Workshop on Software Engineering Project

Final Version

Matan Hazan 315198796

Omer Kempner 322217472

Ori Kintzlinger 318929213

Arye Shapiro 313578379

Benny Skidanov 322572926

Use cases

Table of Contents

1.1. Loading the system 4

2.1. Entrance to the system 4

2.2. exit the system 5

2.3. Registration to the system 5

2.4. Login to the system 6

2.5. Receiving information about stores & products in certain stores 6

2.6.1. Search products 7

2.6.2. Filter desired products. 7

2.7. Add products to the shopping basket of a store. 8

2.8.1. Receiving information about personal shopping cart. 9

2.8.2. Editing personal shopping cart. 9

2.9.1. Purchase products. 10

2.9.2. Shopping by type 10

3.1. Logout 12

3.2. Creating a new store 12

3.7. Viewing personal purchase history 13

4.1.1. Adding a product to a store. 13

4.1.2. Removing a product from a store. 14

4.1.3. Editing a product in a store. 14

4.2.1. Adding purchase and discount policies in a store. 15

4.2.2. Removing purchase and discount policies from a store..16

4.2.3. Displaying purchase and discount policies of a store. 16

4.3. Adding a new owner to a store...17

4.4. Removing an owner from a store. 17

4.5. Adding a new manager to a store. 18

4.6. Changing store manager managing options..19

4.7. Remove store manager 20

4.8.1. Closing a store 21

4.8.2. Opening a closed store 22

4.9. Receive info about the store managers and owners...22

4.10.1. Receive info about messages and requests from consumers..23

4.10.2. Reply to messages and requests from consumers 24

4.11. Receive info about the store purchase history...24

5.1. Performing store actions based on manager permissions 25

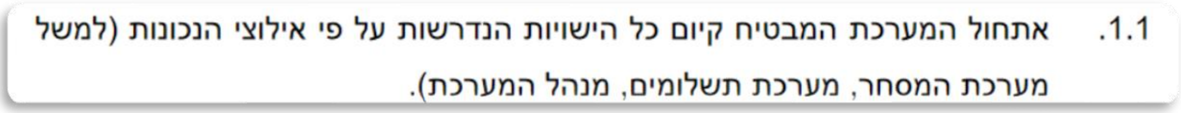
6.4.1. Consumer's purchase history..25

6.4.2. Store's purchase history 26

7. Payment using an external system 27

8. Shipping using an external system..28

Use cases



1.1. אתחול המערכת המבטיח קיום כל הישויות הנדרשות על פי אילוצי הנכונות (למשל
מערכת המסחר, מערכת תשלומים, מנהל המערכת).

Use case: Loading the system

Actor: User

Parameter: None

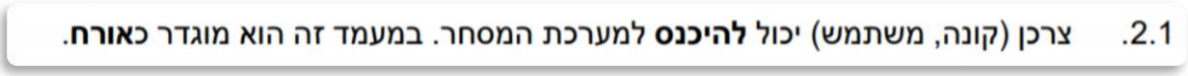
Precondition: None

Postcondition: The system is up. The trading, paying systems and system manager are on.

Textual description: A user starts a new instance of the trading system.

Actions :

1. **The system loads up.**



2.1. צרכן (קונה, משתמש) יכול **להיכנס** למערכת המסחר. במעמד זה הוא מוגדר **כאורח**.

Use case: Entrance to the system

Actor: User

Parameter: None

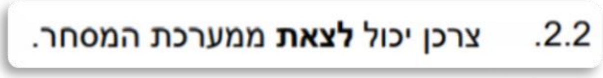
Precondition: The user is not logged in.

Postcondition: The user is a guest in the consuming system.

Textual description: Unregistered (guest) user enters the site.

Actions :

1. **A user enters the system.**



2.2. צרכן יכול **לצאת** ממערכת המסחר.

Use case: Exit the consuming system

Actor: User

Parameter: *None*

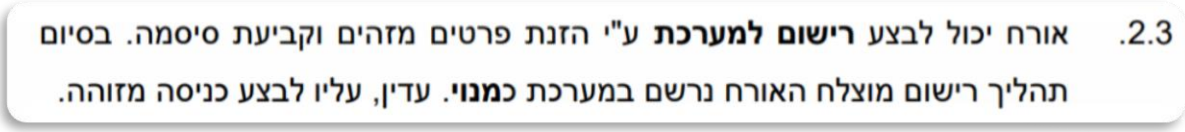
Precondition: The user is currently using the system.

