Use Cases

System use cases

Use Case I.1 - Market activation

A user activates the market system.
Actors: user (who initiates the system, isn't necessarily an admin) - see vocabulary.
Precondition: system must be off.
Parameters: there aren't any.
Postcondition: system must be on, and adheres to all the rules of integrity. The data set in the configuration files (see vocabulary) must upload to the system. In case that the updateDatabase flag (see vocabulary) is on, the system will also the data stored in the DB.
Main Scenario:

1. user: runs the system.
2. system: extract all the predefined data regarding the system from the configuration file, along with: the system admin, stores, products, users and their settings.
3. system: opens the market, with all the uploaded data and keeping all the integrity rules.
4. system: responses a success to the user.
5. system: start serving users.

Alternative Scenarios:

- System fails preparation, the system cancels the action and waits for System Manager action.

Use Case I.2 - change/switch/add an external service

A user changes/switches/adds an external service
Actors: user.
Precondition: system must be on.
Parameters: details of the new service (payment or stock)
Postcondition: the system must continue to work the same as before the change of the service
Main Scenario:

1. user: chooses the service he wants to edit/add/switch

2. user: enters the details of the service
3. system: authenticates the details the user entered
4. system: changes/add/switches the service for the new one
5. system: send a positive response to the user

Alternative Scenarios:

- User's initial authentication details are not verified successfully (4), the system asks for the details again.
- service details are incorrect and requested to enter again.

Use Case I.3 - Using the services of an external payment system

the system contacts a payment system that the market is familiar with and tries to perform payment and receive payment validation

Actors: System

Precondition: the system is activated

Postcondition: the system must continue to work the same as before the change of the service

Main Scenario:

1. the system contacts the external payment system with transaction details to perform payment
2. the external payment sends back a positive validation for the transaction the system tried to perform

Alternative Scenarios:

- the payment service failed to make the payment. The system cancels the action and notifies about the cancellation.

Use Case I.4 - Using the services of an external supply service

The system contacts a supply system which the market is familiar with and asks for approval of a valid supply.

Actors: System

Precondition: System is activated

Parameters: details of the supply

Postcondition: the system must continue to work the same as before the request of service

Main Scenario:

1. system: contacts an external supply system it is familiar with and sending it the details of the required package and client information
2. external service: sends back a positive answer that the request was approved

Alternative Scenarios:

- The external service sends back a negative answer (2) that the request was declined, and the reason for the decline of the request.

Use Case I.5 - Realtime notifications

Use Case I.5.a - Store owner (or manager with proper permissions)notifications:

Actors: A member - store owner or member with proper permissions.

Precondition: System must be active.

PostCondition: the system displays to the store owner the notification in real time if he's logged in, or display him the notification when he logs in.

Main Scenario:

1. One of the following occurs:
    - A product in one of the store owner's stores is bought.
    - One of the store owner's stores is being opened.
    - One of the store owner's stores is being closed.
    - One of the store appointments as a store owner is being removed.
    - Product at the store was reviewed by member.
    - A bid proposition was made by a member.
2. The system notifies the store owner about the event that occured.

Alternatives:

- The member isn't logged in (2) in which case the notification will be stored in the queued notifications set of the store owner and will be shown to the store owner when logging in.

Use Case I.5.b - Store owner notifications:

Actors: A member.

Precondition: System must be active, and the member is logged in.

Main Scenario:

1. The member receives a message.
2. The system notifies the store owner about the event that occured.

Alternatives:

- The store owner isn't logged in (2) in which case the notification will be stored in the queued notifications set of the store owner and will be shown to the store owner when logging in.

Use Case I.6 - User notifications show up when logging in

A member logs in to the system, and the notifications that were meant for them during the time they were logged off are shown.
Actors:. Member
Precondition: System must be active, Member must be logged in.
Postcondition: System must continue to work normally as it should after any login, without considering the notifications.
Main Scenario:

1. member: logs in to the system.
2. system: shows up all notifications of the member to them.

Alternative Scenarios:

- No new notifications to show (2) so the system doesn't show anything to the member.

Guest user use cases

Use Case II.1.1 - Guest entrance

A user enters the system as a Guest and gets a cart.
Actors: user.
Precondition: System must be active.
Parameters: None
Main Scenario:

1. user: send an entrance request.
2. system: accept the user (start handling its requests).
3. system: set the user's state as Guest.
4. system: create cart associated with the user.
5. system: send a successful response to the user.

