Table of Contents

[Glossary 2](#_Toc39507284)

[Initialize System: 1.1 4](#_Toc39507285)

[Register: 2.2 5](#_Toc39507286)

[Login: 2.3 6](#_Toc39507287)

[View info: 2.4 7](#_Toc39507288)

[Search items: 2.5 8](#_Toc39507289)

[Add to shopping basket: 2.6 9](#_Toc39507290)

[View and edit shopping cart: 2.7 10](#_Toc39507291)

[Purchase cart: 2.8 11](#_Toc39507292)

[Log out: 3.1 14](#_Toc39507293)

[Open store: 3.2 14](#_Toc39507294)

[User purchase history: 3.7 15](#_Toc39507295)

[Manage supply: 4.1, 5.1 16](#_Toc39507296)

[Add store owner: 4.3 17](#_Toc39507297)

[Add store manager: 4.5 18](#_Toc39507298)

[Edit permissions: 4.6 19](#_Toc39507299)

[Remove manager: 4.7 20](#_Toc39507300)

[Store purchase history: 4.10, 5.1 21](#_Toc39507301)

[View Store purchase history as Admin: 6.4 22](#_Toc39507302)

[View user purchase history as Admin: 6.4 22](#_Toc39507303)

**Glossary:**

**Buyer-** a subscribed user that purchases a product from a specific store.

**Category-** A group of values that items divided by.

**Discount-** an option for the buyer to purchase a product at reduced cost on specific terms specified by one of the store owners.

**Discount policy-** describes the discount types that the store allows.

**Discount type-** a specific way in which a discount can be given to a product.

**Password-** a combination of characters linked to a subscribed user that is chosen at the process of registration by the user and required for logging in to the system.

**Payment Collection-** an external system responsible for collecting the payment for all purchases made in the system.

**Permission-** describes a privilege a user can have to a certain store, like update inventory.

**Product-** an item or a service that has different attributes and a price.

**Product item-** an item in the shopping cart, including the amount to purchase.

**Product info-** information about the product- name, price, category and description.

**Purchase-** the act of buying a product sold by a specific store.

**Purchase policy-** describes the purchase types that the store allows.

**Purchase type-** a specific way that users can buy according to.

**Search filter-** properties allowing to filter a list of products according to price range and category.

**Security system-** an external system providing password management and encryption for the trading system.

**Seller-** a subscribed user that offers products for sale in a specific store.

**Shopping basket-** a list of products selected by a specific buyer in a specific store.

**Shopping cart-** a list of products selected by a specific buyer from all the stores.

**Store-** contains products that have different properties and identifications.

**Store info-** information about the store- name, owner name and description.

**Store inventory-** specifies the list of products and their amount available for purchase at the store.

**Store manager-** a subscribed user that has been appointed by a store owner to manage his store. A store manager has specific managing permissions given to him by the store owner.

**Store managing-** describes a the management of user managing a specific store.

**Store owner-** a subscribed user that opened the store or was appointed as owner by another store owner.

**Store owning-** describes an ownership of an owner of a specific store.

**Store purchase history-** A list of all the previous purchases made in the store.

**Subscribed user-** a user that is registered to the system.

**Subscribed user purchase history-** A list of all the previous purchases made by the subscribed user.

**Supply Product-** an external system responsible for suppling the products to the costumers for all purchases made in the store.

**System manager-** a subscribed user whose job is to address buyers and sellers’ requests, maintain fair and legal commerce through the system and ensure proper system operation.

**Trading-** purchasing or offering a product for sale.

**Trading system-** a system that enables operating a trading network between sellers and buyers. The system consists of a collection of stores.

**User-** An individual operating the system for any reason in order to accomplish a task.

**Username-** a unique identification for a subscribed user that is chosen at the process of registration by the user and required for logging in to the system.

**Visitor-** a user that visit the system but isn't registered in it.

Initialize System:

Actor: None

Preconditions: System has not been initialized yet

Parameters: None

Actions:

1. System initializes an empty set of registered users.
2. System initializes an empty set of registered stores.
3. System sets-up connection with the Security system.
4. System initializes an empty set of system managers.
5. System creates a new system manager and adds him to the system managers set.
6. System sends the new system manager’s username and password to the security system.
7. System sets active user to be a gust user.
8. System sets-up connection with the Payment Collection system.
9. System sets-up connection with the Product Supply system.