Postcondition: User is outside the system; His shopping cart is deleted.

Textual description: A user that currently uses the system decides to exit.

Actions :

1. **User exists in the system.**



2.3. אורח יכול לבצע **רישום למערכת** ע"י הזנת פרטים מזהים וקביעת סיסמה. בסיום
תהליך רישום מוצלח האורח נרשם במערכת **כמנוי**. עדין, עליו לבצע כניסה מזוהה.

Use case: Registration to the system

Actor: User

Parameter: Registration details (e.g. username, password, etc.)

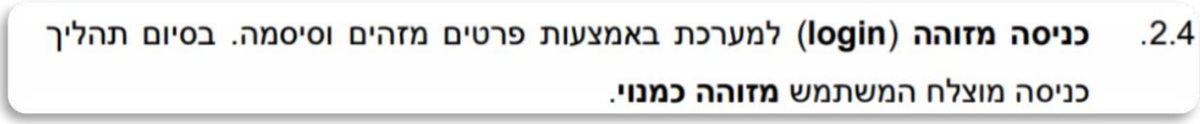
Precondition: The user is not registered.

Postcondition: The user is registered but still has to login to enter the consuming system.

Textual description: The user decides to register to the system to be identified. He does that by entering a username and a password that he will use to log in from now on.

Actions :

1. **User delivers registration details.**
2. **If the user's registration details are correct**
   - *i.* **Registration is approved**
3. **Else**
   - *i.* **Registration fails**



2.4. **כניסה מזוהה** (**login**) למערכת באמצעות פרטים מזהים וסיסמה. בסיום תהליך
כניסה מוצלח המשתמש **מזוהה כמנוי**.

Use case: Login to the system

Actor: User

Parameter: Login details (e.g. username, password, etc.)

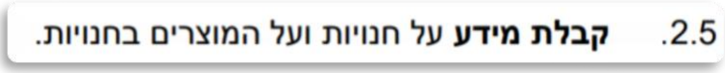
Precondition: The user is registered but not logged in.

Postcondition: The user is logged in to the system.

Textual description: The user decides to log in to the system by entering a username and password.

Actions :

1. **User gives identification details.**
2. **If the user's identification details are correct**
   - *i.* **Login approved**
3. **Else**
   - *i.* **Login fails**



2.5. **קבלת מידע** על חנויות ועל המוצרים בחנויות.

Use case: Receiving
information about stores & products in specific stores

Actor: User

Parameter: store name

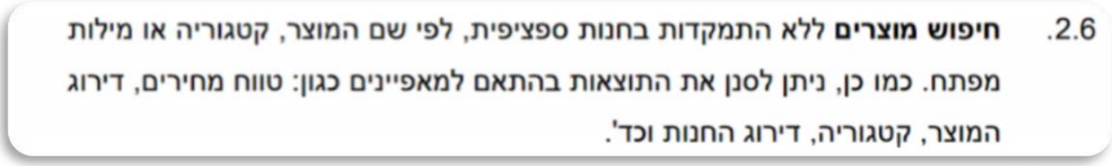
Precondition: *None*

Postcondition: Store's information and products of the store are presented to the user.

Textual description: The user is willing to get information about a particular store and its items.

Actions :

1. **The system presents information about the store and products that the store supplies.**



2.6. **חיפוש מוצרים** ללא התמקדות בחנות ספציפית, לפי שם המוצר, קטגוריה או מילות
מפתח. כמו כן, ניתן לסנן את התוצאות בהתאם למאפיינים כגון: טווח מחירים, דירוג
המוצר, קטגוריה, דירוג החנות וכו'.

2.6.1. search products.

Actor: User

Parameter: product classifiers (e.g., category, product name, keywords, etc.)

Precondition: Products are loaded into the system

Postcondition: A list of products (if they exist) Is presented on the screen

Textual description: User searches for products (without focusing on the specific store) based on their name, category, or keywords.

