Use Case Name	Starting up the system.
Requirement Number	1.1
Actors	User
Parameters	None
Preconditions	1. The machine has the latest version of the system. 2. The machine has a stable connection.
Actions	1. The User runs the application.
Result	The User is appointed Guest and receives an empty Shopping cart.
Data	**Result**
The User has no connection to the server.	Fail
The User has an older version of the client.	Fail

Use Case Name	Exiting the system.
Requirement Number	1.2
Actors	User
Parameters	None
Preconditions	None
Actions	1. The User closes the client application. 2. If the User is logged in, the System runs logging out use case.
Result	None.
Data	**Result**
A Member is existing the System, his Shopping cart is reloaded.	Success
A Guest is exiting the System, his Shopping cart is discarded.	Success

Use Case Name	Registering
Requirement Number	1.3
Actors	Guest
Parameters	*Username, Password.*
Preconditions	The *Username* is not already known in the System.
Actions	1. The User requests to register with some *Username* and *Password.* 2. The System Verifies the *Password*'s security level and that the *Username* does not already exist. 3. If one of the previous verifications fails, the User is not registered and the System notifies the User. 4. Else, a new Member is registered in the System.
Result	A new member can now be logged in to in the System.
Data	**Result**
The User enters a Username that's already in use.	Fail
The User is trying to use an insecure password.	Fail
The User is using a unique username with a secure password.	Success

Use Case Name	Login
Requirement Number	1.4
Actors	Guest
Parameters	*Username, Password.*
Preconditions	1. The requested user was registered in the system beforehand. 2. The system has connection.
Actions	1. Guest inputs *Username* and *Password* 2. If the Guest is already logged in the process is stopped and an error message appears. 3. The System verifies the *Username, Password* combination. 4. If the System doesn't recognize the combination the Guest is notified with the appropriate error message. 5. The System loads the Member's saved Shopping Cart and notifications.
Result	The Guest receives logged in status for the requested Member, and the Shopping Cart receives earlier saved Products.
Data	**Result**
The Username is unknown to the system.	Fail
The password is incorrect.	Fail
The username password combination is correct.	Success

Use Case Name	Searching for a Store.
Requirement Number	2.1
Actors	User
Parameters	*Store_attributes*
Preconditions	None
Actions	1. User requests stores that have some specific attributes. 2. The System presents all stores that apply. 3. User selects a specific store. 4. The System presents all the Products the Store has and any other public information.
Result	A Store.
Data	**Result**
The User searches for a store that doesn't exist.	Success
The User searches for a specific known store.	Success

Use Case Name	Searching for a Product.
Requirement Number	2.2
Actors	User
Parameters	*Product_attributes*
Preconditions	None
Actions	1. User requests a Product with specific attributes from the System. 2. The System presents all Products from any Store, which apply to those attributes. 3. The User selects a specific Product.
Result	A Product.
Data	**Result**
The User is searching for a product that doesn't exist.	Success

The User is searching for a product that exists.	Success

Use Case Name	Adding a Product to the Shopping Cart.
Requirement Number	2.3
Actors	User
Parameters	*Product*
Preconditions	1. The Product is sold by some Store. 2. The User has a Shopping Cart.
Actions	1. The User finds a Product (either from use case 2.2 or 2.1). 2. The User requests from the System to add the Product to the Shopping Cart.
Result	The User's Shopping Cart receives the Product from the appropriate store.
Data	**Result**
The User is adding a product from a Store.	Success
The User is adding a product found from searching a product.	Success
The User is adding more of the product than the store has to offer.	Fail

Use Case Name	Inspecting Shopping Cart
Requirement Number	2.4
Actors	User
Parameters	*Username*
Preconditions	None
Actions	1. User requests for his Shopping Cart. 2. The System presents the Shopping Cart.
Result	A Shopping Cart
Data	**Result**
The User is inspecting his shopping cart when it has no Products.	Success
The User is inspecting his shopping cart when it has 1 or more products.	Success