Register:

Actor: User

Preconditions: User is not logged in

Parameters: Username, password

Actions:

1. User sends the system a register command.
2. User sends the username and password to the system.
3. System checks if the username is empty.
4. If empty:
   1. System sends the following message: "Username cannot be empty"
5. If not empty:
   1. System checks if the username already exists.
   2. If exists:
      1. System sends the following failure message: "This username already exists in the system. Please choose a different one".
   3. If doesn't exists:
      1. System checks if the password is valid.
      2. If not valid:
         1. The system sends the following failure message: "This password is not valid. Please choose a different one".
      3. If valid:
         1. The system registers the user.
         2. The system sends the user the following message "You have been successfully registered!"

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Username doesn't exist, valid password | "You have been successfully registered!" | Good |
| 2 | Username doesn't exist, invalid password | "This password is not valid. Please choose a different one" | Sad |
| 3 | Username exists, valid password | "This username already exists in the system. Please choose a different one" | Sad |
| 4 | Empty username | "Username cannot be empty" | Bad |

Login:

Actor: User

Preconditions: User is not logged in

Parameters: Username, password, adminMood

Actions:

1. User sends the system a login command.
2. User sends the username and password to the system.
3. System checks if the username is empty.
4. If empty:
   1. System sends the following message: "The username is invalid"
5. If not empty:
   1. System checks if the username is registered.
   2. If not registered:
      1. System sends the following failure message: "This user is not registered".
   3. If registered:
      1. System checks if the password matches the username.
      2. If doesn't match:
         1. System sends the following failure message: "This password is incorrect".
      3. If match:
         1. System check if the adminMood true.
         2. If true:
            1. System checks if the user in the admins list.
            2. If in the list:

The user is logged in to the system as Admin.

System sends the following message: "You have been successfully logged in!"

* + - * 1. If not in the list:

System sends the following failure message: "this user is not a system admin"

* + - 1. If false:
         1. The user is logged in to the system.
         2. System sends the following message: "You have been successfully logged in!"

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Username exists, password matches | "You have been successfully logged in!" | Good |
| 2 | Username exists, password doesn't match | "This password is incorrect" | Sad |
| 3 | Username doesn't exist | "This user is not registered" | Sad |
| 4 | adminMode true but username is not admin | "this user is not a system admin" | Sad |
| 5 | Empty username | "The username is invalid" | Bad |

View info:

Actor: User

Preconditions: none

Parameters: store name, product name

Actions:

1. User sends view store info command.
2. User sends the store name.
3. System checks if the store name is empty.
4. If empty:
   1. System sends the following message: "The store name is invalid".
5. If not empty:
   1. System checks if the store exists.
   2. If doesn't exists:
      1. System sends the following message: "This store doesn't exist in this trading system".
   3. If exists:
      1. System sends store info and a list of product's names available in the store.
      2. User sends view product info command.
      3. User sends product name.
      4. System checks if the product name is empty.
      5. If empty:
         1. System sends the following message: "The product name is invalid".
      6. If not empty:
         1. System checks if the product name appears in the list of available store products.
         2. If doesn't appears:
            1. System sends the following message: "This product is not available for purchasing in this store".
         3. If appears:
            1. System sends product info.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Store name exists, product exist in the store | Store & product info | Good |
| 2 | Store name doesn't exists | "This store doesn't exist in this trading system" | Sad |
| 3 | Store name empty | "The store name is invalid" | Bad |
| 4 | Store name exists, product doesn't exist | "This product is not available for purchasing in this store" | Bad |
| 5 | Store name exists, empty product name | "The product name is invalid" | Bad |

Search items:

Actor: User

Preconditions: none

Parameter: product name/ product category/ description key words, search filter

Actions:

1. User sends the system a search product command.
2. User sends product name/ product category/ description key word to the system.
3. System checks if the user's input is not empty.
4. If empty:
   1. System sends the following message: "Must enter search parameter".
5. If not empty:
   1. System searches its shops for products that match the input.
   2. If no items were found:
      1. System sends the following message: "There are no products that match these parameters".
   3. If at least one item was found:
      1. System sends the user a list of all the matching products.
      2. User sends to the system a filter results command.
      3. User sends the search filter to the system.
      4. System searches the item list for products that match the search filter.
      5. If no items were found:
         1. System sends the following message: "There are no products that match this search filter".
      6. If at least one item was found:
         1. System sends the user a list of all the matching products.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Product name/ product category/ description key word with matching products,  search filter with matching products | List of matching products | Good |
| 2 | Product name/ product category/ description key word with no matching products | "There are no products that match these parameters" | Sad |
| 3 | Product name/ product category/ description key word with matching products,  search filter with no matching products | "There are no products that match this search filter" | Sad |
| 4 | Empty input | "Must enter search parameter" | Bad |

Add to shopping basket:

Actor: User

Preconditions: none

Parameter: store name, product name, amount

Actions:

1. User sends add to shopping basket command.
2. User sends store name and product name.
3. System checks if the store name or product name are empty or if the amount <= 0.
4. If true:
   1. System sends the following message: "Must enter store name and product name and amount bigger than 0".
5. If false:
   1. System checks if the store exists in the trading system.
   2. If doesn't exist:
      1. System sends the following message: "The store doesn't exist in the trading system".
   3. If exist:
      1. System checks if the product is available in the store with the requested amount.
      2. If not available:
         1. System sends the following message: "The product isn't available in the store with the requested amount".
      3. If available:
         1. System adds the product to the user's shopping basket.
         2. System sends the following message: "Items have been added to basket".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Store name exists, product available in the store | "Items have been added to basket" | Good |
| 2 | Store name exists, product isn't available in the store | "The product isn't available in the store with the requested amount" | Sad |
| 3 | Store name doesn't exist | "The store doesn't exist in the trading system" | Bad |
| 4 | Empty store name or product or amount <= 0 | "Must enter store name and product name and amount bigger than 0" | Bad |

View and edit shopping cart:

Actor: User

Preconditions: none

Parameters: product name, store name, amount

Actions:

1. User sends the system view shopping cart command.
2. System sends the user a list of the products in his shopping cart.
3. User sends the system edit shopping cart command.
4. User sends the system product name, store name and amount.
5. System checks if the products name, store name or amount is empty.
6. If empty:
   1. System sends the following message: "Must enter store name and product name and amount bigger than 0".
7. If not empty:
   1. System checks if the store exist.
   2. If doesn't exists:
      1. System sends the following message: "This store doesn't exist".
   3. If exists:
      1. System checks if the product appears in the list of products.
      2. If doesn't appears:
         1. System sends the following message: "The product doesn’t exist in your shopping cart".
      3. If appears:
         1. System checks if this amount is available in the in the inventory.
         2. If available:
            1. System updates the amount accordingly.
            2. System sends the following message: "The product has been updated successfully".
         3. If not available:
            1. System sends the following message: " The product isn't available in the store with the requested amount".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Product name exist in shopping cart, store exists | "The product has been updated successfully" | Good |
| 2 | Product name doesn't exist in shopping cart | "The product doesn’t exist in your shopping cart" | Sad |
| 3 | Store doesn't exists | "This store doesn't exist" | Sad |
| 4 | Amount is not available | "The product isn't available in the store with the requested amount". | Sad |
| 5 | Empty product name, store name or amount | "Must enter store name and product name and amount bigger than 0" | Bad |

Purchase cart:

Actor: User

Preconditions: None

Parameters: None

Actions: described by a system sequence diagram

A close up of a piece of paper

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Cart is not empty, products exists in the stores inventory’s | "Purchasing completed successfully" | Good |
| 2 | Empty cart | "The cart is empty" | Sad |
| 3 | Product doesn’t exist in store inventory | "There is currently no stock of %amount %productname products" | Sad |
| 4 | Cart is not empty, products exists in the stores inventory’s | "Purchasing completed successfully" | Good |

**Reserve products:**

Actor: User

Preconditions: None

Parameters: None

Actions:

1. System goes over all the baskets in the cart, foreach basket:
   1. System goes over all the products in the basket, for each product:
2. System checks if the product is available with the requested amount.
3. If not available:
   * + 1. System unreserves all previously reserved products.
       2. System sends the user the following message: “There is currently no stock of %amount %productname products”.
       3. System stopes the iteration and returns
4. If available:
   * + 1. System reserve the product.