Alternative Scenarios:

- The user is disconnecting during 3-5, the system cancels the action and closes the user's connection.

Use Case II.1.2 - Guest disconnection

A Guest can disconnect from the system.
Actors: Guest.
Precondition: System must be active.
Parameters: None
Main Scenario:

1. guest: send a disconnection request.
2. system: delete guest's cart.
3. system: disconnects the guest and closes the connection.

Alternative Scenarios:

- The guest closes the connection (1), the system acts as a disconnection request has been received.

Use Case II.1.3 - Guest registration

A Guest can sign up to the system.
Actors: Guest.
Precondition: System must be active.
Parameters: authentication details
Postcondition: A member account with given authentication details exists.
Main Scenario:

1. guest: send a signup request.
2. system: verify authentication details using authentication details rules.
3. system: insert a new member with the given details.
4. system: sends a successful response to the guest (state left unchanged).

Alternative Scenarios:

- Guest's authentication details are not verified successfully (2), the system cancels the action and sends an error response.
- Guest tries to register with existing username in system (2), the system cancels the action and sends error response

Use Case II.1.4 - Guest login

A Guest can log in to the system.
Actors: Guest.
Precondition: System must be active.
Parameters: authentication details
Postcondition: The guest state has changed to a logged in member.
Main Scenario:

1. guest: send a login request.
2. system: verify authentication details using the existing system's members.
3. system: change guest state to logged in member.
4. system: sends a successful response to the guest.

Alternative Scenarios:

- user's authentication details not verified successfully (2), system cancels action and sends error response.

Use Case II.2.1 - Info request

A Guest can request information about the stores and products in stores.

Use Case II.2.1.a - Get stores info:

A Guest can request information about the stores in the system.

Actors: Guest.

Precondition: System must be active.

Parameters: None

Result: List of the stores currently exist and active in the system

Main Scenario:

1. guest: send a store info request.
2. system: filter the currently active stores in the system.
3. system: sends a successful response to the user containing a list of the stores.

Use Case II.2.1.b - Get products info:

A Guest can request information about the products in an active store of the system.

Actors: Guest.

Precondition: System must be active.

Parameters: store id

Result: List of the products currently exist in the store

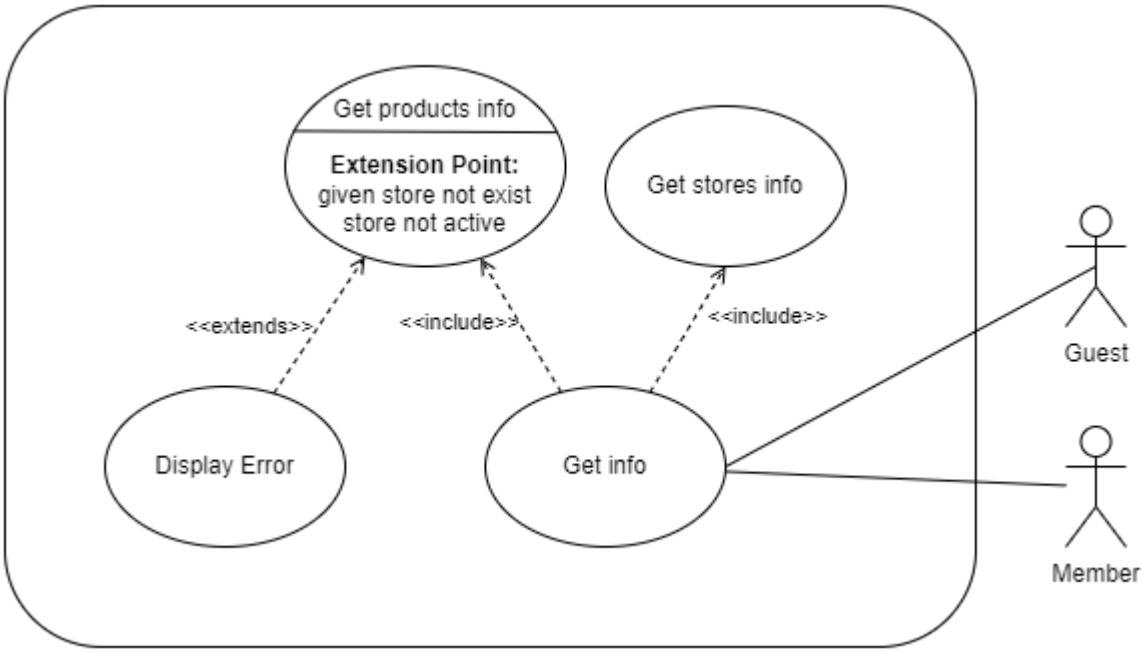
Main Scenario:

1. guest: send a product info request.
2. system: verify that the store exists and is active.
3. system: creates a list of products in the store.
4. system: sends a successful response to the user containing a list of products.