Use Case Name	Removing a Product from the Shopping Cart
Requirement Number	2.4
Actors	User
Parameters	*Product, Shopping_Cart*
Preconditions	*Shopping_Cart* has at least 1 *Product*.
Actions	1. User requests from the System to remove *Product* from his shopping cart. 2. The System fetches the User's *Shopping_Cart*. 3. If the *Shopping_Cart* doesn't have the *Product*, the user is notified with the appropriate error message.
Result	*Shopping_Cart* no longer has *Product*.
Data	**Result**
The User is attempting to remove a product from an empty cart.	Fail
The User is attempting to remove a product that exists in his shopping cart.	Success

Use Case Name	Purchasing a Shopping Cart, with verified delivery and payment.
Requirement Number	2.5
Actors	User
Parameters	*Shopping_Cart, Payment, Supplyment*
Preconditions	1. The *Shopping_Cart* is not empty. 2. For every Product in the *Shopping_Cart* the Store which sells it has the amount of that Product that is listed in the *Shopping_Cart*. 3. The *Supplyment* and *Payment* succeeded.
Actions	1. The User requests from the System to buy his shopping cart with a specific Payment type and Supplier. 2.The System verifies the *Payment*. 3. The System verifies the *Supplyment.* 4. The System triggers the payment and supply, updates the Store with the amounts and triggers the purchase event on the notification bus.
Result	The Purchase is completed, the relevant Members are notified and the relevant amounts on the Stores are updated.
Data	**Result**
The User is attempting to buy an empty shopping cart.	Fail
The User is attempting to buy a shopping cart with at least 1 product.	Success
The User is attempting to buy a shopping cart without selecting payment type or supplier.	Fail

Use Case Name	Purchasing a Shopping Cart, when payment method fails.
Requirement Number	2.5
Actors	User
Parameters	*Shopping_Cart, Payment, Supplyment*
Preconditions	1. The *Shopping_Cart* is not empty. 2. For every Product in the *Shopping_Cart* the Store which sells it has the amount of that Product that is listed in the *Shopping_Cart*. 3. The *Payment* fails to verify.
Actions	1. The User requests from the System to buy his shopping cart with a specific Payment type and Supplier. 2.The System attempts to verify the *Payment*. 3. The System informs the User that the payment failed and that his purchase didn't go through.
Result	The Purchase is not completed and the User is notified.
Data	**Result**
The User is attempting to buy with illegal credit card information.	Fail
The User is attempting to use a credit card without enough funds.	Fail
The User is attempting to buy a shopping cart without selecting payment type.	Fail

Use Case Name	Purchasing a Shopping Cart, when delivery service fails.
Requirement Number	2.5
Actors	User
Parameters	*Shopping_Cart, Payment, Supplyment*
Preconditions	1. The *Shopping_Cart* is not empty. 2. For every Product in the *Shopping_Cart* the Store which sells it has the amount of that Product that is listed in the *Shopping_Cart*. 3. The *Payment* verifies successfuly. 4. The *Supplyement* fails to verify.
Actions	1. The User requests from the System to buy his shopping cart with a specific Payment type and Supplier. 2.The System verifies the *Payment*. 3. The System attempts to verify the *Supplyment*. 4. The System fails to verify. 5. The System cancels the *Payment*. 6. The notifies the User.
Result	The Purchase is not completed and the User is notified.
Data	**Result**
The User is attempting to buy with illegal address.	Fail
The User is attempting to use a supplier that doesn't operate in his address.	Fail
The User is attempting to buy a shopping cart without selecting supplier type.	Fail

Use Case Name	Logging out
Requirement Number	3.1
Actors	Member
Parameters	*Member*
Preconditions	The *Member* is logged in
Actions	1. The Member requests to log out from the System. 2. The System saves his/her current Shopping Cart content 3. The System changes the Member's status to Guest.
Result	The Member becomes a Guest and the Shopping Cart is saved.
Data	**Result**
A Guest is attempting to log out.	Fail
The Member is logging out with products in his cart.	Success
The Member is logging out with no products in his cart.	Success