**Compute price:**

Actor: User

Preconditions: None

Parameters: None

Actions:

1. System adds the price of all the items in the shopping cart
2. System calculates the discount (Use Case Calculate discount)
3. System reduces the discount from the total price
4. System checks the purchase policy (Use Case Check policy)

**Calculate discount:**

Actor: User

Preconditions: None

Parameters: None

Actions:

1. System checks for each item in the cart if there is an active discount
2. If there is a discount:
   1. System checks if the discount terms apply
   2. If true:
      1. System adds the discount to the total discount

**Check policy:**

Actor: User

Preconditions: None

Parameters: None

Actions:

1. System checks for each item in the cart if the purchase policy matches the store policy
2. If false:
   1. System removes the item from the cart
   2. Systems sends the following message: “Your purchase doesn’t match the store’s policy”

**Payment:**

Actor: User, PaymentCollection

Preconditions: None

Parameters: None

Actions:

1. System sends the PaymentCollection the total price calculated
2. PaymentCollection asks the user for payment information
3. PaymentCollection sends the System if the payment was made successfully

**Save history:**

Actor: User

Preconditions: None

Parameters: None

Actions:

1. Systems saves the purchase to the user’s purchase history
2. For each store in the purchase:
   1. System saves the purchase to the store’s purchase history

Log out:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: None

Actions:

1. User sends the system a logout commend.
2. User is logged out of the system.
3. System sends the following message: "You have been successfully logged out!"

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | ------- | "You have been successfully logged out!" | Good |

Open store:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Store name, store description

Actions:

1. User sends open store command.
2. User sends the system store name and description.
3. System checks if store name or description is empty.
4. If empty:
   1. System sends the following message: "Must enter store name and description".
5. If not empty:
   1. System checks if a store with the same name already exists.
   2. If exists:
      1. System sends the following message: "Store name already exists, please choose a different one".
   3. If doesn't exist:
      1. System creates the store.
      2. System makes the user store owner.
      3. System sends the following message: "The new store is now open!"

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Store name doesn't exists, store description | "The new store is now open!" | Good |
| 2 | Store name exist | "Store name already exists, please choose a different one" | Sad |
| 3 | Empty store name or description | "Must enter store name and description" | Bad |

User purchase history:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: none

Actions:

1. User sends user purchase history command.
2. System sends the user's purchasing history.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | ------- | User purchase history | Good |

Manage supply:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Store name, product info, amount

Actions:

1. User sends update inventory command.
2. User sends store name, product info and amount.
3. System checks if store name or product info is empty or if the amount <= 0.
4. If true:
   1. System sends the following message: "Must enter store name, product info, and amount that is bigger than 0".
5. If false:
   1. System checks if the store exists.
   2. If doesn't exists:
      1. System sends the following message: "This store doesn't exist".
   3. If exists:
      1. System checks if the user has inventory editing privileges to the store.
      2. If doesn't have privileges:
         1. System sends the following message: "Must have editing privileges".
      3. If have privileges:
         1. System checks if the product already exists in the store inventory.
         2. If doesn't exists:
            1. System adds the product with the given info and amount.
            2. System sends the following message: "The product has been added"
         3. If exists:
            1. System adds the amount to the given product amount.
            2. System sends the following message: "The product has been updated".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Store exists, product exists | "The product has been updated" | Good |
| 2 | Store exists, product doesn't exists | "The product has been added" | Good |
| 3 | Store doesn't exists | "This store doesn't exist" | Sad |
| 4 | Sore exists, user doesn't have privileges | "Must have editing privileges" | Sad |
| 5 | Empty store name or product info, or amount smaller than 0 | "Must enter store name, product info, and amount that is bigger than 0" | Bad |