Alternative Scenarios:

- store does not exist in the system (2), cancels action and sends error response.
- store is not currently active (2), cancels action and sends error response.

Use Case II.2.1 - diagram



Use Case II.2.2 - Search products

A Guest can search products that exist in stores, using search properties and filter properties.
Actors: Guest.
Precondition: System must be active.
Parameters: search properties (mandatory) and filter properties (optional)
Result: List of the products found.
Main Scenario:

1. guest: send a product search request.
2. system: filter the products that match the given search and filter properties.
3. system: sends a successful response to the user containing a list of products.

Use Case II.2.3 - Adding products

A Guest can add products to his store bag, stored in the personal cart.
Actors: Guest.
Precondition: System must be active.
Parameters: product id, count, product details (optional)
Main Scenario:

1. guest: send an add product request.

2. system: add the product to the store bag of the guest, by the given count and store it in the personal cart.
3. system: sends a successful response to the user.

Use Case II.2.4 - Manage cart

A Guest can view and edit products in his personal cart.

Use Case II.2.4.a - View cart:
A Guest can view products in his personal cart.
Actors: Guest.
Precondition: System must be active.
Parameters: None
Result: personal cart details.
Main Scenario:
1. guest: send a view cart request.
2. system: sends a successful response to the user with the cart details.

Use Case II.2.4.b - Delete product from cart:
A Guest can delete a product from his personal cart.
Actors: Guest.
Precondition: System must be active, cart must contain the product.
Parameters: product id
Result: personal cart details.
Main Scenario:
1. guest: send a delete product request.
2. system: deletes the product from the guest's personal cart.
3. system: sends a successful response to the user with the cart details.

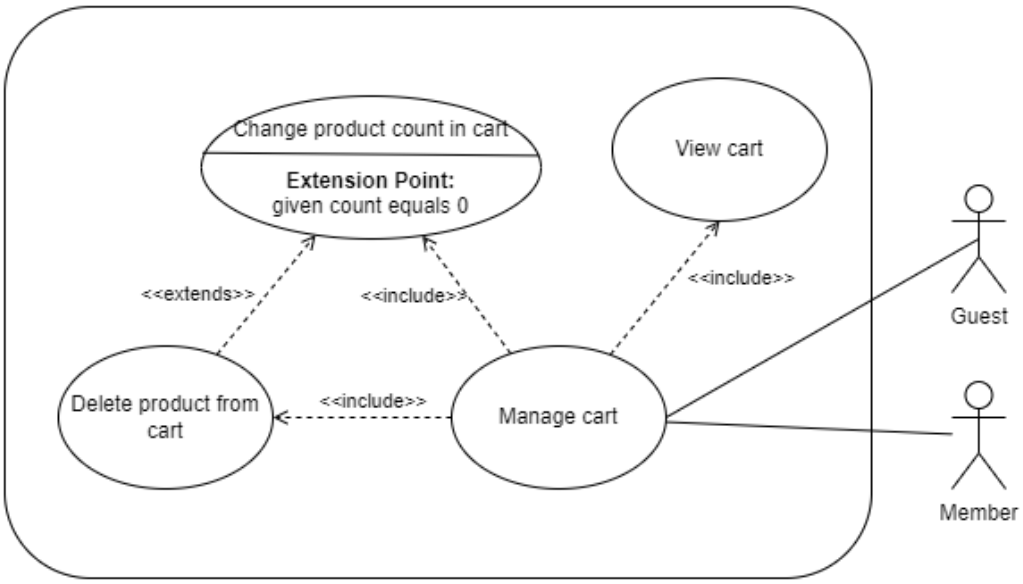
Use Case II.2.4.c - Change product count in cart:
A Guest can edit the product's count in his personal cart.
Actors: Guest.
Precondition: System must be active, cart must contain the product.
Parameters: product id, new count.
Result: personal cart details.
Main Scenario:
1. guest: send an edit product count request.

2. system: changes the product count to the given count in the guest's personal cart.
3. system: sends a successful response to the user with the cart details.