Actions :

1. **User delivers the desired name or category, or keyword of products.**
2. **The system presents the compatible products for the user's search.**

2.6.2. filter desired products.

Actor: User

Parameter: product filters (e.g., price - range, rating, etc.)

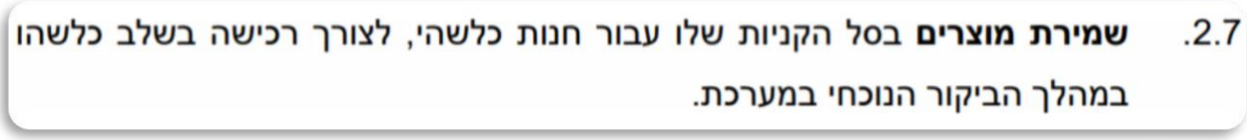
Precondition: A search has been done.

Postcondition: A list of products that passed the filter criteria is presented.

Textual description: User filters selected products based on price range, product/store rating, category, etc.

Actions :

1. **User delivers filter criteria of products (e.g. price range, rating, etc.)**
2. **The system presents the products that are compatible with the selected filter criteria**



2.7. **שמירת מוצרים** בסל הקניות שלו עבור חנות כלשהי, לצורך רכישה בשלב כלשהו
במהלך הביקור הנוכחי במערכת.

Use case: Add products to the shopping basket of a store.

Actor: User

Parameter: product

Precondition: None

Postcondition: desired products are saved in the shopping basket of the store.

Textual description: The user wants to add products to a shopping basket of a store to save those products for further purchase in the current session.

Actions :

1. **User selects products to be saved to his shopping cart.**
2. **The product is added to the shopping basket of the specific store, which is part of the whole shopping cart.**

2.8. **קבלת מידע** על עגלת קניות ועריכתה.

2.8.1. Receiving information about personal shopping cart.

Actor: User

Parameter: *None*

Precondition: None

Postcondition: Information (content) about the shopping cart is presented.

Textual description: The user wants to receive information about his shopping cart.

Actions :

1. **Content of the shopping cart is presented to the user (products, total price, quantity, etc.)**

2.8.2. Editing personal shopping cart.

Actor: User

Parameter: New content

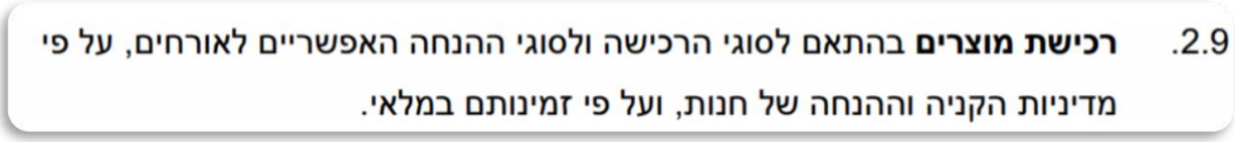
Precondition: Shopping cart is non - empty.

Postcondition: The changes that were made by the user are updated in the shopping cart.

Textual description: The user wants to edit his shopping cart (quantities, delete products, etc.

Actions :

1. **User edits the content of the shopping cart (changes quantities, sizes, etc.)**
2. **The edits the user has made are saved in his shopping cart**



2.9. **רכישת מוצרים** בהתאם לסוגי הרכישה ולסוגי ההנחה האפשריים לאורחים, על פי
מדיניות הקניה וההנחה של חנות, ועל פי זמינותם במלאי.

2.9.1. Purchase products.

Actor: User

***Parameter*:** products, shopping policy, discount policy

Precondition: The shopping cart contains the products that the user would like to purchase.

Postcondition: The purchase either being approved or fails.

Textual description: The User wants to purchase products; it depends on the shopping and discount policies (whether the user type, which is guest, can use the wanted payment and discount type) and whether the desired products to purchase are in stock.

Actions :

1. User is at purchase point
2. A check occurs, which verifies that the user meets the shopping policy requirements

   2a. If the user type and the product's shopping type don't suit the shopping policy, the purchase fails.
3. The user acts according to the product's shopping type as described in use case **2.9.2**.
4. A check occurs, which verifies that the user meets the discount policy requirements.

   4a. If the user type and the chosen shopping type don't suit the shopping policy, the user doesn't enjoy the discount.