Add store owner:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Username, store name

Actions:

1. User sends add store owner command.
2. User sends username and store name.
3. System checks if the username or store name is empty.
4. If empty:
   1. System sends the following message: "Must enter username and store name".
5. If not empty:
   1. System checks if store name exists:
      1. If doesn't exist:
         1. System sends the following message: "This store doesn't exist".
      2. If exist:
         1. System checks if the username exists
         2. If doesn't exists:
            1. System sends the following message: "This username doesn't exist".
         3. If exists:
            1. System checks if the user is one of the store owners.
            2. If isn't owner:

System sends the following message: "You must be a store owner for this action".

* + - * 1. If is owner:

System checks if the username is already one of the store's owner.

If is already store owner:

System sends the following message: "This username is already one of the store's owners".

If isn't store owner:

System adds the username as a store owner.

System sends the following message: "Username has been added as one of the store owners successfully".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Username exists, store name exists | "Username has been added as one of the store owners successfully" | Good |
| 2 | Username doesn’t exists | "This username doesn't exist" | Sad |
| 3 | Store doesn't exists | "This store doesn't exist" | Sad |
| 4 | Username exists, store name exists, user isn't a store owner | "You must be a store owner for this action" | Sad |
| 5 | Empty username or store name | "Must enter username and store name" | Bad |

Add store manager:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Username, store name

Actions:

1. User sends add store manager command.
2. User sends username and store name.
3. System checks if the username or store name is empty.
4. If empty:
   1. System sends the following message: "Must enter username and store name".
5. If not empty:
   1. System checks if store name exists:
      1. If doesn't exist:
         1. System sends the following message: "This store doesn't exist".
      2. If exist:
         1. System checks if the username exists
         2. If doesn't exists:
            1. System sends the following message: "This username doesn't exist".
         3. If exists:
            1. System checks if the user is one of the store owners.
            2. If isn't owner:

System sends the following message: "You must be a store owner for this action".

* + - * 1. If is owner:

System checks if the username is already one of the store's managers.

If is already store manager:

System sends the following message: "This username is already one of the store's managers".

If isn't store manager:

System adds the username as a store manager.

System sends the following message: "Username has been added as one of the store managers successfully".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Username exists, store name exists | "Username has been added as one of the store managers successfully" | Good |
| 2 | Username doesn’t exists | "This username doesn't exist" | Sad |
| 3 | Store doesn't exists | "This store doesn't exist" | Sad |
| 4 | Username exists, store name exists, user isn't a store owner | "You must be a store owner for this action" | Sad |
| 6 | User is already manager | "This username is already one of the store's managers" | Sad |
| 5 | Empty username or store name | "Must enter username and store name" | Bad |

Edit permissions:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Username, permissions list, store name

Actions:

1. User sends the system edit permissions commend.
2. User sends username, permissions list and store name.
3. System checks if the username permissions list or store name is empty.
4. If empty:
   1. System sends the following message: "Must enter username, permissions list and store name".
5. If not empty:
   1. System checks if the store exists.
   2. If doesn't exists:
      1. System sends the following message: "This store doesn't exist".
   3. If exists:
      1. System checks if the username exists.
      2. If doesn't exists:
         1. System sends the following message: "This username doesn't exist".
      3. If exists:
         1. System checks if the active user is one of the store's owners.
         2. If false:
            1. System sends the following message: "You must be this store owner for this command".
         3. If true:
            1. System checks if the username is one of the store's managers appointed by the user.
            2. If false:
6. System sends the following message: "You can't edit this user's privileges".
   * + - 1. If true:
7. System updates the user's privileges.
8. System sends the following message: "Privileges have been edited successfully".