Use Case Name	Opening a Store
Requirement Number	3.2
Actors	Member
Parameters	None
Preconditions	The Member is logged in
Actions	1. The Member requests to open a Store with some attributes. 2. The System verifies the attributes are legal, if it fails it notifies the Member. 3. The System makes a new Store and appoints the Member as a Founder.
Result	A new Store is created.
Data	**Result**
A Guest is attempting to open a store.	Fail
A Member is attempting to open a store with invalid store attributes.	Fail
A Member is attempting to open a store with legal attributes.	Success

Use Case Name	Writing a review for a Product
Requirement Number	3.3
Actors	Member
Parameters	*Member_ID, Product, review_string*
Preconditions	The Member previously Purchased the Product
Actions	1. The Member requests to submit a review for a Product. 2. The System verifies that the *Product* is still sold by the Store, and that the *review_string* is legal. 3. If the System fails to verify one of the above, the review is not submitted, and the Member is notified.
Result	The System posts the review and notifies relevant Members.
Data	**Result**
A Guest is attempting to write a review.	Fail
A Member is trying to write a review with 500+ characters.	Fail
A Member is trying to write a review for a product he did not purchase.	Fail
A Member is trying to write a review for a product he did purchase.	Success

Use Case Name	Rating a Product and Store
Requirement Number	3.4
Actors	Member
Parameters	*Rating, Store, Product*
Preconditions	The Member haven't rated this Store and Product before.
Actions	1. The Member requests from the System to rate with a specific rating. 2. The System verifies the preconditions and that the *Rating* has a legal value. 3. If the System fails to verify, the Member is notified, and the rating is not submitted.
Result	The rating is submitted.

Data	**Result**
A Member is trying to rate twice the same Store and Product.	Fail
A Guest is trying to rate.	Fail
A Member is trying to rate with legal rating value.	Success
A Member is trying to rate with illegal rating value.	Fail

Use Case Name	Sending a message to a Store
Requirement Number	3.5
Actors	Member
Parameters	*Store, Subject_string, body_string*
Preconditions	None
Actions	1. The Member selects a store and requests to send a message with a *Subject_string* title and *body_string.* 2. The System sends the Message to the relevant Members and notifies them.
Result	The relevant Members are notified and receive the message.

Data	**Result**
A Member is attempting to send a message with invalid body or title length.	Fail
A Member is attempting to send a message to a Store that doesn't exist.	Fail
A Member is attempting to send a message with legal values for title and subject.	Success

Use Case Name	Getting information about former purchases
Requirement Number	3.7
Actors	Member
Parameters	*None*
Preconditions	Member is logged in
Actions	1. The member requests the system for his purchase history 2. The system fetches his purchase history
Result	The relevant Member receive his purchase history

Data	**Result**
The Member is not logged in	Fail
The Member's purchasing history is empty	Success
The Member has made some purchases	Success

Use Case Name	Receiving information
Requirement Number	3.8
Actors	Member
Parameters	*The identification parameter the user wants to change*
Preconditions	Member is logged in
Actions	1. The member requests the system to view his information 2. The system fetches the member's information and shows it to him
Result	The relevant Member gets his information

Data	**Result**
The Member is logged in	Success
The Member is not loggen in	Fail

Use Case Name	changing identification parameters
Requirement Number	3.8
Actors	Member
Parameters	*The identification parameter the user wants to change*
Preconditions	Member is logged in
Actions	1. The member requests the system to change some of the parameters 2. The system prompts the member to enter new identification parameters 3. The member enter new identification parameters 4. The system verifies the parameters are valid and changes them
Result	The relevant Member is notified that the parameters have changed

Data	**Result**
The Member's new identifications are valid	Success
The Member's new identification are invalid	Fail
The member is not logged in	Fail

Use Case Name	Upgrading the account security
Requirement Number	3.9
Actors	Member
Parameters	*answer_string*
Preconditions	The member is logged in
Actions	1. The Member requests the system to upgrade his account security 2. The System prompts the member with possible security questions 3. The member selects his preferred question 4. The member enters answer_string as an answer to a security question 5. The system saves the question and the member's answer.
Result	The relevant Member is notified the process has been completed successfully.