Alternative Scenarios:

- user sends count=0 in the request (1), system continues from step (2) in Use Case II.2.4.b.

Use Case II.2.4 - diagram



Use Case II.2.5 - Purchase cart content

A Guest can purchase his cart content according to the buy and discount policy for guests, and according to the product's availability in store.

Use Case II.2.5.a - Purchase cart content (available products):

A Guest can purchase his personal cart content if all the products are available.

Actors: Guest.

Precondition: System must be active, non-empty cart.

Parameters: payment method, payment details, delivery details

Postcondition: empty user cart, delivery supplier is handling the order.

Main Scenario:

1. guest: send a purchase request.
2. system: verify that all products are fully available in stores.
3. system: apply store and discount policy.

4. system: apply use case I.3.
5. system: apply use case I.4.
6. system: insert new order details into the system.
7. system: sends order details to the delivery company.
8. system: cleans the user's cart.
9. system: sends a successful response to the user with the order invoice and empty cart.
10. System: adds a transaction for the store and the member performing the purchase

Use Case II.2.5.b - Purchase cart content (products not available):

A Guest can't purchase his personal cart content if not all the products are available.

Actors: Guest.

Precondition: System must be active, non-empty cart.

Parameters: payment method, payment details, delivery details

Main Scenario:

1. guest: send a purchase request.
2. system: verify that all products are fully available in stores.
3. system: sends an error response telling which products are not available

Use Case II.2.5.c - Purchase cart content (payment fails):

A Guest can't purchase his personal cart content if the payment fails.

Actors: Guest.

Precondition: System must be active, non-empty cart.

Parameters: payment method, payment details, delivery details

Main Scenario:

1. guest: send a purchase request.
2. system: verify that all products are fully available in stores.
3. system: apply store and discount policy.
4. system: apply use case I.3 which returns failure.
5. system: sends an error response to the user.

Use Case II.2.5.d - Purchase cart content (delivery fails):

A Guest can't purchase his personal cart content if the delivery supply fails.

Actors: Guest.

Precondition: System must be active, non-empty cart.

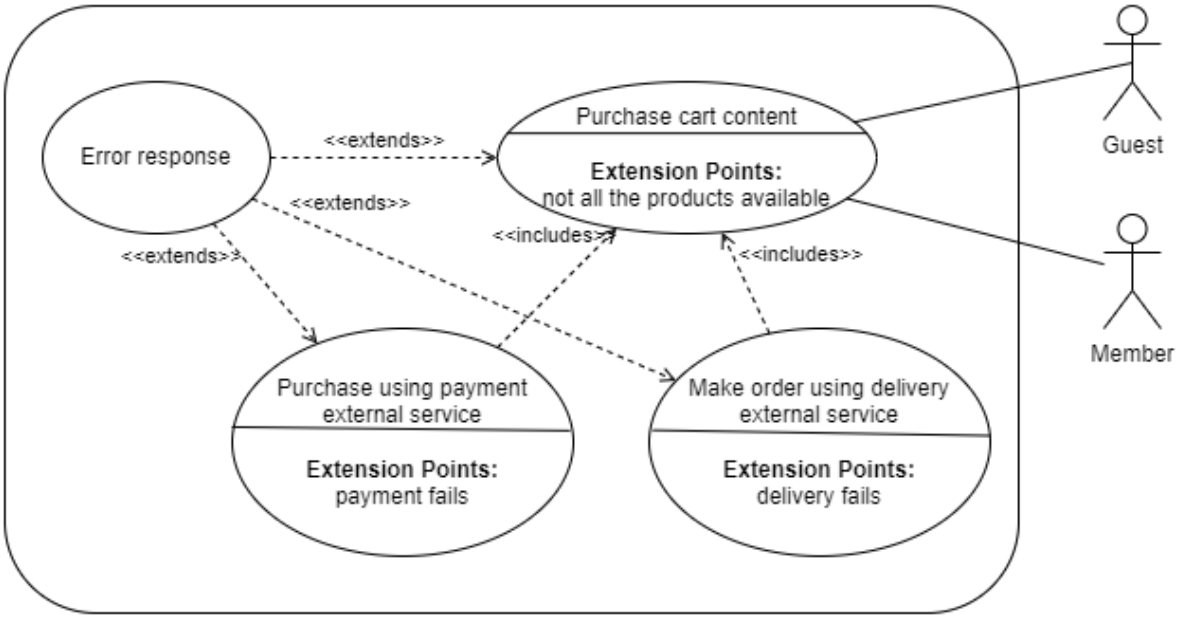
Parameters: payment method, payment details, delivery details

Main Scenario:

1. guest: send a purchase request.