   4b. else

   4.b.1. If there is a simple discount on the product, the price is updated accordingly.

   4.b.2. If there is a conditional discount on the product and the shopping cart meets the condition, the price is updated accordingly.
5. If the product is out of stock

   5a. The system presents a stock descriptive message, and the purchase fails.
6. Client is delivered to payment external system

   6a. If payment system fails to make the payment, the purchase fails.
7. Client is delivered to shipping external system

   7a. If shipping system fails to make the payment, the purchase fails.

2.9.2. Shopping by type

Actor: User

Parameter: Products, shopping policy

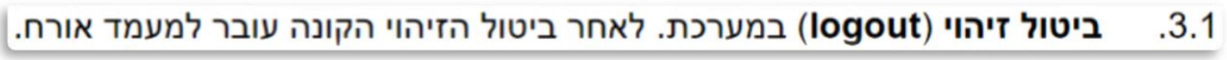
Precondition: The product has a shopping type

Postcondition: The product has been purchased according to its shopping type

Textual description: The User wants to purchase a product according to the product's shopping type. The shopping type can be different between products; therefore, the user has to act accordingly

Actions :

1. Shopping type branching :
    a. Product has instant purchase type.
    b. Product has bidding purchase type :
        i. User bids price for a specific product.
        ii. If **all** the store owners accept the bided price
            i. User can add the product to the shopping cart
        iii. If **one** store owner declines the bid offer
            i. User's offer is declined
        iv. If **one** of the store owners suggests a counteroffer
            i. If the user Accepts the counteroffer price
                i. User can add the product to the shopping cart
            ii. If the user declines the counteroffer price
                i. Offer is declined and removed



3.1. **ביטול זיהוי** (**logout**) במערכת. לאחר ביטול הזיהוי הקונה עובר למעמד אורח.

Use case: Logout

Actor: User

Parameter: None

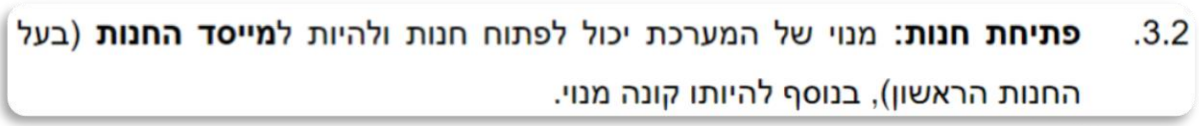
Precondition: User is logged in

Postcondition: User is no longer logged in

Textual description: the user leaves the system.

Actions :

1. **User leaves the system**



3.2. **פתיחת חנות:** מנוי של המערכת יכול לפתוח חנות ולהיות ל**מייסד החנות** (בעל
החנות הראשון), בנוסף להיותו קונה מנוי.

Use case: Creating a new store

Actor: User

Parameter: Store name

Precondition: User logged into the system, the store name isn't taken

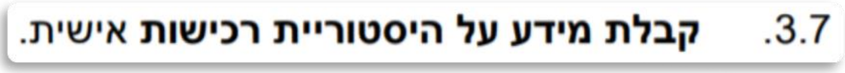
Postcondition: New store has been created, and the user is her owner

Parameter: Store details

Textual description: The user creates a new store.

Actions :

1. **User fills all the information about the new store**
2. **If store information is incorrect :**
   - i. **The system requests the user to correct the incorrect details.**
3. **Else:**
   - i. **The system creates a new store with the information the user filled.**



3.7 **קבלת מידע על היסטוריית רכישות** אישית.

Use case: Viewing personal purchase history

Actor: User

Parameter: store name

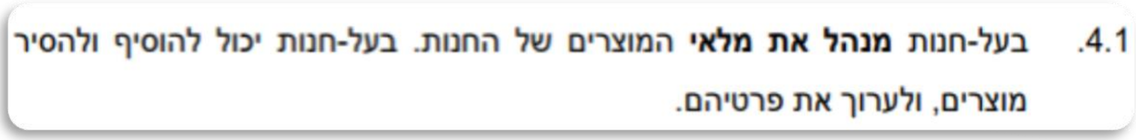
Precondition: None

Postcondition: Personal purchase history is presented to the user

Textual description: The user wants to observe his personal purchase history.

