7. The subtotal is updated

**Extensions (or Alternative Flows):**

- 4a. There are multiple of the same item in the cart
  1. Customer selects how many of the item they would like to remove
  2. Customer confirms the amount to remove
- 6a. The customer doesn't want to remove the item from cart
  1. The customer changes their mind and selects no
- 7a. There is no resulting change in the order
  1. The subtotal is not changed and thus not updated



**Use Case Name:** Make Account

**Scope:** Store

**Level:** user-goal

**Primary Actor:** Customer

**Stakeholders and Interests:**

- Customer: wants to store their information so that it's saved for next time
- Company: has to keep track of the customer's login information

**Preconditions:** Customer is asked if they want to create an account

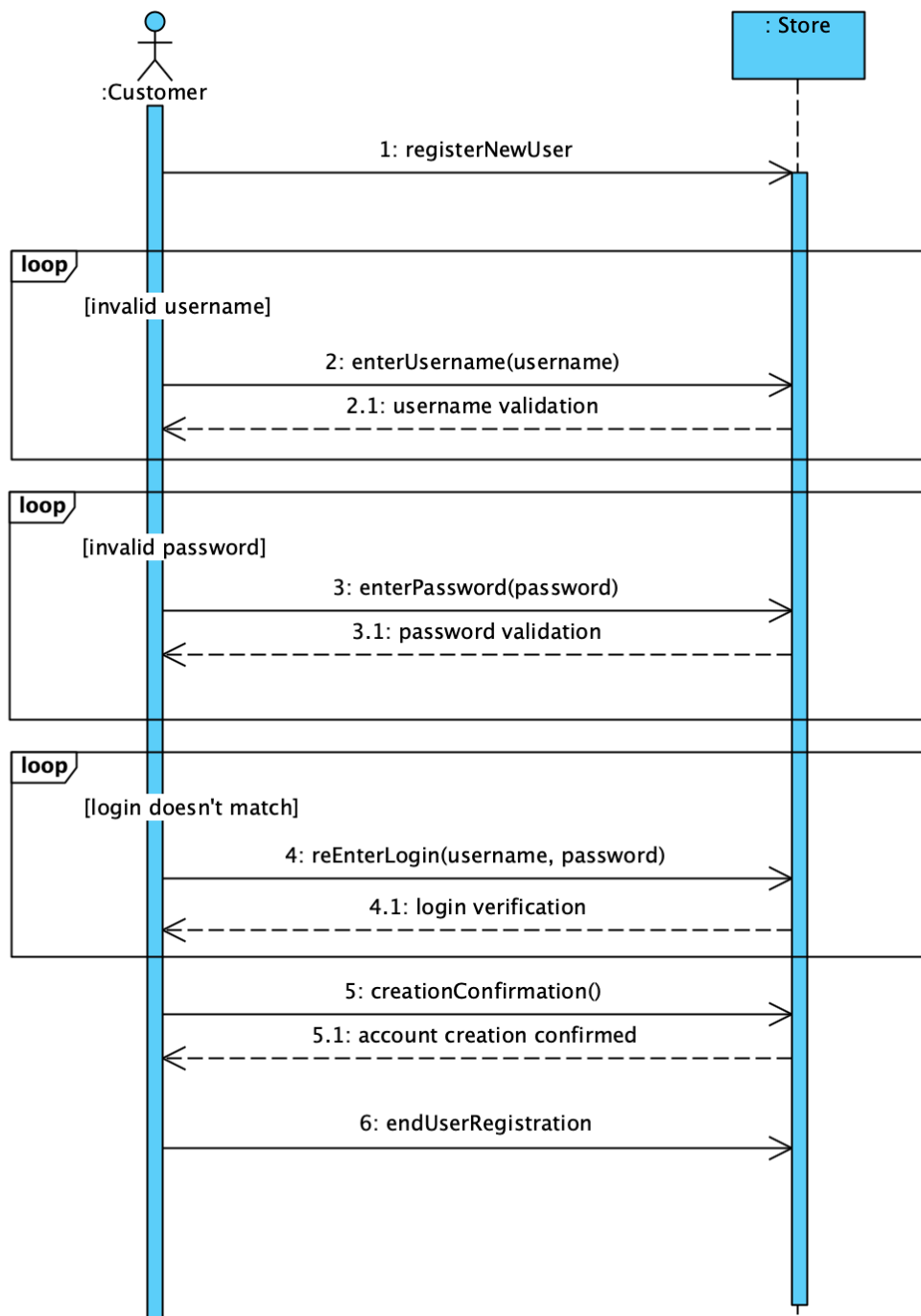
**Success Guarantee (or Postconditions):** The customer's login information is saved and their cart is tied to their account

**Main Success Scenario (or Basic Flow):**

1. Customer selects they want to create an account
2. The customer is prompted for a username
3. The customer is prompted for a password
- Repeat steps 2 and 3 until the login is validated*
4. The user is asked to re-enter their username and password for validation
5. The customer is asked to confirm the account creation

**Extensions (or Alternative Flows):**

- 2a. The username is invalid
  1. The customer is informed why the username is invalid (either it is taken or does not meet requirements)
  2. The customer is asked to enter another username
- 3a. The password is invalid
  1. The customer is informed why the password is invalid (i.e. the requirements it violates)
  2. The customer is asked to enter another password
- 4a. The customer's username does not match
  1. The customer is informed that their usernames don't match
  2. The customer is prompted to try again
- 4b. The customer's password doesn't match
  1. The customer is informed that their passwords don't match
  2. The customer is prompted to try again



**Use Case Name:** View Product Availability

**Scope:** Store

**Level:** Subfunction

**Primary Actor:** Company

**Stakeholders and Interests:**

- Customer: Wants to have the option to pick from a variety of items and have what is needed available.
- Company: Wants to be able to properly track the inventory of all of the products available.

**Preconditions:** The website is up and running and the product being looked for is valid and have an unique id and product page

