I. System Use-Cases:

1. Use-Case: Initialize Marketplace System

1. Actor: System Admin
2. Preconditions: None
3. Parameters: Admin Credentials
4. Postconditions:
    1. 'System Admin' is initialized
    2. System has established connections with external services (payment, delivery)
5. Result: Marketplace system is initialized and ready for use
6. Actions:
    1. System Admin: Runs the marketplace process
    2. System: Registers a 'System Admin' profile
    3. System: Associates System Admin with 'System Admin' instance
    4. System: Establishes connections with payment and delivery services

2. Use-Case: Add connection with an external service

1. Actor: System Admin
2. Preconditions:
    1. Current marketplace state
    2. An active connection with another external service does not exist
3. Parameters:
    1. External Service
    2. All required parameters to establish connection with the service
4. Postconditions:
    1. Current marketplace state (i.e. state has not been altered)
    2. An active connection with the external service exists
5. Result: All traffic related to the external service is routed to it
6. Actions:
    1. System Admin: Specifies external service to connect to
    2. System Admin: Specifies relevant details to allow connection
    3. System: Establishes connection with appropriate external service

2.1. Use-Case: Edit connection with an external service

1. Actor: System Admin
2. Preconditions:
    1. Current marketplace state
    2. An active connection with the external service exists
3. Parameters: Modification details
4. Postconditions:
    1. Current marketplace state (i.e. state has not been altered)
    2. The connection with the external service has updated parameters
5. Result: All traffic related to the external service is routed to it according to the parameters specified
6. Actions:
    1. System Admin: Specifies modification details
    2. System: Forwards request to external service
    3. System: Updates external service state according to response

2.2. Use-Case: Swap connection with an external service

1. Actor: System Admin
2. Preconditions:
    1. Current marketplace state
    2. An active connection with an external service exists
3. Parameters:
    1. External Service
    2. All required parameters to establish connection with the service
4. Postconditions:
    1. Current marketplace state (i.e. state has not been altered)
    2. An active connection with the original external service does not exist
    3. An active connection with the new external service exists
5. Result: All traffic related to the external service is routed to the new service
6. Actions:
    1. System Admin: Specifies external service to connect to
    2. System Admin: Specifies relevant details to allow connection
    3. System: Disconnects from original external service
    4. System: Establishes connection with appropriate external service

3. Use-Case: Call Payment Service

1. Actor: System
2. Preconditions:
    1. A checkout operation has been performed by a user
    2. A connection with a payment service exists
3. Parameters: Order details (contains information regarding a specific transaction)
4. Postconditions: User's checkout has succeeded or failed
5. Result: Payment confirmation/refusal
6. Actions:
    1. System: Forwards order details to external service
    2. System: Receives external service response
    3. System: Returns response

4. Use-Case: Call Delivery Service

1. Actor: System
2. Preconditions:
    1. A checkout operation has been performed by a user
    2. A payment service has confirmed the transaction
    3. A connection with a delivery service exists
3. Parameters:
    1. Delivery details
    2. Client credentials
4. Postconditions: None
5. Result: Delivery request confirmation/refusal
6. Actions:
    1. System: Forwards order details to external service
    2. System: Receives external service response
    3. System: Returns response

5. Use-Case: Real-Time Notifications

1. Actor: System
2. Preconditions:
    1. One of the following conditions has been satisfied:
        - A client has purchased a product from a shop
        - A shop is closed
        - A shop is re-opened
        - A user nomination has been rescinded
        - A user received a message/inquiry
    2. Users are logged in
3. Parameters:
    1. Usernames
    2. Condition/Message
4. Postconditions: All users related to the satisfied conditions have a pending message
5. Result: None
6. Actions:
    1. System: Creates a message according to the satisfied condition
    2. System: Notifies all usernames a message is pending

6. Use-Case: Delayed Notifications

1. Actor: System
2. Preconditions: Users are logged out
3. Parameters:
    1. Usernames
    2. Condition/Message
4. Postconditions: Database contains messages destined for the specified users
5. Result: None
6. Actions:
    1. System: Creates a message according to the satisfied condition
    2. System: Stores all messages and their recipients' usernames

II. User Related Use-Cases:

Guest Use-Cases:

1. General Guest Use-Cases:

1. Use-Case: Access Marketplace

　1. Actor: User
　2. Preconditions: None
　3. Parameters: None
　4. Postconditions:
　　1. 'Guest' instance representing the user exists
　　2. 'Guest' instance has an empty shopping cart
　　3. 'Guest' instance is associated with the user
　5. Result: User can perform general and purchase related actions
　6. Actions:
　　1. System: Creates a new 'Guest' instance with an empty shopping cart
　　2. System: Presents to the user relevant guest actions and data