2. system: verify that all products are fully available in stores.
3. system: apply store and discount policy.
4. system: apply use case I.3 for purchasing.
5. system: apply use case I.4 which returns failure.
6. system: cancels the payment using use case I.3.
7. system: sends an error response to the user.

Use Case II.2.5 - diagram



Member user use cases

Members are able to do use cases **II.2.1-II.2.5** exactly the same way as guests.
New precondition is that the user must be logged in.

Use Case II.3.0(1.2) - Member disconnection

A Member can disconnect from the system.
Actors: Logged in member.
Precondition: System must be active, user must be logged in.
Parameters: None
Main Scenario:

4. member: send a disconnection request.
5. system: saves the member's cart.
6. system: disconnects the member and closes the connection.

Alternative Scenarios:

- The member closes the connection (1), the system acts as a disconnection request has been received.

Use Case II.3.1 - Member logout

A Member can logout from the system.
Actors: Logged in member.
Precondition: System must be active, user must be logged in.
Parameters: None
Main Scenario:

1. member: send a logout request.
2. system: saves the member's cart.
3. system: changes user state to guest.
4. system: sends a successful response with a new empty cart.

Use Case II.3.2 - Create new store

A Member can create a new store and become its founder.
Actors: Logged in member.
Precondition: System must be active, user must be logged in.

Parameters: store information
Main Scenario:

1. member: sends request for creating new store.
2. system: creates a new store with given details.
3. system: sets the member to be the founder of the store.
4. system: sends a successful response with store id.

Use Case II.3.3 - Review a product

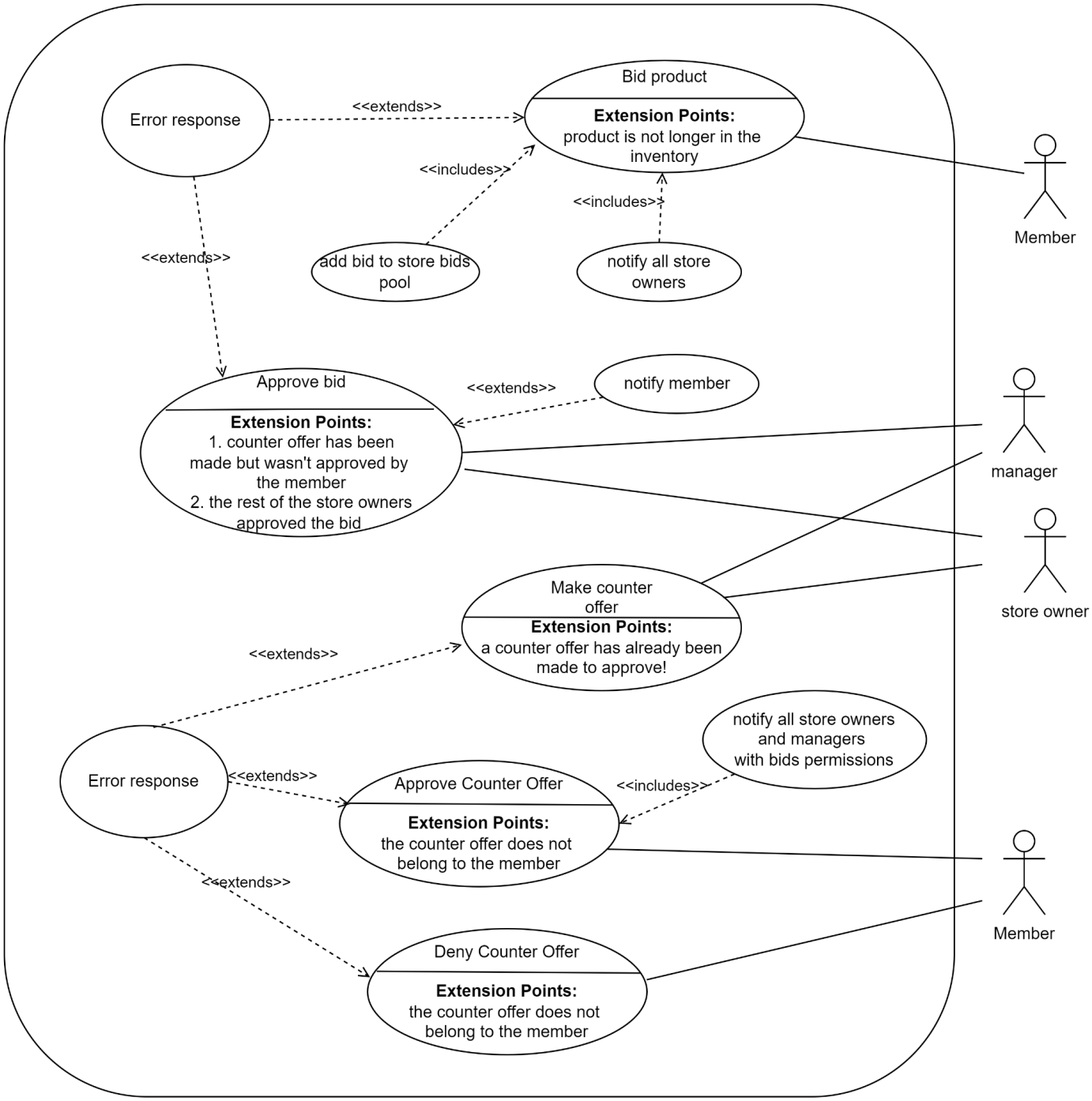
A Member can review a product in the store.
Actors: Logged in member, store owners.
Precondition: System must be active, user must be logged in, the referred product must exist in the inventory.
Parameters: store identification, product identification, review details
Main Scenario:

1. member: adds a product to his cart.
2. member: views the existing product reviews.
3. member: enters a new review of the product.
4. system: adds the product review record to the product's reviews records.
5. system: notifies all the store owners.
6. System: displays the new product reviews set to the member

system: response with a success message and displays the new product reviews set to the member.
Alternative Scenarios:

- The product is out of stock(1), the system replies with the appropriate failure message.
- Member is in fact a guest trying to leave a review (3), the system replies with the appropriate failure message and does not add the review.
- The product is no longer in the stock (4) , the system replies with the appropriate failure message and does not add the review.
- Some of the store owners are not logged in (5) in which case the notification will be added to the queued notification.

Use Case II.3 - bid a product



Store owner / manager (with permissions) use cases

Use Case II.4.1 - Inventory management

An owner can manage the store inventory (i.e. adding and remove products, changing the products' details)

Use Case II.4.1.a - Adding new product to the store:
Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in.
Parameters: new product details, amount in inventory.
Main Scenario:

1. owner: sends request for adding new product to the store.
2. system: creates new product collection using the given details.
3. system: sends a successful response.

Use Case II.4.1.b - Removing products from the store:
Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in, product id must be of an active product in store.
Parameters: product id.
Main Scenario:

1. owner: sends request for removing existing product from the store.
2. system: remove product from the store.
3. system: sends a successful response.

Use Case II.4.1.c - Changing product's quantity in inventory:
Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in, product id must be of an active product in the store.
Parameters: product id, new quantity.
Main Scenario:

1. owner: sends request for updating product's quantity in the store.
2. system: update product's quantity in store inventory.

3. system: sends a successful response.

<u>Use Case II.4.1.d - Changing product's details in the store:</u>

<u>Actors:</u> Store owner.