**Success Guarantee (or Postconditions):** The availability of a product is checked and updated if needed

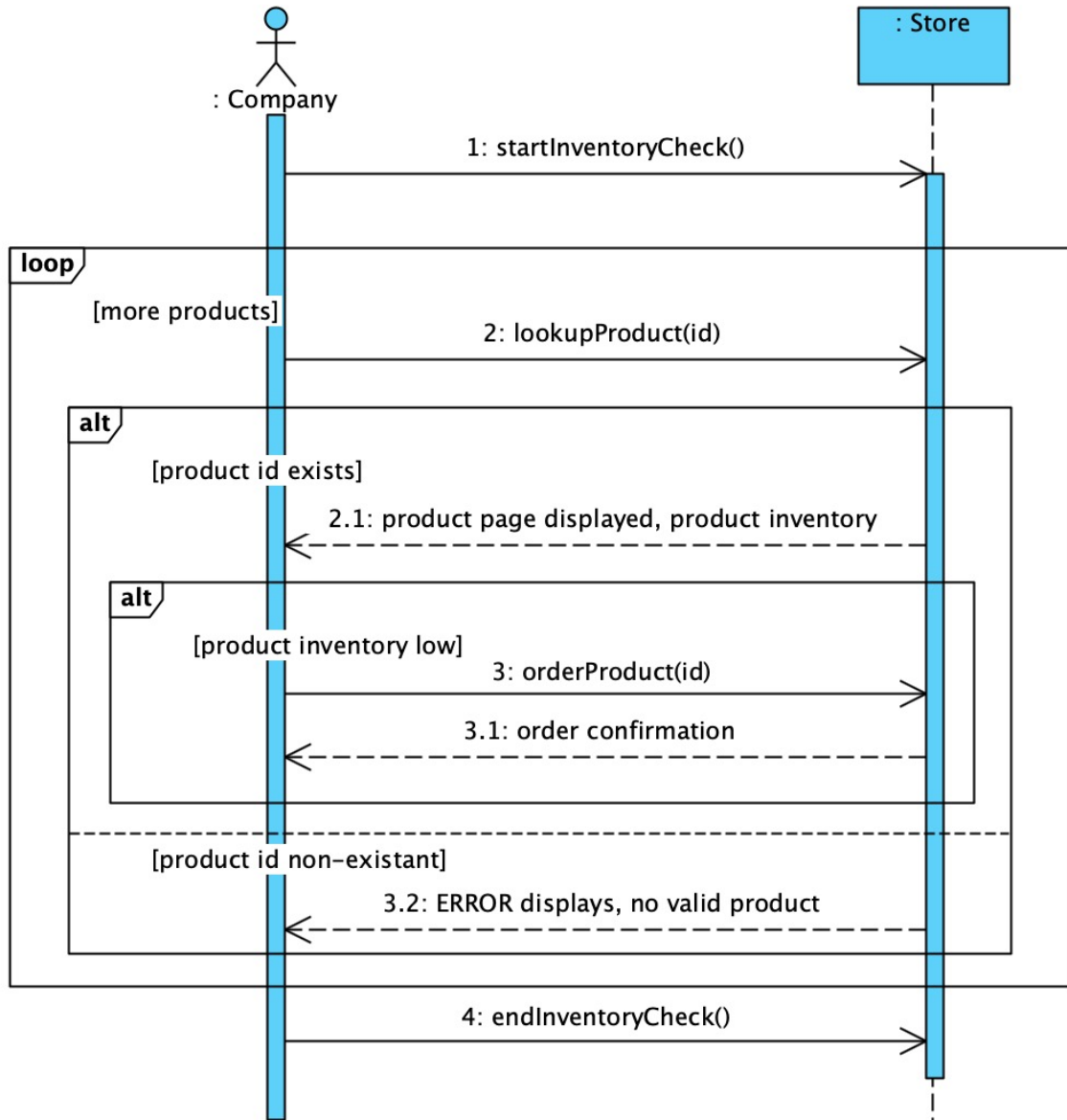
**Main Success Scenario (or Basic Flow):**

1. A product inventory is needed to be checked
2. The product is looked up based on its unique id
3. The inventory available will be displayed on the product page
4. The company will request more inventory to be shipped to the containment center if product is getting low.

*Steps 2-4 can be repeated if multiple products are having their inventory checked*

**Extensions (or Alternative Flows):**

- 2a. The unique id does not exist
  1. The product being looked up doesn't exist and results an ITEM NOT FOUND message



**Use Case Name:** Delete a Product

**Scope:** Store

**Level:** Subfunction

**Primary Actor:** Manager

**Stakeholders and Interests:**

- Customer: Wants to have the option to pick from a variety of items and have what is needed available.
- Manager: Wants to remove product from the store to generate more revenue and make more space for more popular items

**Preconditions:** There is a product the company wants to remove

**Success Guarantee (or Postconditions):** The product is removed from the site

**Main Success Scenario (or Basic Flow):**

1. A product is deemed needed to be removed from the store
2. The product is deleted from the sites database based on its unique id
3. The products id is freed to be used by another future product
4. The products page is removed as result of a freed unique id

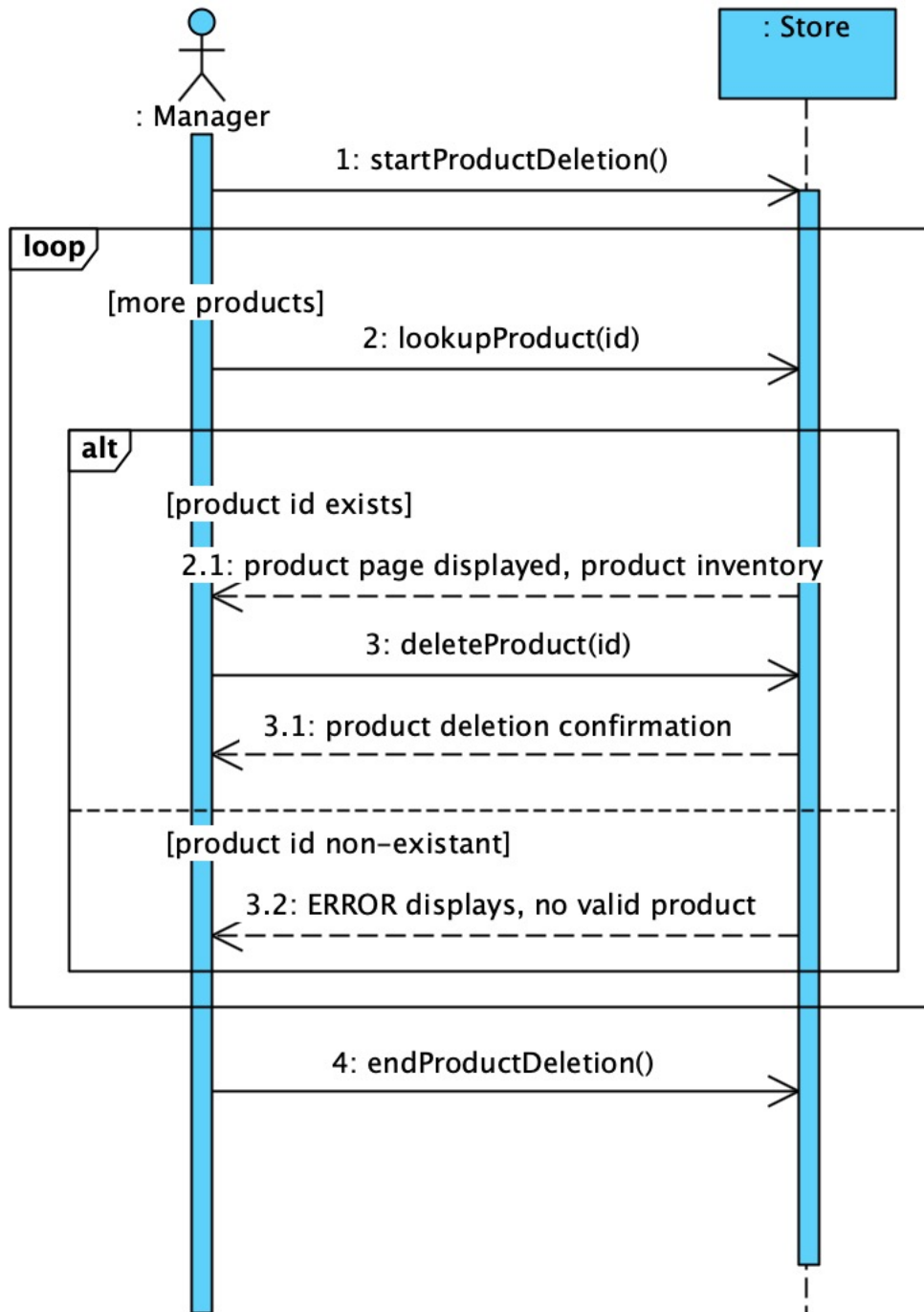
*Steps 2-4 can be repeated if multiple products are being deleted*