2. Use-Case: Exit Marketplace (Guest)

　1. Actor: User
　2. Preconditions: User has an existing active profile
　3. Parameters: Username
　4. Postconditions:
　　1. 'Shopping Cart' is emptied
　　2. 'Guest' instance is deleted
　5. Result: User can no longer perform any actions within the system
　6. Actions:
　　1. System: Empties the 'Shopping Cart'
　　2. System: Deletes the associated 'Guest' instance
　　3. System: Closes marketplace system instance

3. Use-Case: Register

1. Actor: User
2. Preconditions: 'Guest' instance associated with the user exists
3. Parameters: Identifying details (TBD Specifics)
4. Postconditions:
    1. New 'Member' instance exists
    2. The new 'Member' instance holds all identifying details given by the user
5. Result: A new 'Member' is added to the system
6. Actions:
    1. User: Inputs all relevant identifying details
    2. User: Confirms input
    3. System: Checks for data validity
        i. System: Finds that data is invalid
            a. System: Present error message
        ii. System: Finds that data is valid
            a. System: Create new 'Member' instance with the given identifying details

4. Use-Case: Login

1. Actor: User
2. Preconditions: 'Guest' instance associated with the user exists
3. Parameters:
    1. Username
    2. Password
4. Postconditions:
    1. User is identified as 'Member' with its associated details
    2. All delayed notifications have been presented to the user
5. Result: User can perform any member related operations
6. Actions:
    4. System: Initializes login process
    5. User: Inputs username
    6. User: Inputs password
    7. User: Confirms input
    8. System: Checks for data validity
        i. System: Finds that data is invalid
            a. System: Present error message
        ii. System: Finds that data is valid
            b. System: Associate user with appropriate 'Member' instance
            c. System: Displays any delayed notifications addressed at the user during his absence

2. Guest Payment Use-Cases:

1. Use-Case: Get Shop Info

1. Actor: User
2. Preconditions: User has an associated 'User' instance
3. Parameters: Shop ID
4. Postconditions: None
5. Result: Display relevant shop info, including products that the shop is offering
6. Actions:
    1. User: Requests shop details
    2. System: Searches for shop
    3. System: Finds that shop exists
        i. System: Displays relevant shop info
    4. System: Finds that shop doesn't exist
        i. System: Displays to user that shop wasn't found

2. Use-Case: Search Products

1. Actor: User
2. Preconditions: User has an associated 'User' instance
3. Parameters: Keywords and filters
4. Postconditions: None
5. Result: Products corresponding to the given parameters (TBD: Specific details to be displayed regarding said product)
6. Actions:
    1. System: Initialize search process
    2. User: Inputs keywords
    3. User: Inputs filters (Optional)
    4. User: Confirms input
    5. System: Searches according to the given parameters
    6. System: Displays the relevant products (or nothing if no products were found)

4.1. Use-Case: Add to shopping cart

1. Actor: User
2. Preconditions: User has an existing instance
3. Parameters:
    1. Username\Guest ID
    2. Product ID
    3. Shop ID
4. Postconditions: User's shopping cart contains the corresponding product
5. Result: None
6. Actions:
    1. User: Selects product to add to cart
    2. System: Adds product ID to the relevant shop's 'Shopping Bag'

4.2. Use-Case: Check Shopping Cart

1. Actor: User
2. Preconditions: User has an existing instance
3. Parameters: Username\Guest ID
4. Postconditions: None
5. Result: The products contained in the shopping cart (TBD: Specifics)
6. Actions:
    1. User: Requests shopping cart current product catalog
    2. System: Retrieves product specifications from each 'Shopping Bag'

4.3. Use-Case: Remove From Shopping Cart

1. Actor: User
2. Preconditions:
    1. User has an existing instance
    2. 'Shopping Cart' contains at least 1 product
3. Parameters:
    1. Username\Guest ID
    2. Product ID
    3. Shop ID
4. Postconditions: User's shopping cart does not contain the product
5. Result: None
6. Actions:
    1. User: Requests a product be removed from his shopping cart
    2. System: Removes product from the 'Shopping Bag' representing the shop ID

4.3. Use-Case: Edit Product Specifications In Shopping Cart

1. Actor: User
2. Preconditions:
    1. User has an existing instance
    2. 'Shopping Cart' contains at least 1 product
    3. Desired product quantity is within the shop's stock
3. Parameters:
    1. Username\Guest ID
    2. Product ID
    3. Shop ID
    4. Product Quantity
    5. Additional product modification details (TBD)
4. Postconditions: User's shopping cart's content reflects changes (TODO: Probably split to be more specific)
5. Result: None
6. Actions:
    1. User: Requests product modification from a product in shopping cart
    2. System: Checks that the product quantity does not exceed the shop's stock
    3. System: Modifies product according to request

