**Use Cases**

**System Use Cases:**

Initialize a Market:

Actor: user

Pre-Condition: None

Parameters:

* user\_details
* market\_details

Actions:

1. System request for user\_details and market\_details.
2. creates a new market and set the user as System manager.
3. System presents all existing service options.
4. User choose external services.
5. System shows the user whether the actions succeed or not.
6. System If action succeed - shows the current Market details.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | user | user |
| Parameters | good user details and good market details | bad user details or/and bad market details |
| Result | the market is loaded | the system displays an error message and asks for new details |

Change market connection to External Service:

Actor: System-Manager

Pre-Condition:

* user must be logged in.
* market must be initialized.
* all uses of the service must be finished.
* The new External Service must implement the corresponding Service Interface.

Parameters: new\_external\_service

Actions:

1. System checks for existing ongoing use of the current service.
2. if found – waits for all existing uses of the service to be done.
3. System ask the user for new the service
4. User inserts new\_external\_service.
5. System change the service implementor to new\_external\_service.
6. shows approving message to user.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | System manager | Market manager |
| Parameters | Good external service’s details | Bad external service’s details – service which doesn’t exist |
| Result | The system is connected to the new external service | the system displays an error message that the service doesn’t exist |

Payment:

Actor: user/ time

Pre-Condition:

* shoppingCart must not be empty.
* `` of any kind has been done

Finish Condition:

* result message has been received from the external service.

Parameters:

* payment\_details
* shopping\_cart
* user\_details

Actions:

1. System transfer request to the current payment service
2. Waits for service to finish and receives response res
3. If res is positive:
   1. transfer the shopping\_cart to the supply service (please lookout for Supply use case) with user\_details.
   2. cleans the shopping\_cart.
   3. Sends notification to buyer.
4. If res is negative, tell shows the user the error message.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Good payment and user details and good shopping cart | Illegal payment details |
| Result | Payment succeeds | the system displays an error message about the payment details |

Supply:

Actor: system

Pre-Condition:

* Acquisition of any kind has been completed

Parameters:

* shopping\_cart
* user\_details

Actions:

1. System transfers supply request including shopping\_cart and user\_details to current supply service.
2. Sends notifications to user.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | System | System |
| Parameters | Good shopping cart and user details | Shopping cart which doesn’t match the user details |
| Result | Supply passes to be the responsibility of the supply system | the system displays an error message about the unmatched shopping cart |

Shop manager live Notification for acquisition:

Actor: System

Pre-Condition:

* Acquisition made.
* Existing shop

Parameters:

* message

Actions:

1. for all listener in shop’s acquisition listeners:
   1. notify listener with message.
   2. if listener is logged in:
      1. pops (prints) new message.
   3. else:
      1. listener adds message to its notifications box.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | System | System |
| Parameters | Good message | Null |
| Result | Message is shown/saved | the system displays an error message about the message |

Offline Notification for account:

Actor: System

Pre-Condition:

* notification has been triggered
* existing user
* user is logged off

Parameters:

* message

Actions:

1. notify user with message
2. user adds notification to its notification box
3. User logging in
4. System shows all notification its notification box, including previous message

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | System | System |
| Parameters | Good message | Null |
| Result | Message is saved and displayed when user is logged in | the system displays an error message about the message |

**User Use Cases:**

Guest Log in:

Actor: user

Pre-Condition:

* Existing Market.

Parameters: None

Actions:

1. User starts using system
2. System creates new temporary user (Guest):
   1. System creates new shopping cart
3. System shows all guest options

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | ---- | ---- |
| Result | ---- | ---- |

User leaves the Market:

Actor: user

Pre-Condition:

* Existing Market.

Parameters: user\_id

Actions:

1. User chooses option “logout” or just closes the application.
2. If User is guest:
   1. System removes the user’s shopping cart.
   2. System removes temporary user (please lookout for remove user use case).
3. User logs out

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Logged in user\_id | Illegal user\_id |
| Result | User is logged out or deleted | the system displays an error message about the user\_id |

Register:

Actor: user

Pre-Condition:

* Existing Market.
* User logged in as guest

Parameters: user details (name, password, etc.)