**Extensions (or Alternative Flows):**

\*a. At any point the products removal process can stop, resulting in leaving the product on the store

2a. The unique id does not exist

1. The product being looked up doesn't exist and results an ITEM NOT FOUND message





**Use Case Name:** Stock Item

**Scope:** Store

**Level:** subfunction

**Primary Actor:** Company

**Stakeholders and Interests:**

- Company: wants to be able to add their item to the shop so it can be sold
- Customer: wants to be able to purchase the item once it is stocked

**Preconditions:** The item is out of stock

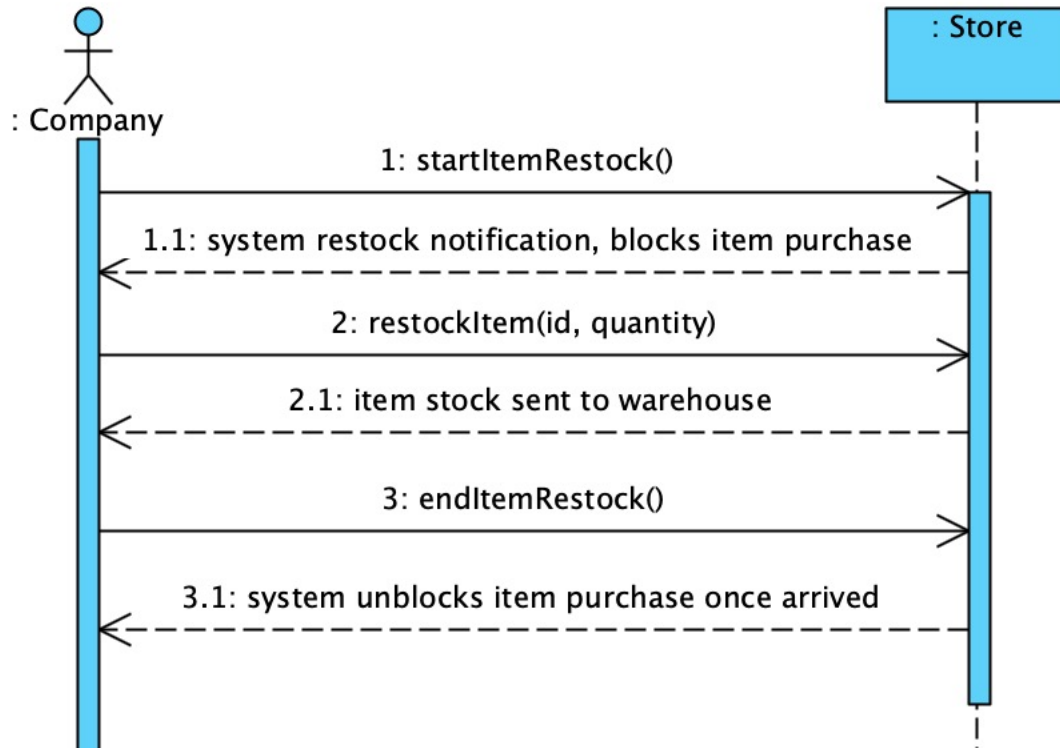
**Success Guarantee (or Postconditions):** The item is stocked to the appropriate level

**Main Success Scenario (or Basic Flow):**

1. The system notifies the company that the item is out of stock
2. The system marks the item as unavailable to purchase
3. The company gets more of the item
4. The company sends the ordered items to the store's warehouse
5. Company updates the amount available in the system
6. System refreshes the item's stock
7. The item is made available for sale again

**Extensions (or Alternative Flows):**

- 3a. Company decides not to add more of the item
  1. The system confirms the choice, and resends the message in a week



**Use Case Name:** Add Item to Wishlist

**Scope:** Customer

**Level:** user-goal

**Primary Actor:** Customer

**Stakeholders and Interests:**

- Customer: Wants to add the product to their wishlist.

**Preconditions:** The product must be in the store.

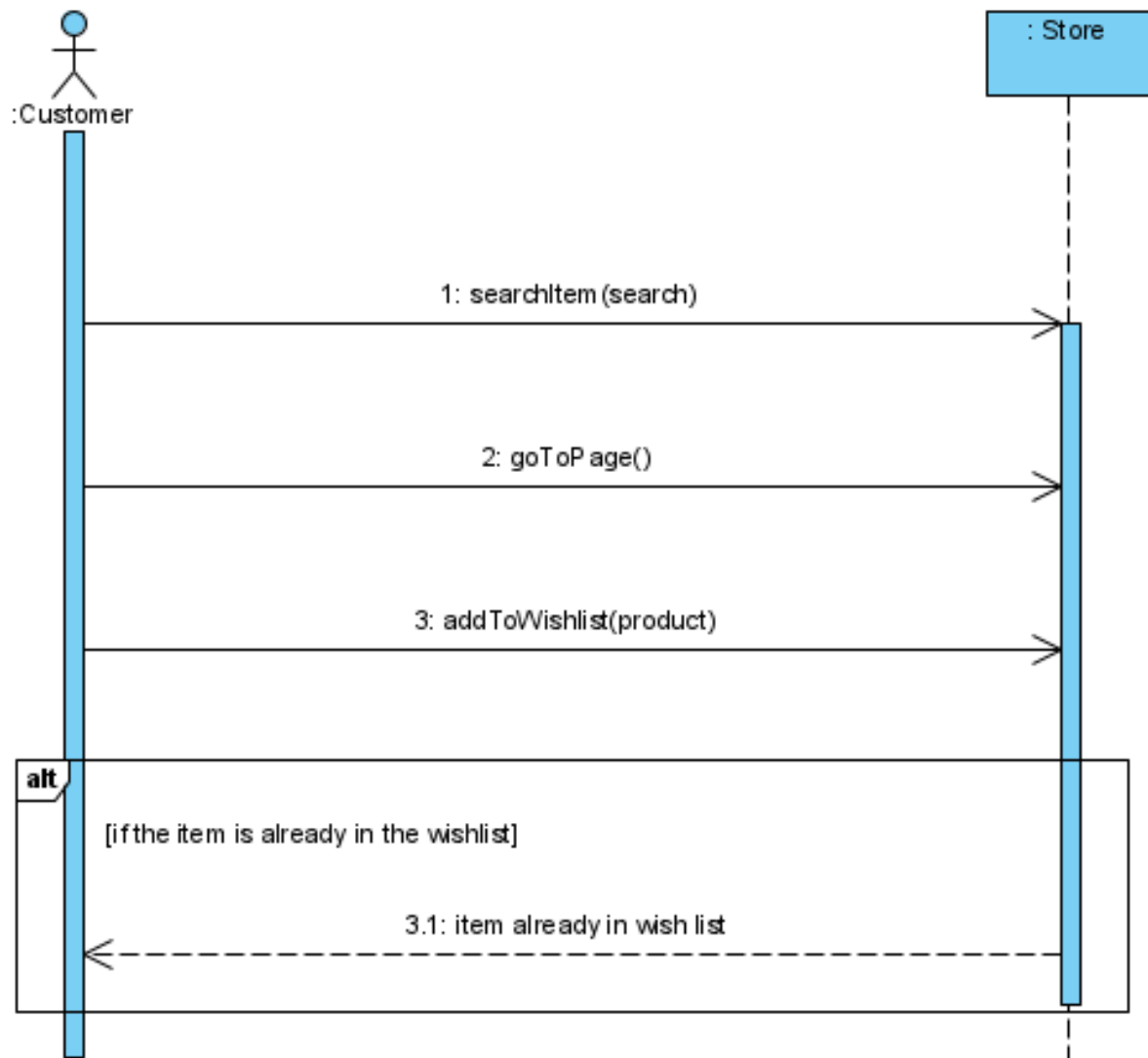
**Success Guarantee (or Postconditions):** The product will be added to the user's wishlist.