Data	**Result**
The member enters an empty string as an answer	Fail
The member submits a valid answer	Success
The member has the maximum security possible	Fail

Use Case Name	Update product information
Requirement Number	4.1,5
Actors	Manager
Parameters	*Store, Product*
Preconditions	1. Store manager must have proper permission from store owner 2. Store manager must have proper permission from store owner
Actions	1. The Manager chooses one of the options: edit or delete 2. The System verifies the Manager has the right permissions to perform those actions 3. If the Manager chose delete: 3.1. The system deletes the product 4. If the Manager chose to edit: 4.1. The system prompts the Manager for updated values for the product. 4.2. The Manager enters new values 4.3. The system updates the product
Result	The relevant Manager is notified about the change

Use Case Name	Manage store inventory and update products
Requirement Number	4.1,5
Actors	Manager, Store,Store Manager
Parameters	*None*
Preconditions	1. Manager is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Manager selects a store and requests to see the store's products. 2. The System shows the manager all the available products 3. If the manager requests to edit a product
Result	The relevant Member gets the information about the store

Data	Result
The Manager doesn't have the right permissions	Fail
The Manager has permissions and he deletes/updates a product	Success
The Manager doesn't want to delete/update a product and just see the inventory	Success

Use Case Name	Change store’s buying policy
Requirement Number	4.2,5
Actors	Owner,Store Manager
Parameters	*Store, discount_policy*
Preconditions	1. The owner is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to change store buying policy 2. The System verifies the owner is the store owner 3. The owner enters his new buying policy 4. The system verifies that the new policy isn’t in conflict with the founder’s Consistency rules. 5. If the policy is valid the system updates the store’s buying policy
Result	The system updates the store policy and notifies the owner

Data	**Result**
The Owner is not really the owner of the store	Fail
The new buying policy is in conflict with the founder’s consistency rules	Fail
The Owner has the right permissions and the new policy is valid	Success

Use Case Name	Change store’s discount policy
Requirement Number	4.2,5
Actors	Owner,Store Manager
Parameters	*Store, discount_policy*
Preconditions	1. The owner is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to change store discount policy 2. The System verifies the owner is the store owner 3. The owner enters his new discount policy 4. The system verifies that the new policy isn’t in conflict with the founder’s Consistency rules. 5. If the policy is valid the system updates the store’s discount policy
Result	The system updates the store policy and notifies the owner

Data	**Result**
The Owner is not really the owner of the store	Fail
The new buying policy is in conflict with the founder’s consistency rules	Fail
The Owner has the right permissions and the new policy is valid	Success

Use Case Name	Set Consistency rules for a store
Requirement Number	4.3,5
Actors	Founder,Store Manager

Parameters	*Store, rules*
Preconditions	1. The founder is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Founder requests the system to set the store's consistency rules 2. The System verifies the user is the founder of the store 3. The Founder enters his desired rules 4. The system updates the store consistency rules
Result	The system sets the consistency rules and notifies the Founder

Data	**Result**
The Founder is not really the founder of the Store	Fail
The founder is the founder of the store and he enters his desired policies	Success

Use Case Name	Appoint store owner
Requirement Number	4.4,5
Actors	Owner,Store Manager
Parameters	*new_owner,Store*
Preconditions	1. The Owner is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to add a store owner to his Store 2. The System verifies the Owner is the Store owner 3. The system verifies the new_owner is not already the store owner
Result	The system sets the new store manager and updates his permission and notifies the Owner.