Actions:

1. User chooses to register
2. System request user for user details
3. System creates new user:
   1. Creates new and empty shopping cart.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Legal user details | Name which is already exists in the system |
| Result | User created successfully | the system displays an error message about the existing name |

Login:

Actor: user

Pre-Condition:

* guest

Parameters:

* Name
* Password
* Additional user’s logging options

Actions:

1. User choose to login.
2. System asks for name, password, and user’s additional logging options
3. System validates each of the above.
4. If all correct:
   1. System imports the corresponding user.
   2. Changes the user’s shopping cart
5. Else:
   1. System notifies the user for invalid answers.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Legal user log in details | Password which unmatched the name |
| Result | User logged in successfully | the system displays an error message about the unmatching password |

Receive information of a shop:

Actor: user

Pre-Condition:

* Existing shop

Parameters:

* *shop\_id*

Actions:

1. User chooses a *shop*
2. System presents *shop’s* option
3. User chooses “show info”
4. System prints shop’s info

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Legal shop identifier | Shop which doesn’t exist |
| Result | Shop information is displayed | the system displays an error message about the shop which doesn’t exist |

Product Search:

Actor: user

Pre-Condition:

* At least 1 Existing Shop
* Existing product

Parameters:

* *Search\_text*

Actions:

1. User chooses search option
2. System request what kind of search required:
   1. “by category”
   2. “by name”
   3. “by tag”
3. User chooses *option*
4. System ask for *seach\_text*
5. System search all product by *option* and *search\_text*.
6. System shows all results

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Legal *search\_text* | Search by name which doesn’t exist in system |
| Result | Results of the search is displayed | the system displays an error message about the name which doesn’t exist |

Save Products in Shopping Cart:

Actor: user

Pre-Condition:

* Existing user (can be temporary)
* Existing Product
* Existing Shop

Parameters:

* Product\_id
* amount
* shop\_id

Actions:

1. User chooses “save”.
2. User inserts amount
3. If user does not have product list for the shop:
   1. System initializes new list for shop
4. System adds the product with its amount to user’s shopping basket.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | User |
| Parameters | Legal details | Negative amount |
| Result | The product is added to the shopping basket of the shop | the system displays an error message about the negative amount |

Show and change Shopping Cart:

Actor: user

Pre-Condition:

* Existing user (can be guest)

Parameters:

* *product*

Actions:

1. User choose to edit his\her shopping cart.
2. System prints shopping cart.
3. User choose *product* to edit.
4. System show *product* edit options.
5. User choose an option and edits.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | user | user |
| Parameters | Details of a valid product. | Details of a product that does not exist. |
| Result | The product is changed. | the system displays an error message and asks for new details. |

Acquisition:

Actor: user

Pre-Condition:

* visitor

Parameters:

* *shopping\_cart*

Actions:

1. Visitor chooses to buy *shopping\_cart*
2. System applies payment (please lookout for System’s Payment use case).
3. Supply service returns *supply\_details*
4. System notifies all shops in *shopping\_car* of its bought products.
5. System cleans user’s shopping cart.
6. If user is logged in:
   1. System print to user the *supply\_details*
7. Else:
   1. System notify user with *supply\_details (please lookout for user’s Notification’s use case).*

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | user | user |
| Parameters | Valid shopping cart | An empty shopping cart |
| Result | the Acquisition has been done. | the system displays an error message and asks for new details. |

**Shop Owner Cases:**

Stock Management:

Actor: Shop Owner

Pre-Condition:

* Existing shop.
* User must be logged in.
* User must be the shop owner.

Parameters: *product\_price.*

Actions:

1. User Choose edit products.
2. System present options:
   1. “add new product”
   2. “remove existing product”
   3. “edit existing product”
3. System shows all existing products.
4. User chooses a product.
5. System shows product’s details
6. User choose details to edit (price)
7. user inserts *product\_price*
8. System changes product’s price

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | Valid product price. | Negative product price. |
| Result | Product has been updated. | change is not done. The system displays an error message and asks for new details. |

Change Shop’s Policy:

Actor: Shop Owner

Pre-Condition:

* Existing shop.
* User must be logged in.
* User must be the shop owner.