**Main Success Scenario (or Basic Flow):**

1. User searches for the product.
2. User clicks on the product page.
3. User clicks the "add to wishlist" button,
4. Product is added to the user's wishlist.

**Extensions (or Alternative Flows):**

- 4a. The product is already in the user's wishlist.
  1. The site tells the user that the item is already in their wishlist.



**Use Case Name:** Ship/Send Item

**Scope:** Store

**Level:** Subfunction

**Primary Actor:** Customer

**Stakeholders and Interests:**

- Company: The company makes profit if the item is successfully delivered and not returned.

- Customer: The customer is the one who ordered the item.

**Preconditions:** An item is ordered by the customer.

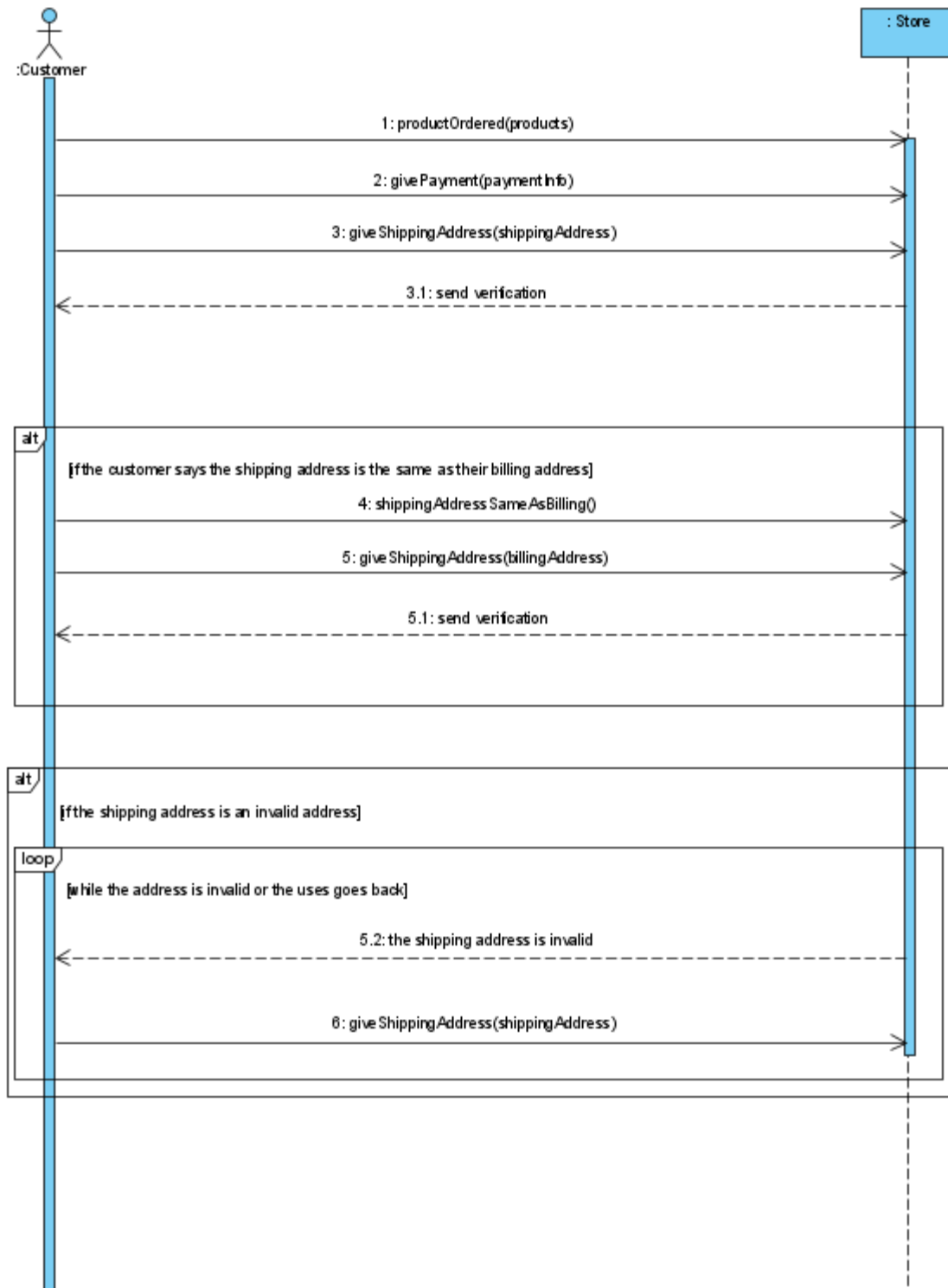
**Success Guarantee (or Postconditions):** The item is delivered to the customer.

**Main Success Scenario (or Basic Flow):**

1. The customer orders a product.
2. The customer inputs their shipping address.
3. The item is paid for.
4. The item is sent to the shipper with the correct address.

**Extensions (or Alternative Flows):**

- 2a. The customer selects that their shipping address is the same as their billing address.
  1. The store will fill in the shipping address with the billing address.
- 2b. The address is not valid.
  1. The site will tell the customer that the address is not valid.
  2. The site will not let them proceed till a valid address is input.