Data	**Result**
The Owner is not the Store owner	Fail
The new_owner already owns the Store	Fail
The owner is the store owner and the new_owner is not	Success

Use Case Name	Remove Store owner
Requirement Number	4.5,5
Actors	Owner,Store Manager
Parameters	*Store, owner_to_remove*
Preconditions	1. The Owner is logged in

	2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to remove a store owner from his Store 2. The system verifies the Owner is the Store owner 3. The System verifies owner_to_remove was appointed by the Owner
Result	The system removes the permissions of owner_to_remove and all other members that were appointed by owner_to_remove and notifies the Owner

Data	**Result**
The owner is not the Store owner	Fail
The owner_to_remove wasn’t appointed by Owner	Fail
The Owner is the owner and owner_to_remove was appointed by him	Success

Use Case Name	Appoint store manager
Requirement Number	4.6,5
Actors	Owner,Store manager
Parameters	*new_manager, Store*
Preconditions	1. The Owner is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to add a store manager to his Store 2. The System verifies the Owner is the Store owner 3. The system verifies new_manager is not the current Store owner/manager.
Result	The system updates new_manager permissions and notifies the Owner

Data	**Result**
The owner is not the Store owner	Fail
The new_manager is already the Store’s manager/owner	Fail
The new_manager is not the Store’s manager/owner	Success

Use Case Name	Change store manager permissions
Requirement Number	4.7,5
Actors	Owner,Store manager
Parameters	*manager, new_permissions,Store*
Preconditions	1.The Owner is logged in

	2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to manager permissions 2. The System verifies the Owner is the Store owner 3. The Owner enters the new_permissions
Result	The system updates the manager permissions and notifies the Owner

Data	**Result**
The owner is not the Store owner	Fail
the manager is not the Store’s manager	Fail
The owner chooses correct permissions	Success

Use Case Name	Remove store manager
Requirement Number	4.8,5
Actors	Owner,Store Manager
Parameters	*manager_to_remove,Store*
Preconditions	1. The Owner is logged in 2. Store manager must have proper permission from store owner
Actions	1. The Owner requests the system to remove manager_to_remove from his position as Store manager 2. The System verifies the Owner is the Store owner 3. The system verifies manager_to_remove was appointed by the Owner
Result	The system removes manager_to_remove permissions as store manager and notifies the Owner

Data	**Result**
The owner is not the Store owner	Fail
The manager_to_remove was not appointed by the Owner	Fail
The manager_to_remove was appointed by the Owner	Success

Use Case Name	Filling complaint to System admin
Requirement Number	3.6
Actors	User

Parameters	*title_string, body_string, purchase_id*
Preconditions	None
Actions	1. The user requests the system to send a complaint to the system administrator about a purchase. 2. The system prompts the user to enter title,body and purchase id. 3. the user enter title_string,body_string and purchase_id 4. The system sends the message and notifies the system administrator
Result	The relevant Member is notified about the message

Data	**Result**
The User enters an invalid title/body	Fail
The user didn’t purchase anything	Fail
The user purchased something and he filled valid title and body	Success

Use Case Name	Close a store
Requirement Number	4.9, 5
Actors	Store Founder, Store owner, Store manager
Parameters	Store ID
Preconditions	1. Store exists 2. User is the store’s founder 3. Store manager must have proper permission from store owner
Actions	1. The store founder asks the system to close a store 2. The store founder inputs a store that he founded 3. The system verifies that the store exists and is not in “closed” state already 4. The system verifies the relevant user permissions 5. The system changes the store status to “closed” 6. The system sends the store owner and manager a “Store closing message” 7. The system removes the store’s products from the product search view
Result	1. Store’s status is changed to “closed” 2. Store is only visible to admins and store owners 3. Store’s product are not visible on search 4. “Store closing message” is sent to store’s owner and manager

Data	**Result**
User enters a store ID that doesn’t exist	Fail
User enters a store ID of a store he is not the founder of	Fail