Parameters: *discount\_policy id*

Actions:

1. User Choose edit policy.
2. System present options
3. User chooses to add new discount
4. System presents discount policy options
5. User chooses *option*
6. User inserts *discount\_policy.*
7. System adds policy to shop’s discount list.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | Valid discount policy id | Invalid *discount\_policy id* |
| Result | The shop’s policy is changed. | The shop’s policy is not changed. An error message is displayed. |

Shop Owner Appointment:

Actor: Shop Owner

Pre-Condition:

* Existing shop.
* User must be logged in.
* User must be the shop owner.
* *User2* - Another Existing user
* *User2* – not a shop owner yet

Parameters: *user2*

Actions:

1. User choose to add new shop owner
2. System requests for user info.
3. User inserts user2
4. System creates new Shop Owner (state)
5. If user2 is in shop’s employees list:
   1. Changes user2 role (State) to shop owner
6. Else:
   1. Adds user2 to shop employees list with shop owner as role.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | User that is a member. | A user that is not a member. |
| Result | User2’s role is changed. | User2’s role is not changed. the system displays an error message and asks for new details. |

Shop Manager Appointment:

Actor: Shop Owner

Pre-Condition:

* Existing shop.
* User must be logged in.
* User must be the shop owner.
* *User2* - Another Existing user
* *User2* – not a shop manager or owner yet

Parameters: *user2*

Actions:

1. User choose to add new shop manager.
2. System requests for user info.
3. User inserts *user2.*
4. System creates new shop\_manager.
5. Systems sets shop\_manager permissions to have:
   1. Receive information and response.
   2. Acquisition History
6. If *user2* is in shop’s employees list:
   1. Changes *user2* role (State) to shop manager.
7. Else:
   1. Adds *user2* to shop employees list with shop manger as role.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | User that is a member. | User that is not a member. |
| Result | The shop owner becomes a shop manager sucssesfuly. | The shop owner stays a shop owner. The system displays an error message. |

Edit Manager Permission:

Actor: shop owner

Pre-Condition:

* Existing shop.
* User must be logged in.
* User must be the shop owner.
* *User2* – existing shop manager

Parameters:

* *Permission*
* *User2*

Actions:

1. User choose to edit manger permission
2. System request for user details
3. User inserts *user2*
4. System present edit options
5. User choose to delete permission
6. System shows all existing premissions
7. User choose *permission* to remove.
8. System removes permission from *user2’s* shop manager role’s permission list

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | Shop manager, permission | User that is not a shop manager, permission |
| Result | The permission is updated. | The permission is not updated. The system displays an error message. |

Close a Shop:

Actor: shop founder

Pre-Condition:

* shop founder must be logged in
* shop must exist

Parameters: shop id

Actions:

1. shop founder choose to close the shop
2. System changes the shop to be unavailable
3. For all manager and owner in shop:
   1. notifies the shop closed
4. For all acquisition by time:
   1. notifies user the shop closed

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop founder | Shop founder |
| Parameters | Valid shop id | Shop id that does not exist |
| Result | Shop is closed | Shop is not closed. The system displays an error message. |

Show Employees Info:

Actor: shop - owner

Pre-Condition:

* shop owner is logged in
* shop exists

Parameters: shop id

Actions:

1. show owner choose to show all employees info
2. for all employee in shop employee list:
   1. show user details.
   2. show user permission.

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop owner | Shop owner |
| Parameters | Valid shop id | Invalid shop id |
| Result | Details are displayed | Details are not displayed. Error message is displayed. |

Show Acquisition History:

Actor: user

Pre-Condition:

* user must be logged in
* user must have permission to shop history
* shop exists

Parameters: none

Actions:

1. user chooses to show acquisition history
2. System imports shop history.
3. System shows history

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | User | - |
| Parameters | None | - |
| Result | Show Acquisition History is displayed. | - |

**Shop Manager Cases:**

Read information and response:

Actor: shop managers

Pre-Condition:

* The shop manager must be logged in
* Shop must exist
* Shop must be available
* Shop manager must have the permission

Parameters: message id

Actions:

1. User chooses to show all information
2. User chooses a message
3. System shows option
4. User chooses to respond
5. User insert his answer
6. System removes the message from shop’s messages
7. System notifies the user with the response

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | Shop managers | Shop managers |
| Parameters | Valid message id | Invalid message id |
| Result | Message is taken care and removed. | Message is not removed. An error message is displayed. |

**System Manager Cases:**

Show Acquisition Info:

Actor: System-Manager

Pre-Condition:

* System manager must be logged in

Parameters: *shop*

Actions:

1. User choose to show acquisition history.
2. System presents history option.
3. User chooses acquisition history by shop.
4. System request for shop details.
5. User inserts *shop.*
6. System search acquisitions - shop acquisition history
7. System shows acquisitions

Acceptance Tests Paths:

|  |  |  |
| --- | --- | --- |
|  | Good Scenario | Bad Scenario |
| Participants | System manager | System manager |
| Parameters | Valid shop id | Invalid shop id |
| Result | The details are displayed | The details are not displayed. An error message is displayed. |