**Use Case Name:** Add New Product

**Scope:** Store

**Level:** Subfunction

**Primary Actor:** Manager

**Stakeholders and Interests:**

- Manager: The one adding the product to the store.
- Customer: The one that will buy the product.
- Shipper: The place that ships an item to the customer.

**Preconditions:** There is a product the company wants to add.

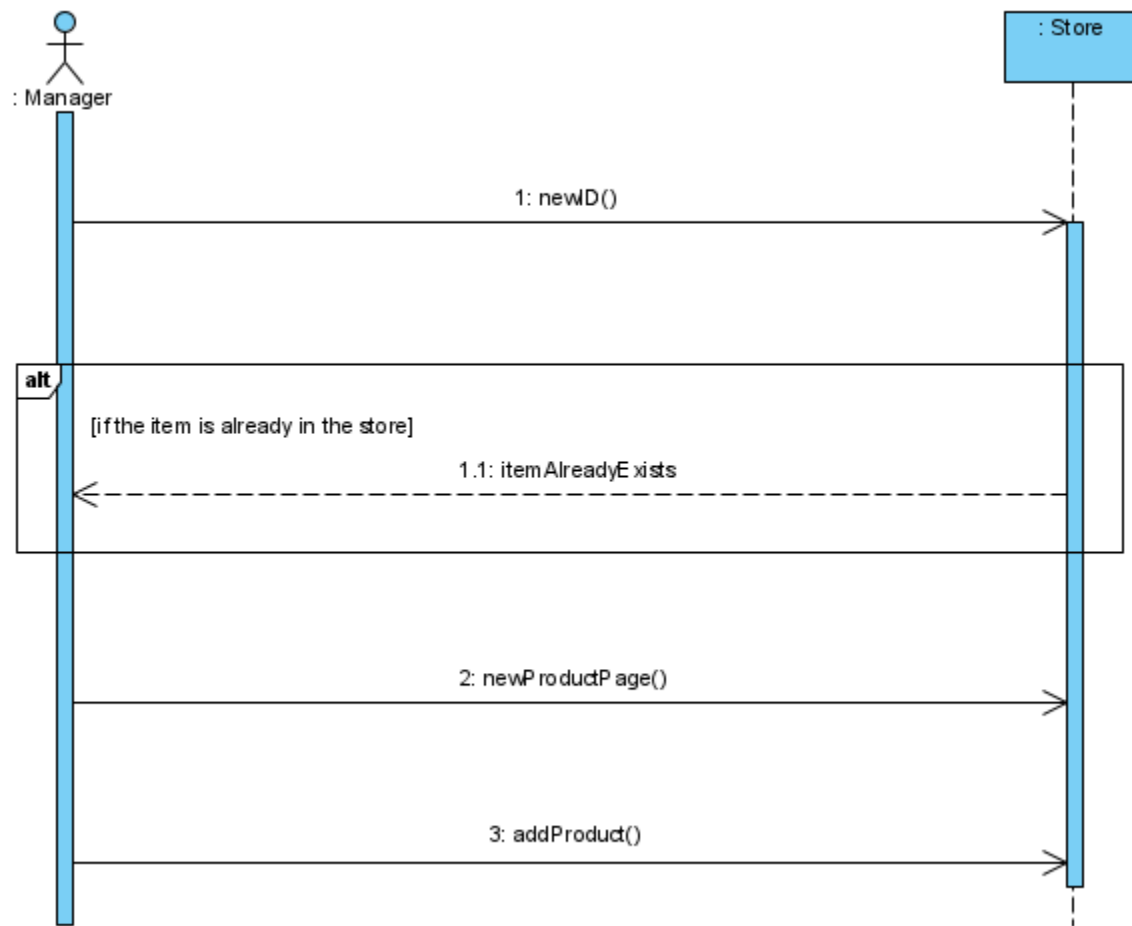
**Success Guarantee (or Postconditions):** The product is added to the site.

**Main Success Scenario (or Basic Flow):**

1. A product is chosen to be added to the store.
2. The product is given a unique id.
3. The product is given a product page based off of a template.
4. The product is added to the store.

**Extensions (or Alternative Flows):**

- 1a. The product is already in the store.
  1. The system will tell the company that the product already exists.





**Use Case Name:** Make Purchase

**Scope:** Store

**Level:** user goal

**Primary Actor:** Customer

**Stakeholders and Interests:**

- Customer: Wants to buy product
- Company: When an item is purchased, revenue goes back to the owner

**Preconditions:** The item wanted by the customer must be in stock

**Success Guarantee (or Postconditions):**

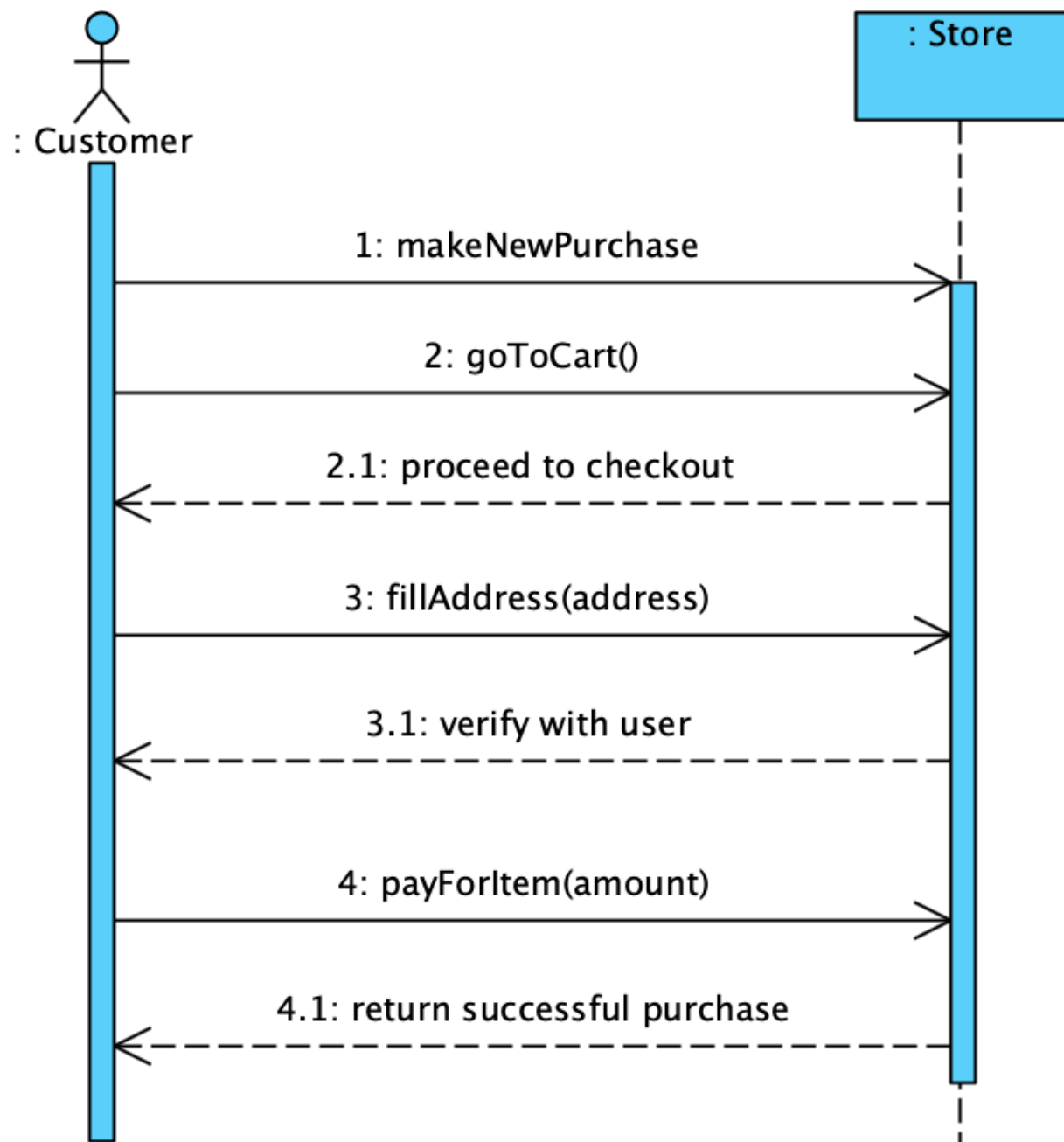
The customer is successfully able to purchase the product they wanted