5. Use-Case: Checkout

1. Actor: User
2. Preconditions: User has at least one product in shopping cart
3. Parameters:
    1. User's 'Shopping Cart'
    2. User credentials
    3. Delivery Details (optional)
4. Postconditions:
    1. 'Shopping Cart' is empty
    2. 'Call Delivery Service' is executed according to the given properties
    3. Order details are stored in the database
5. Result: None
6. Actions:
    1. User: Requests transaction finalization
    2. System: Creates an 'Order' with the given parameters and includes a timestamp
    3. System: Calls 'Payment Service' to confirm transaction validity
        i. System: If system receives a negative response from payment service, process is aborted and user is informed.
    4. System: Receives a positive response from the payment service and informs user that the transaction has been completed successfully
    5. System: Saves the order details in the database
    6. System: Asks user what form of delivery he would be interested in
    7. User: Inputs relevant delivery details
    8. System: Calls 'Delivery Service' to initiate product shipment

<u>**Member Use-Cases:**</u>

3. General Member Use-Cases:

0. Use-Case: Exit Marketplace (Member)

1. Actor: Member
2. Preconditions: User is <u>logged in</u>
3. Parameters: Username (TODO: liable to change)
4. Postconditions:
    1. User is <u>logged out</u>
    2. 'Shopping Cart', as well as other member specific details are preserved
5. Result: User is no longer able to perform marketplace related actions
6. Actions:
    1. User: Requests to leave the marketplace
    2. System: <u>Logs user out</u>
    3. System: Closes marketplace system instance

1. Use-Case: Logout

1. Actor: Member
2. Preconditions: User is <u>logged in</u>
3. Parameters: Username
4. Postconditions:
    1. User is not logged in
    2. 'Guest' instance representing the user exists
    3. 'Guest' instance has an empty shopping cart (TBD)
    4. 'Guest' instance is associated with the user
5. Result: User is associated with a 'Guest' instance
6. Actions:
    1. User: Requests to log out
    2. System: Marks associated 'Member' instance as logged out
    3. System: Creates a new 'Guest' instance with an empty shopping cart
    4. System: Presents to the user relevant guest actions and data

2. Use-Case: Set Up Shop

1. Actor: Member
2. Preconditions: User is logged in
3. Parameters: Username
4. Postconditions:
    1. A 'Shop' instance exists
    2. The 'Shop' instance is associated with the 'Member' as its founder
    3. The 'Shop' is active
5. Result: The user can now perform shop related actions as its founder
6. Actions:
    1. User: Requests to open a new shop
    2. System: Creates a new 'Shop' instance with the user as its founder and sets it as an active shop

4. Shop Owner Use-Cases:

1.1. Use-Case: Stock Management (Product Addition)

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner or manager with sufficient permissions
    3. Product does not exist in shop
    4. Product quantity is positive
3. Parameters:
    1. Username
    2. Shop ID
    3. Product ID
    4. Product quantity
4. Postconditions: Specified product is associated with the given shop
5. Result: None
6. Actions:
    1. User: Requests to add a product to the shop
    2. System: Adds the product to the shop with the specified quantity

1.2. Use-Case: Stock Management (Product Removal)

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner or manager with sufficient permissions
    3. Product exists in shop
3. Parameters:
    1. Username
    2. Shop ID
    3. Product ID
4. Postconditions: Product does not exist in shop
5. Result: None
6. Actions:
    7. User: Requests to remove a product from the shop
    8. System: Removes product association with the shop

1.3. Use-Case: Stock Management (Product Modification)

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner or manager with sufficient permissions
    3. Product exists in shop
    4. Product quantity is positive
3. Parameters:
    1. Username
    2. Shop ID
    3. Product ID
    4. Product quantity
4. Postconditions: The specified product's quantity is modified
5. Result: None
6. Actions:
    1. User: Requests to modify a product's quantity in the shop
    2. System: Adds the product to the shop with the specified quantity

2. Use-Case: Modify Shop's Purchase/Sale Types and Policies

1. Actor: Member
2. Preconditions:
   1. User is logged in
   2. User is a shop owner or manager with sufficient permissions
3. Parameters:
   1. Username
   2. Shop ID
   3. Policy details (TBD Specifics)
4. Postconditions: Shop policy is modified according to the details specified
5. Result: None
6. Actions:
   1. User: Requests to modify a shop's policies
   2. System: Shop policy is adjust according to the specified details

4. Use-Case: Appoint Shop Owner

1. Actor: Member
2. Preconditions:
   1. User is logged in
   2. User is a shop owner
   3. Appointed user is a member and not a shop owner
3. Parameters:
   1. Username
   2. Appointed member username
   3. Shop ID
4. Postconditions:
   1. Appointed member is a shop owner of the given shop ID
   2. The user is assigned as the appointed member's unique nominator
5. Result: Appointed user can now perform shop owner operations
6. Actions:
   1. User: Requests the nomination of a member to 'Shop Owner'
   2. System: Assigns member the 'Shop Owner' state of the shop
   3. System: Assigns the user as the member's unique nominator