Actions :

1. **User requests for personal shopping history.**
2. **The system presents the user's past purchases**



4.1 בעל-חנות **מנהל את מלאי** המוצרים של החנות. בעל-חנות יכול להוסיף ולהסיר מוצרים, ולערוך את פרטיהם.

4.1.1. Adding a product to a store.

Actor: User

Parameter: product, store name

Precondition: The user exists in the system and has a store that he is an owner/seller.

Postcondition: A product is added to the store.

Textual description: The user requests to add a product to the store; the result depends on whether the user provided all necessary legal fields for the product.

Actions :

1. **The user requests to add a product to the store with all required product information.**
2. **If the user does not provide a mandatory product field:**
   i. **The system returns a descriptive error message.**
   ii. **The system returns to stage 1.**
3. **If the user enters an already existing product:**
   i. **The system returns a descriptive error message.**
   ii. **The system returns to stage 1.**
4. **The product is added to the system successfully.**

4.1.2. Removing a product from a store.

Actor: User

Parameter: product, store name

Precondition: The User exists in the system and has a store with at least one product. He is an owner/seller.

Postcondition: A product is removed from the store.

Textual description: The user requests to remove a product from a store.

Actions :

1. **The user requests to remove a product from the store.**
2. **If the product does not exist in the system:**
   1. ***i.*** **The system returns a descriptive error message.**
   2. ***ii.*** **The system returns to stage 1.**
3. **The system prompts the user for confirmation.**
4. **User approves.**
5. **The product is removed from the system successfully.**

4.1.3. Editing a product in a store.

Actor: User

Parameter: product's new field/s

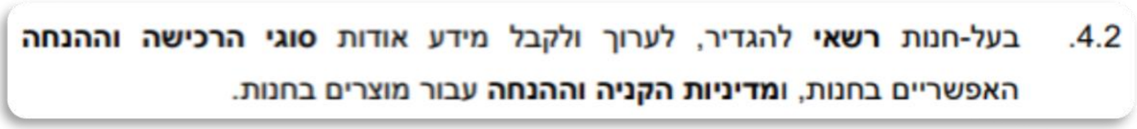
Precondition: The User exists in the system and has a store with at least one product that he is an owner\seller.

Postcondition: The picked product is edited in the store.

Textual description: The user requests to edit a product with the new desired fields; the request depends on whether he provided new legal areas.

Actions :

1. The user requests to edit a product.
2. The system prompts the user for confirmation.
3. User approves.
4. If the product doesn't exist in the system:
    - *i.* The system returns a descriptive error message.
    - *ii.* The system returns to stage 1.
5. if one of the edited fields is invalid:
    - *i.* The method returns a descriptive error message.
    - *ii.* The system returns to stage 1.
6. The product is successfully edited.



4.2. בעל-חנות **ראשי** להגדיר, לערוך ולקבל מידע אודות **סוגי הרכישה וההנחה**
האפשריים בחנות, **ומדיניות הקניה וההנחה** עבור מוצרים בחנות.

4.2.1. *Adding purchase and discount policies in a store.*

Actor: User

Parameter: New policy

Precondition: The User exists in the system and owns a store.

Postcondition: A new purchase\discount policy is added to the store.

Textual description: The user requests to add a policy or add a discount in one of the stores he owns.

Actions :

1. The user requests to add a discount or purchase policy.
2. The system prompts the user to pick a policy (out of an existing list).
3. The user picks the desired policy.
4. If the policy does not exist in the system:
    - *i.* The system returns a descriptive error message.
    - *ii.* The system returns to stage 1.
5. User approves.
6. The policy is successfully added to the store.

4.2.2. Removing purchase and discount policies from a store.

Actor: User

Parameter: Store's policy

Precondition: The User exists in the system and owns a store.

Postcondition: A new purchase\discount policy is removed from the store.

Textual description: The user requests to remove a discount or purchase policy in one of the stores he owns.