**Main Success Scenario (or Basic Flow):**

1. Customer searches for an item from the store
2. Customer picks out that item from the store
3. Item goes into the Customers' cart
4. Customer goes to their cart
5. Customer checks out and item is purchased

**Extensions (or Alternative Flows):**

- 1a. The item isn't sold from the store
  1. Customer must request item for the future
  2. Customer must find either an off-brand replacement or another store that sells the product
- 2a. The item is out of stock
  1. Customer can request to be notified when the item is back in stock



**Use Case Name:** Return Item

**Scope:** Store

**Level:** user-goal

**Primary Actor:** Customer

**Stakeholders and Interests:**

- Customer: Is able to return their product worry free
- Company: Receives product back from customer

**Preconditions:** Item has been received by the customer in order to return it

**Success Guarantee (or Postconditions):** The item is successfully returned to the company

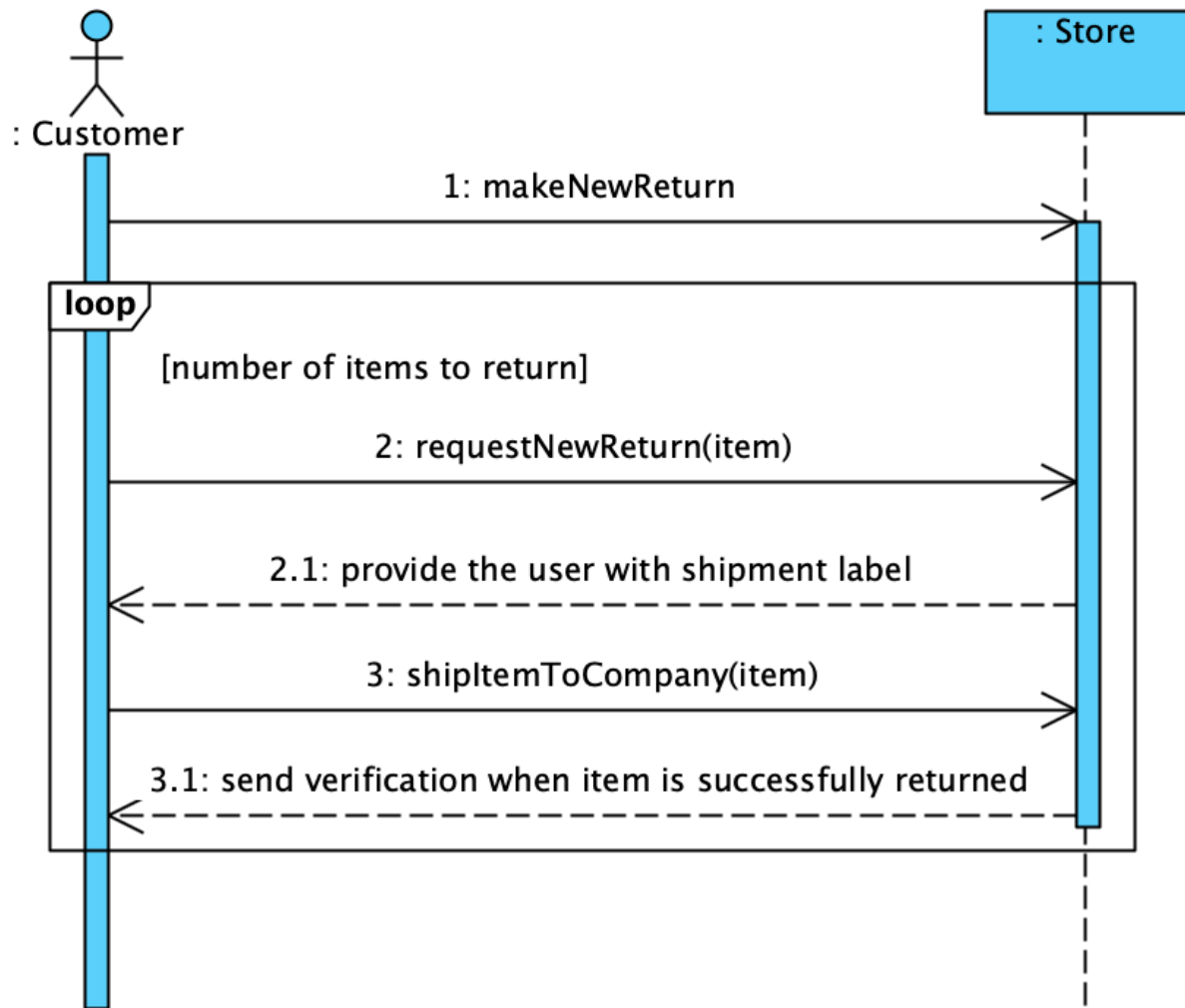
**Main Success Scenario (or Basic Flow):**

1. The customer wishes to return an order
2. The customer requests a return
3. The customer receives a shipment label to use to return the item
4. The customer uses the shipment label to ship the item through a separate shipping company (FedEx, UPS, USPS)
5. The company receives the item returned

*Steps 1-5 can be repeated due to how many items they wish to return*

**Extensions (or Alternative Flows):**

- 2a. The customer is not qualified to return the item
  1. Customer must keep the item, donate the item, or throw it away
- 3a. The shipment label is wrong
  1. Customer must request a new one
- 4a. The shipment label is expired
  1. The customer must keep the item, donate the item, or throw it away
- 5a. The item never returns to the company
  1. The company must cut their losses due to lost package



**Use Case Name:** Generate Report

**Scope:** Store

**Level:** user goal

**Primary Actor:** Company

**Stakeholders and Interests:**

- Company: Wants to visualize their product sales and customer satisfaction/customer usage rate