User enters a store ID of a store he founded	Success

Use Case Name	Reopen a closed store
Requirement Number	4.10, 5
Actors	Store founder, Store owner, Store manager
Parameters	Store ID
Preconditions	1. Store exists and in “closed” status 2. User is the store’s founder 3. Store manager must have proper permission from store owner
Actions	1. Store founder chooses to reopen a store 2. Store founder inputs a store ID 3. The system verifies that the store exists and in not in “open” state already 4. The system verifies the relevant user permissions 5. The system changes store’s status to “open” 6. The system sends a “Store reopening message” to the store owner and manager
Result	1. Store’s status Is changed to “open” 2. Store’s manager and owner receive a “Store reopening message”
Data	**Result**
User enters a store ID that doesn’t exist	Fail
User enters a store ID that is not closed	Fail
User enters a store ID of a store he founded that is currently closed	Success
User enters a character that is not a number	Fail
Use Case Name	Get store employees information
Requirement Number	4.11, 5
Actors	Store founder, Member, Store manager
Parameters	Member ID
Preconditions	1. Member must exist 2. Member ID must be of a store employee 3. Store manager must have proper permission from store owner
Actions	1. Store founder asks the system to get information about store employees 2. Store founder chooses the employee he wants to get information on 3. The system verifies the employee exists in the store 4. The system verifies the relevant user permissions 5. The system provides the store founder with the information
Result	1. Store founder is shown the employee’s information
Data	**Result**
User enters a member ID of a member that doesn’t exist	Fail
User enters a negative number	Fail
User enters a member that is not an employee in the store	Fail
User enters a member ID of a store employee	Success

Use Case Name	Get store managers permission
Requirement Number	4.11, 5
Actors	Store owner, Store manager

Parameters	Member ID
Preconditions	1. Store manager must exist 2. Member ID must be of a store manager of the store the owner owns 3. Store manager must have proper permission from store owner
Actions	1. Store owner asks the system to get the store manager’s permissions list 2. Store owner chooses the store manager he wants to get information on 3. The system verifies that the manager the owner asked for is the manager of the store he owns 4. The system verifies the relevant user permissions 5. The system provides the store owner with the permissions list
Result	1. Store owner is shown the store manager’s permission’s list

Data	**Result**
User enters a member ID of an employee that is not a manager	Fail
User enters a member ID of a member that is not an employee	Fail
User enters a member ID that doesn’t exist	Fail
User enters a member ID of a store manger of a different store	Fail
User enters a member ID of a store manager of the store he owns	Success

Use Case Name	Receive and read member’s questions
Requirement Number	4.12, 5
Actors	Store owner, Store manager
Parameters	None
Preconditions	Store manager must have proper permission from store owner
Actions	1. The store owner chooses to read member’s questions 2. The system verifies the relevant user permissions 3. The system shows the member’s questions to the store owner
Result	1. Store owner is shown the member’s questions list

Data	**Result**
User (Not a store owner) chooses the read member’s questions option	Fail
User (Store owner) chooses the read member’s questions option	Success

Use Case Name	Answer member’s questions
Requirement Number	4.12, 5
Actors	Store owner, Store manager
Parameters	Answer text
Preconditions	Store manager must have proper permission from store owner
Actions	1. Store owner chooses a member’s questions 2. Store owner chooses to answer the question 3. Store owner inputs the answer text 4. The system verifies the text is not empty

	5. The system verifies relevant user permissions 6. The system adds the question to the member’s question
Result	1. The answer is added to the member’s question
Data	**Result**
User enters an empty string as answer	Fail
User enters a non-empty string as answer	Success

Use Case Name	Get store’s purchase history
Requirement Number	4.13, 5
Actors	Store owner, Store manager
Parameters	Store ID
Preconditions	1. Store ID must be of a store that is owned by the store owner 2. Store manager must have proper permission from store owner
Actions	1. The store owner asks the system to get store purchase history 2. The store owner inputs the store ID 3. The system verifies that the store exists 4. The system verifies the relevant user permissions 5. The system provides the purchase history for that store
Result	Store’s purchase history is provided to the store owner
Data	**Result**
User enters a store ID that doesn’t exist	Fail
User enters a store ID of a store he doesn’t own	Fail
User enters a store ID of the store he owns	Success