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Username exists, store name exists | "Privileges have been edited successfully" | Good |
| 2 | Username doesn’t exists | "This username doesn't exist" | Sad |
| 3 | Store name doesn't exists | " This store doesn't exists" | Sad |
| 4 | Username isn't store manager appointed by the user | "You can't edit this user's privileges" | Sad |
| 5 | Empty username, store name or permissions list | "Must enter username, permissions list and store name" | Bad |

Remove manager:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Username, store name

Actions:

1. User sends the system remove manager commend.
2. User sends username and store name.
3. System checks if the username or store name is empty.
4. If empty:
   1. System sends the following message: "Must enter username and store name".
5. If not empty:
   1. System checks if the store exists.
   2. If doesn't exists:
      1. System sends the following message: "This store doesn't exist".
   3. If exists:
      1. System checks if the username exists.
      2. If doesn't exists:
         1. System sends the following message: "This username doesn't exist".
      3. If exists:
         1. System checks if the user is one of the store owner.
         2. If false:
            1. System sends the following message: "You must be this store owner for this command".
         3. If true:
            1. System checks if the username is one of the store's managers appointed by the user.
            2. If false:

System sends the following message: "This username is not one of this store's managers appointed by you".

* + - * 1. If true:

System removes the username as one of the store's managers.

System sends the following message: "Manager removed successfully!"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Input | Expected Output | | Type |
| 1 | Username is manager of the store appointed by the user, store | "Manager removed successfully!" | | Good |
| 2 | Store name doesn't exist | | "This store doesn't exist" | Sad |
| 3 | Username doesn't exist | | "This username doesn't exist" | Sad |
| 4 | User is not the store owner | | "You must be this store owner for this command" | Sad |
| 5 | Username is not one of the store's manager appointed by the user | | "This username is not one of this store's managers appointed by you" | Sad |
| 6 | Empty username or store name | | "Must enter username and store name" | Bad |

Store purchase history:

Actor: Subscribed user

Preconditions: User is logged in

Parameters: Store name

Actions:

1. User sends store purchase history command.
2. User sends store name.
3. System checks if the store name is empty.
4. If empty:
   1. System sends the following message: "Must enter store name".
5. If not empty:
   1. System checks if the store exists.
   2. If doesn't exist:
      1. System sends the following message: "This store doesn't exist".
   3. If exist:
      1. System checks if the user is one of the store owner or one of the store's manager with view purchasing history permissions.
      2. If false:
         1. System sends the following message: "You are not allowed to view this store's purchasing history".
      3. If true:
         1. System sends purchasing history.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Existing store name, user with permissions. | Purchasing history | Good |
| 2 | Store name doesn’t exist | "This store doesn't exist" | Sad |
| 3 | User without permissions | "You are not allowed to view this store's purchasing history" | Sad |
| 4 | Empty store name | "Must enter store name" | Bad |

View Store purchase history as Admin:

Actor: Subscribed admin user   
Preconditions: User is logged in in admin mode  
Parameters: Store name

Actions:

1. User sends store purchase history command.
2. User sends store name.
3. System checks if the store name is empty.
4. If empty:
   1. System sends the following message: "Must enter store name".
5. If not empty:
   1. System checks if the store exists.
   2. If doesn't exist:
      1. System sends the following message: "This store doesn't exist".
   3. If exist:
      1. System sends purchasing history.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Existing store name, user logged in as admin | Purchasing history | Good |
| 2 | Store name doesn’t exist | "This store doesn't exist" | Sad |
| 3 | User not in admin mode | "You are not allowed to view this store's purchasing history" | Sad |
| 4 | Empty store name | "Must enter store name" | Bad |

View user purchase history as Admin:

Actor: Subscribed admin user   
Preconditions: User is logged in in admin mode  
Parameters: Store name

Actions:

1. User sends user purchase history command.
2. User sends username.
3. System checks if the username is empty.
4. If empty:
   1. System sends the following message: "Must enter username".
5. If not empty:
   1. System checks if the user exists.
   2. If doesn't exist:
      1. System sends the following message: "This user doesn't exist".
   3. If exist:
      1. System sends purchasing history.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Input | Expected Output | Type |
| 1 | Existing username, user logged in as admin | Purchasing history | Good |
| 2 | username doesn’t exist | "This user doesn't exist" | Sad |
| 3 | User not in admin mode | "You are not allowed to view this user's purchasing history" | Sad |
| 4 | Empty username | "Must enter username" | Bad |