6. Use-Case: Appoint Shop Manager

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner
    3. Appointed user is a member and not a shop owner or manager
3. Parameters:
    1. Username
    2. Appointed member username
    3. Shop ID
4. Postconditions:
    1. Appointed member is a shop manager of the given shop ID
    2. The user is assigned as the appointed member's unique nominator
5. Result: Appointed user can now perform shop manager operations
6. Actions:
    1. User: Requests the nomination of a member to 'Shop Manager'
    2. System: Assigns member the 'Shop Manager' state of the shop
    3. System: Assigns the user as the member's unique nominator

7.1. Use-Case: Add Shop Manager Permissions

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner
    3. Respective user is a shop manager of the shop
3. Parameters:
    1. Username
    2. Shop manager's username
    3. Shop ID
    4. Permissions
4. Postconditions: Shop manager has the specified permissions selected
5. Result: Shop manager can perform actions requiring specified permissions
6. Actions:
    1. User: Specifies shop to manage
    2. User: Specifies the shop manager to add permissions to
    3. User: Specifies permissions to add
    4. System: Modifies 'Shop Manager' state permissions

7.2. Use-Case: Remove Shop Manager Permissions

1. Actor: Member
2. Preconditions:
    1. User is logged in
    2. User is a shop owner
    3. Respective user is a shop manager
3. Parameters:
    1. Username
    2. Shop manager's username
    3. Shop ID
    4. Permissions
4. Postconditions: Shop manager cannot perform actions requiring specified permissions
5. Result: Shop manager cannot perform actions requiring specified permissions
6. Actions:
    1. User: Specifies shop to manage
    2. User: Specifies the shop manager to remove permissions from
    3. User: Specifies permissions to remove
    4. System: Modifies 'Shop Manager' state permissions

9. Use-Case: Close Shop

1. Actor: Shop Founder
2. Preconditions:
    1. User is logged in
    2. User is the shop founder
    3. Shop is open
3. Parameters:
    1. Username
    2. Shop ID
4. Postconditions:
    1. Shop status is inactive (regular members are unable to get information regarding the shop and its products)
    2. Existing shop owners and managers retain their status
5. Result: Shop owners and managers receive a notification regarding the action
6. Actions:
    1. User: Specifies shop to close
    2. System: Sets shop's status to inactive

11. Use-Case: Request Shop Personnel Info

    1. Actor: Member
    2. Preconditions:
        1. User is logged in
        2. User is a shop owner
    3. Parameters:
        1. Username
        2. Shop ID
    4. Postconditions: Shop manager cannot perform actions requiring specified permissions
    5. Result: The system displays information regarding the shop's personnel as well as the shop managers' permissions
    6. Actions:
        1. User: Specifies shop to inspect
        2. System: Retrieves list of shop managers and owners
        3. System: Retrieves list of personnel permissions

13. Use-Case: Get Shop Purchase History (Shop Owner)

    1. Actor: Member
    2. Preconditions:
        1. User is logged in
        2. User is a shop owner of the specified shop
    3. Parameters:
        1. Username
        2. Shop ID
        3. Time interval
        4. Filter details (optional)
    4. Postconditions: None
    5. Result: System displays product purchase history (retaining all original details)
    6. Actions:
        1. User: Specifies shop to inspect
        2. User: Requests purchase history
        3. User: Specifies time interval between which to search
        4. User: Specifies search filters
        5. System: Retrieves list of transactions

5. Shop Manager Use-Cases:

All operations according to given permissions

6. System Admin Use-Cases:

0. Use-Case: Register (Admin)

1. Actor: System Admin
2. Preconditions: None
3. Parameters:
   1. Username
   2. Password
   3. Identifying details (TBD Specifics)
4. Postconditions:
   1. New 'System Admin' instance exists associated with the given username
   2. The new 'System Admin' instance holds all identifying details given by the system admin
5. Result: A new 'System Admin' is added to the system
6. Actions:
   1. User: Inputs all relevant identifying details
   2. User: Confirms input
   3. System: Checks for data validity
      i. System: Finds that data is invalid
         a. System: Present error message
      ii. System: Finds that data is valid
         a. System: Create new 'Member' instance with the given identifying details

4. Use-Case: Get Shop Purchase History (Admin)

1. Actor: System Admin
2. Preconditions:
   1. User is logged in
   2. User is a 'System Admin'
3. Parameters:
   1. Username
   2. Shop ID
   3. Time interval
   4. Filter details (optional)
4. Postconditions: None
5. Result: System displays product purchase history (retaining all original details)
6. Actions:
   1. User: Specifies shop to inspect
   2. User: Requests purchase history
   3. User: Specifies time interval between which to search
   4. User: Specifies search filters
   5. System: Retrieves list of transactions