**Preconditions:** The website is up and running

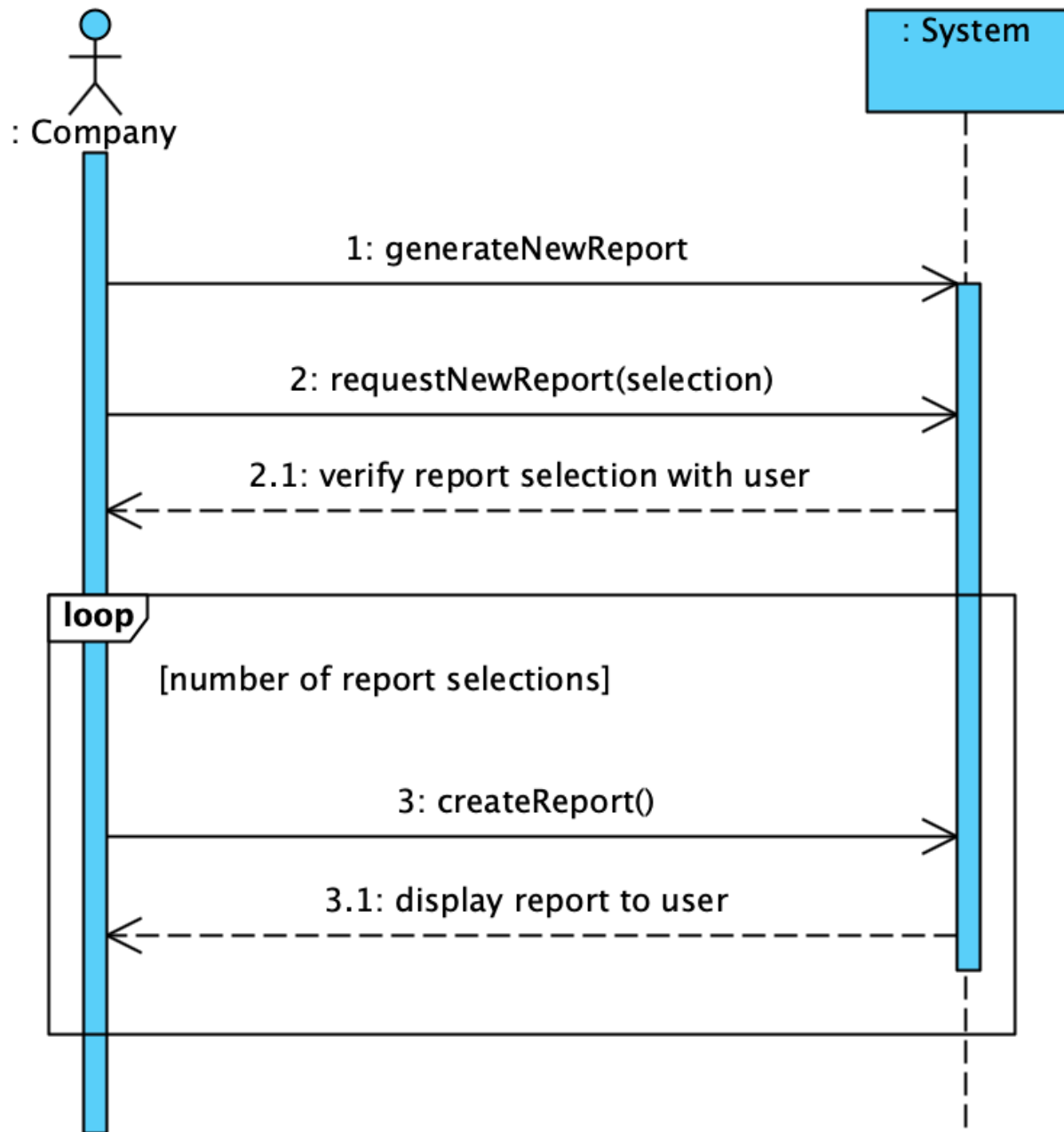
**Success Guarantee (or Postconditions):** The website produces a report that displays the companys' weekly sales, new customer accounts, and ratings

**Main Success Scenario (or Basic Flow):**

1. The company requests report
  1. The company selects certain aspects to receive a report of
2. The system confirms the selection
3. The report is generated through the system
4. The report is sent to the company

**Extensions (or Alternative Flows):**

- The system does not respond
  - The system must be restarted manually in order to generate report



## Store System

addToCart  
searchForItem(id)  
updateCart(quantity)  
updateQuantity(additionalQuantity)  
navigateToCart()  
removeItem(item, quantity)  
confirmRemoval()  
registerNewUser  
enterUsername(username)  
enterPassword(password)  
reEnterLogin(username, password)  
creationConfirmation()  
endUserRegistration  
startInventoryCheck()  
orderProduct(id)  
endInventoryCheck()  
startProductDeletion()  
deleteProduct(id)  
endProductDeletion()  
startRestockItem()  
itemRestock(id, quantity)  
endRestockItem()  
goToPage()  
addToWishList(product)  
givePayment(paymentInfo)  
giveShippingAddress(shippingAddress)  
shippingAddressSameAsBilling()  
productOrdered(products)  
newProductPage()  
newID()  
addProduct()  
makeNewPurchase  
goToCart()  
fillAddress(address)  
payForItem(amount)  
makeNewReturn  
requestNewReturn(item)  
shipItemToCompany(item)  
generateNewReport()  
requestNewReport(selection)  
createReport()

<b>Operation:</b>	addToCart
<b>Cross References:</b>	Add Item to Cart
<b>Preconditions:</b>	There is an instance of a cart created
<b>Postconditions:</b>	The cart has an item added to it
<b>Operation:</b>	searchForItem(id)
<b>Cross References:</b>	Add Item to Cart
<b>Preconditions:</b>	The customer knows what item they are searching for
<b>Postconditions:</b>	The customer has navigated to the item page
<b>Operation:</b>	updateCart(quantity)
<b>Cross References:</b>	Add Item to Cart
<b>Preconditions:</b>	The customer has navigated to the item page, and the item isn't in the cart
<b>Postconditions:</b>	The item and its desired quantity is added to the cart
<b>Operation:</b>	updateQuantity(additionalQuantity)
<b>Cross References:</b>	Add Item to Cart
<b>Preconditions:</b>	The customer has navigated to the item page, and the item is already in the cart
<b>Postconditions:</b>	The quantity of the item in the cart is increased by additionalQuantity
<b>Operation:</b>	navigateToCart()
<b>Cross References:</b>	Remove Item from Cart
<b>Preconditions:</b>	The cart exists
<b>Postconditions:</b>	The cart is displayed to the user
<b>Operation:</b>	removeItem(item, quantity)
<b>Cross References:</b>	Remove Item from Cart
<b>Preconditions:</b>	there are at least "quantity" instances of "item" in the cart
<b>Postconditions:</b>	the item is
<b>Operation:</b>	confirmRemoval()
<b>Cross References:</b>	Remove Item from Cart
<b>Preconditions:</b>	The customer has selected to remove an item from their cart
<b>Postconditions:</b>	If confirmed, the removal is finalized Else, the removal is cancelled
<b>Operation:</b>	registerNewUser
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The customer has elected to create an account.
<b>Postconditions:</b>	A new account is registered.
<b>Operation:</b>	enterUsername(username)
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The customer is prompted to enter a username.
<b>Postconditions:</b>	The username is validated.