Use Case Name	Closing store permanently
Requirement Number	6.1
Actors	Admin, Store Owner, Store Manager
Parameters	Store ID
Preconditions	1. Store exists
Actions	1. The admin inputs a store to close 2. The system verifies that the store exists 3. The system verifies the relevant user permissions 4. The system sends “Store closing message” to store owners and managers 5. The system cancels the membership of the store owners and managers
Result	The store is no longer available to users and store managers and owners are no longer members
Data	**Result**
User enters a store ID of a store that doesn’t exist	Fail
User enters a store ID of a store that is already closed	Fail
User enters a store ID of an open store	Success

Use Case Name	Canceling membership
Requirement Number	6.2
Actors	Admin, Member
Parameters	Member ID
Preconditions	1. Member exists and registered
Actions	1. The admin inputs a member ID he wishes to remove 2. The system verifies that the member exists 3. The system verifies the relevant user permissions 4. The system cancels all permissions and roles of the member.
Result	The membership is canceled

Data	**Result**
User enters a member ID of a member that doesn’t exist	Fail
User enters a character that is not a number	Fail
User enters an empty string	Fail
User enters a member ID of an existing member	Success

Use Case Name	Read and comment to complaints
Requirement Number	6.3
Actors	Admin, Member
Parameters	Comment text
Preconditions	None
Actions	1. The admin get information and reads the members complaints 2. The admin chooses a complaint 3. The admin comments on the member complaint 4. The system verifies the text is not empty 5. The system verifies the relevant user permissions
Result	A comment is added to the member’s complaint

Data	**Result**
User enters an empty string as comment	Fail
User enters a non-empty string as comment	Success

Use Case Name	Send messages to buyers and sellers
Requirement Number	6.3
Actors	Admin, Buyer/Seller
Parameters	Message text
Preconditions	None

Actions	1. The admin chooses a buyer or seller 2. The admin chooses to send the member a message 3. The admin inputs the message text 4. The system verifies the text is not empty 5. The system verifies the relevant user permissions 6. The system sends to message to the selected member
Result	A message is sent to the selected member
Data	**Result**
User enters an empty string as message	Fail
User enters a non-empty string as message	Success

Use Case Name	Receive messages from buyers and sellers
Requirement Number	6.3
Actors	Admin, Buyer/Seller
Parameters	None
Preconditions	None
Actions	1. The system gets a message to send from a buyer or seller 2. The system notifies the admin that a message is waiting 3. The admin opens the message and reads it
Result	A message is received and read by the admin
Data	**Result**
No internet connection	Fail
A buyer sends a non-empty message to admin	Success

Use Case Name	Get purchase history of member
Requirement Number	6.4
Actors	Admin, Member
Parameters	Member ID
Preconditions	1. Member exists
Actions	1. The admin chooses a member he wants to know the purchase history of 2. The system verifies the member exists 3. The system verifies the relevant user permissions 4. The system sends the admin the saved history of the member
Result	Member’s purchase history is provided to the admin
Data	**Result**
User enters a member ID that doesn’t exist	Fail
User enters a character that is not a number	Fail
User enters a member ID of an existing member	Success

Use Case Name	Get purchase history of a store
Requirement Number	6.4
Actors	Admin

Parameters	Store ID
Preconditions	1. Store exists
Actions	1. The admin chooses a store he wants to know the purchase history of 2. The system verifies the store exists 3. The system verifies the relevant user permissions 4. The system sends the admin the saved history of the store
Result	Store’s purchase history is provided to the admin

Data	**Result**
User enters a store ID that doesn’t exist	Fail
User enters a character that is not a number	Fail
User enters a store ID of an existing store	Success

Use Case Name	Get system information and statistics
Requirement Number	6.5
Actors	Admin
Parameters	None
Preconditions	None
Actions	1. The admin asks the system to get the system information 2. The system verifies the relevant user permissions 3. The admin chooses what kind of information / statistics he is interested in 4. The system provides the information to the admin
Result	Admin is presented with the information he requested

Data	**Result**
User (not an admin) tries to get system information	Fail
User (admin) tries to get system information	Success