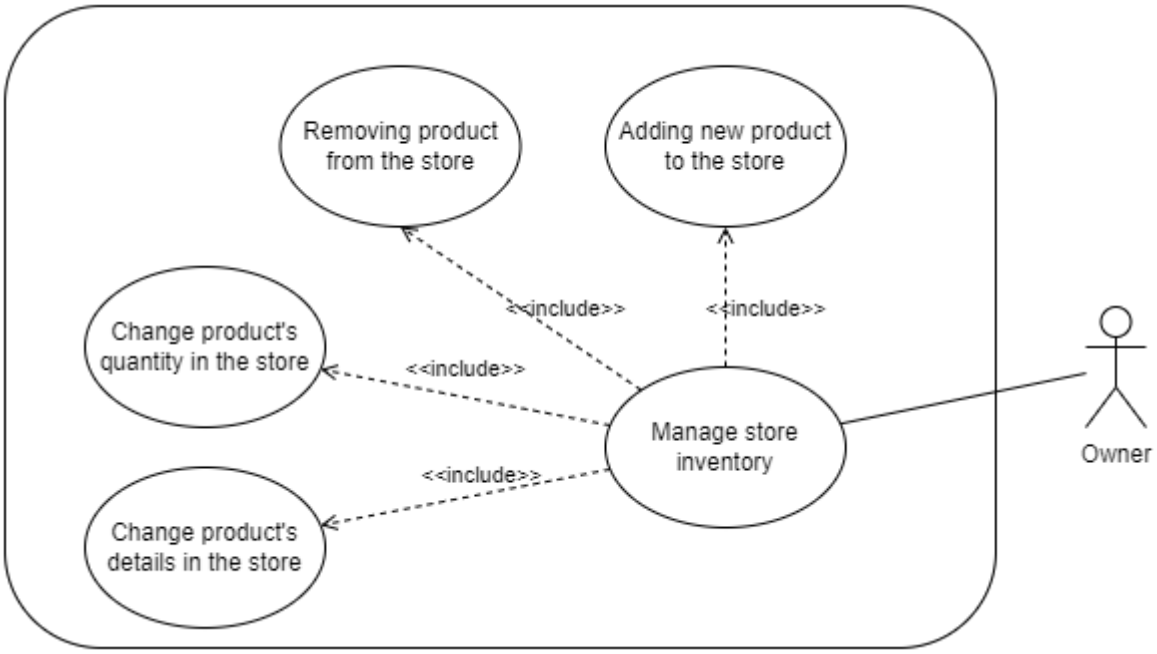
<u>Precondition:</u> System must be active, store must be active, store owner must be logged in, product id must be of an active product in the store.

<u>Parameters:</u> product id, new product's details.

<u>Main Scenario:</u>

1. owner: sends request for updating product's details in the store.
2. system: update product's details in store.
3. system: sends a successful response.

<u>Use Case II.4.1 - diagram</u>



Use Case II.4.2 - Change type of purchases and discount policy

<u>Actors:</u> Store owner.

<u>Precondition:</u> System must be active, store must be active, store owner must be logged in.

<u>Parameters:</u> store id, purchases and discount policy.

<u>Postcondition:</u> New policy should apply

<u>Main Scenario:</u>

1. owner: sends request for changing policy in the store.

2. system: sets new policy for the store.
3. system: sends a successful response.

Use Case II.4.4 - Make store co-owner

Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in.
Parameters: member id.
Postcondition: Store owners hierarchy is a tree (no cycles).
Main Scenario:

4. owner: sends request for making new owner in the store.
5. system: verify member exists and not already owner.
6. System: waits for the approval of the other owners for the appointement of the new co-owner
7. system: sets new owner under the actor's (owner) in the hierarchy.
8. system: sends a successful response.

Alternative Scenarios:

- The member does not exist (2), the system cancels the action.
- The member is already an owner of the store (2), the system cancels the action.
- The appoitment is rejected by one of the other owners so the new co-owner is not added to the system.

Use Case II.4.5 - Remove store co-owner

Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in.
Parameters: member id.
Postcondition: Store owners hierarchy is updated (no cycles).
Main Scenario:

1. owner: sends request for removing owner in the store.
2. system: verify member exists and an owner.
3. system: update the hierarchy branch of the remover with the new exclusion.
4. system: sends a successful response to remover.
5. system: sends a notification message to the member that was removed.

Alternative Scenarios:

- The Remover isn't an owner/does not have the suitable permissions for such action (2), the system cancels the action.
- The member isn't an owner of the store (2), the system cancels the action.
- The remover does not have the suitable position in the hierarchy for the remove action, the system cancels the action with an appropriate message(3).
- The removed member isn't logged in(5., the notification adds up to the queued notifications and displayed to the member when he connects to the system.

Use Case II.4.6 - Make store manager

Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in.
Parameters: member id.
Postcondition: The new manager permissions are getting info (4.12, 4.13)
Main Scenario:

1. owner: sends request for making new manager in the store.
2. system: verify member exists and not already manager or co-owner.
3. system: sets the new manager with the owner as appointor.
4. system: sends a successful response.

Alternative Scenarios:

- The member does not exist (2), the system cancels the action.
- The member is already a manager or co-owner of the store (2), the system cancels the action.

Use Case II.4.7 - Change store manager permissions

Actors: Store owner.
Precondition: System must be active, store must be active, store owner must be logged in, permissions before action restrict the member's actions..
Parameters: member id of manager, new permissions.
Postcondition: The manager's new permissions updated and restrictions have been applied.
Main Scenario:

1. owner: sends request for making new permissions for manager of the store.
2. system: verify member exists and is manager.
3. system: sets the new manager permissions.

4. system: sends a successful response.