<b>Operation:</b>	enterPassword(password)
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The customer is prompted to enter a password.
<b>Postconditions:</b>	The password is validated.
<b>Operation:</b>	reEnterLogin(username, password)
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The customer is prompted to re-enter their valid username and password
<b>Postconditions:</b>	The two logins are compared, then validated if they match
<b>Operation:</b>	creationConfirmation()
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The two logins matched
<b>Postconditions:</b>	If the user confirms, the account is created Else, account creation is canceled
<b>Operation:</b>	endUserRegistration
<b>Cross References:</b>	Make Account
<b>Preconditions:</b>	The registration was sucessful
<b>Postconditions:</b>	The process of creating an account ends
<b>Operation:</b>	startInventoryCheck()
<b>Cross References:</b>	View Product availability
<b>Preconditions:</b>	Website is up and running and has products with different product ids
<b>Postconditions:</b>	The inventory checking has been started
<b>Operation:</b>	orderProduct(id)
<b>Cross References:</b>	View Product availability
<b>Preconditions:</b>	Product exists in the database and is coupled with a unique id
<b>Postconditions:</b>	Product is ordered to wearhouse
<b>Operation:</b>	endInventoryCheck()
<b>Cross References:</b>	View Product availability
<b>Preconditions:</b>	All products need to be inventory checked have been
<b>Postconditions:</b>	The inventory checking has been finished
<b>Operation:</b>	startProductDeletion()
<b>Cross References:</b>	Product Deletion
<b>Preconditions:</b>	Website is up and running and has products with different product ids
<b>Postconditions:</b>	The product deletion has been started
<b>Operation:</b>	deleteProduct(id)
<b>Cross References:</b>	Product Deletion
<b>Preconditions:</b>	Product exists in the database and is coupled with a unique id
<b>Postconditions:</b>	Product is deleted from database

<b>Operation:</b>	endProductDeletion()
<b>Cross References:</b>	Product Deletion
<b>Preconditions:</b>	All products needed to be deleted have been
<b>Postconditions:</b>	The inventory deletion has been finished
<b>Operation:</b>	startRestockItem()
<b>Cross References:</b>	Restock Item
<b>Preconditions:</b>	Website is up and running and has products with different product ids
<b>Postconditions:</b>	Restock has been started
<b>Operation:</b>	itemRestock(id, quantity)
<b>Cross References:</b>	Restock Item
<b>Preconditions:</b>	Product exists in the database and is coupled with a unique id
<b>Postconditions:</b>	Product is ordered to warehouse in the specific quantity mentioned
<b>Operation:</b>	endRestockItem()
<b>Cross References:</b>	Restock Item
<b>Preconditions:</b>	All products needed to be restocked have been
<b>Postconditions:</b>	Restock has been finished
<b>Operation:</b>	goToPage()
<b>Cross References:</b>	Add Item To WishList
<b>Preconditions:</b>	The item page exists
<b>Postconditions:</b>	The customer has the item page on their screen
<b>Operation:</b>	addToWishList(product)
<b>Cross References:</b>	Add Item To Wishlist
<b>Preconditions:</b>	The customer has an account
<b>Postconditions:</b>	The item is in the customer's wishlist
<b>Operation:</b>	givePayment(paymentInfo)
<b>Cross References:</b>	Ship/Send Item
<b>Preconditions:</b>	The customer is buy an item
<b>Postconditions:</b>	The store has the customer's payment info
<b>Operation:</b>	giveShippingAddress(shippingAddress)
<b>Cross References:</b>	Ship/Send Item
<b>Preconditions:</b>	The customer bought an item
<b>Postconditions:</b>	The store has the customer's shipping address
<b>Operation:</b>	shippingAddressSameAsBilling()
<b>Cross References:</b>	Ship/Send Item
<b>Preconditions:</b>	The customer is buying an item
<b>Postconditions:</b>	The store has the customer's shipping address

<b>Operation:</b>	productOrdered(products)
<b>Cross References:</b>	Ship/Send Item
<b>Preconditions:</b>	The customer is buying an item
<b>Postconditions:</b>	The item is purchased
<b>Operation:</b>	newProductPage()
<b>Cross References:</b>	Add New Product
<b>Preconditions:</b>	There is a new product to add
<b>Postconditions:</b>	A new product page is created
<b>Operation:</b>	newID()
<b>Cross References:</b>	Add New Product
<b>Preconditions:</b>	There is a new product
<b>Postconditions:</b>	The new product has its own unique ID
<b>Operation:</b>	addProduct()
<b>Cross References:</b>	Add New Product
<b>Preconditions:</b>	There is a fully ready new product to add
<b>Postconditions:</b>	The new product is added to the store
<b>Operation:</b>	makeNewPurchase
<b>Cross References:</b>	Make New Purchase
<b>Preconditions:</b>	There is an instance of a purchase created
<b>Postconditions:</b>	The item/items are purchased
<b>Operation:</b>	goToCart
<b>Cross References:</b>	Make New Purchase
<b>Preconditions:</b>	The cart exists
<b>Postconditions:</b>	The customer is able to open the cart and see the items inside
<b>Operation:</b>	fillAddress(address)
<b>Cross References:</b>	Make New Purchase
<b>Preconditions:</b>	The customer has an address in which they live at
<b>Postconditions:</b>	The customer has an address on file to ship to in the future and for the current purchase
<b>Operation:</b>	payForItem(amount)
<b>Cross References:</b>	Make New Purchase
<b>Preconditions:</b>	Customer has items to pay for
<b>Postconditions:</b>	The customer pays for their items and the purchase is complete
<b>Operation:</b>	makeNewReturn
<b>Cross References:</b>	Make New Return
<b>Preconditions:</b>	An item needs to be returned
<b>Postconditions:</b>	The item is successfully returned

**Operation:** requestNewReturn(item)  
**Cross References:** Make New Return  
**Preconditions:** The customer needs to return an item  
**Postconditions:** The company approves the return request

**Operation:** shipItemToCompany(item)  
**Cross References:** Make New Return  
**Preconditions:** The customer has a ready shipment label to return the item  
**Postconditions:** The company receives the returned item

**Operation:** generateNewReport()  
**Cross References:** Generate Report  
**Preconditions:** The report does not exist  
**Postconditions:** The report is created

**Operation:** requestNewReport(selection)  
**Cross References:** Generate Report  
**Preconditions:** The report selection hasn't been made  
**Postconditions:** The report selection is chosen and verified by the system with the user

**Operation:** createReport()  
**Cross References:** Generate Report  
**Preconditions:** The report has yet to be made  
**Postconditions:** The report is created and sent to the company