Alternative Scenarios:

- The member is not a manager of the store (2), the system cancels the action.

Use Case II.4.9 - Make store inactive

Actors: Founder.

Precondition: System must be active, store must be active, founder must be logged in, the store's managers and owners exist in the system.

Parameters: member id of founder, store id of the store to be closed.

Postcondition: The store owners and managers have been notified properly and the closed store products won't show it's products in the product search.

Main Scenario:

1. founder: sends request for closing one of his stores.
2. system: verify that the member exists and is a founder of the store.
3. system: verify that the store is indeed currently opened.
4. system: mark the store as closed.
5. system: sends a notification message to the store owners and managers.
6. system: sends a successful response.

Alternative Scenarios:

- The member is not the founder of the store (2), the system cancels the action.
- The store is already closed (3), the system cancels the action.

Use Case II.4.11 - Get store roles information

Store owner can get information about the roles in store, and managers permissions.

Use Case II.4.11.a - Get store roles list:

Actors: Store owner.

Precondition: System must be active, store owner must be logged in.

Parameters: member id of the store owner, store id of the relevant store.

Main Scenario:

1. store owner: sends a request for information about the store's roles.
2. system: verify that the member exists and is a store owner of the store.
3. system: sends a successful response containing the store's members id's with their roles.

Alternative Scenarios:

- The member is not a store owner of the store (2), the system cancels the action.

Use Case II.4.11.b - Get store's managers permissions:

Actors: Store owner .

Precondition: System must be active, store owner must be logged in.

Parameters: member id of the store owner, store id of the relevant store.

Main Scenario:

1. store owner: sends a request for information about the store manager's permissions.
2. system: verify that the member exists and is a store owner of the store.
3. system: sends a successful response containing the store manager's id along with their permissions.

Alternative Scenarios:

- The member is not a store owner of the store (2), the system cancels the action.

Use Case II.4.13 - Get purchase history in store by store owner

Actors: Store owner .

Precondition: System must be active, store owner must be logged in.

Parameters: member id of store owner, store id of the relevant store.

Main Scenario:

1. store owner: sends a request for a purchase history of a certain store. .
2. system: verify that the member exists and is a store owner of the store.
3. system: sends a successful response containing the history of the store purchases history.

Alternative Scenarios:

- The member is not a store owner of the store (2), the system cancels the action.

Admin user use cases

Use Case II.6.2 - Removal of a member from the market

Actors: System manager

Precondition: System must be active, system manager must be logged in.

Parameters: Admin id, member id.

Main Scenario:

1. The admin requests the removal of a member with it's credentials(admin id) and inputs the required information regarding the member(member id).
2. The system verifies that the admin of the given id exists in the system, and that the member id corresponds to an existing member in the system, and that the member has no roles in the market.
3. The system removes the member and his cart from the system.

Alternative Scenarios:

- The admin id entered by the system manager (2) is for an admin not currently existing in the system so the action can't be performed so an error message is returned via an alert.
- The member id does not exist in the system (2) so the action can't be performed so an error message is returned via a notification.
- The member to be removed has roles at the market (2), in which case the system aborts the action and responds with the appropriate alert.

Use Case II.6.4 - Get the purchase history of a store/buyer by a system manager (Admin)

Actors: System manager

Precondition: System must be active, system manager must be logged in.

Parameters: Admin id, store/buyer id.

Main Scenario:

1. The admin requests the purchase history of the store/buyer and inputs the required credentials(admin id) and the information regarding the store/buyer(store/buyer id).
2. The system verifies that the admin of the given id exists in the system, and that the store/buyer id corresponds to an existing store/buyer in the system.
3. The system outputs the purchase history of the entered store/buyer.

Alternative Scenarios:

- The admin id entered by the system manager is for an admin not currently existing in the system so the action can't be performed so an error message is returned via a notification.
- The store/buyer id does not exist in the system so the action can't be performed so an error message is returned via a notification.

Use Case II.6.5 - Watch the login statistics of the system (Admin)

Actors: System manager
Precondition: System must be active, system manager must be logged in.
Parameters: Admin id, dates.
Main Scenario:

1. Admin asks for the statistics for certain dates
2. System: verifies the user asking is admin
3. System: counts all the logged in users in the system during the dates enterd
4. System: returns the statistics to present to the admin

Alternative Scenarios:

- The verification of the admin fails and the action is not performed
- The admin enterd todays date so the system also sends notifications for new users entering the system.