Actions :

1. **The user requests to remove a purchase or discount policy.**
2. **If the policy does not exist in the store:**
   - i. **The system returns a descriptive error message.**
   - ii. **The system returns to stage 1.**
3. **The system prompts the user for confirmation.**
4. **User approves with "OK."**
5. **The policy is successfully removed.**

4.2.3. Displaying purchase and discount policies of a store.

Actor: User

Parameter: Store name

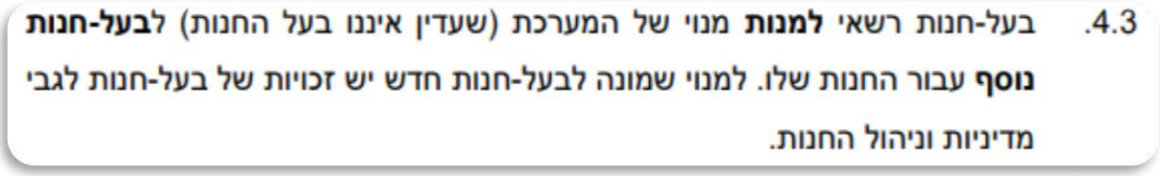
Precondition: The User exists in the system and owns a store. Additionally, a few policies are defined for the store.

Postcondition: All purchase and discount policies are displayed in different lists.

Textual description: The user requests to display all policies in the store.

Actions :

1. **The user requests to display all policies.**
2. **System returns all purchase and discount policies in the store.**



4.3. בעל-חנות ראשי **למנות** מנוי של המערכת (שעדיין איננו בעל החנות) **לבעל-חנות**
נוסף עבור החנות שלו. למנוי שמונה לבעל-חנות חדש יש זכויות של בעל-חנות לגבי
מדיניות וניהול החנות.

Use case: Adding a new owner to a store.

Actor: User

Parameters: new owner username, store name

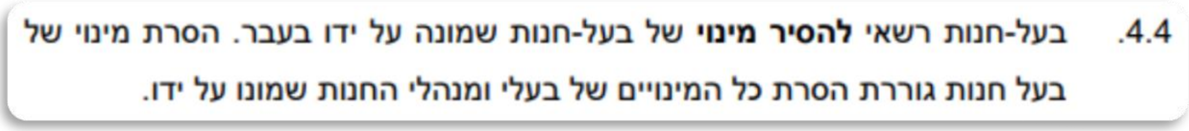
Precondition: The User exists in the system and owns a store.

Postcondition: A new owner is added to the store.

Textual description: The user requests to add a new owner to the store he is an owner of.

Actions :

1. **The user requests to add a new owner with the new owner username.**
2. **The system prompts the user for confirmation.**
3. **User approves.**
4. **If the username does not exist or is already an owner:**
   - *i.* **The system returns a descriptive error message.**
   - *ii.* **The system returns to stage 1.**
5. **The new owner is successfully added to the store.**



4.4. בעל-חנות ראשי **להסיר מינוי** של בעל-חנות שמונה על ידו בעבר. הסרת מינוי של
בעל חנות גוררת הסרת כל המינויים של בעלי ומנהלי החנות שמונו על ידו.

Use Case: Removing an owner from a store.

Actor: User

Parameter: owner\manager username, store name

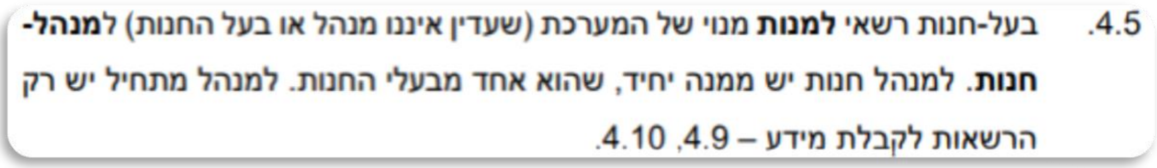
Precondition: The User exists in the system and owns a store. Additionally, the user has owners that he added.

Postcondition: An owner is removed from the store, and all owners\managers under him in the hierarchy are removed.

Textual description: The user makes a request to remove an owner\manager from the store owner page.

Actions :

1. User requests to remove an owner\manager in the store with the username of the owner\manager to remove.
2. The system prompts the user for confirmation.
3. User approves.
4. If the username of the removed owner\manager does not exist in the system:
    - *iii.* The system returns a descriptive error message.
    - *iv.* The system returns to stage 1.
5. The system removes the owner\manager and those under the removed owner\manager hierarchy.



4.5 בעל-חנות ראשי **למנות** מנוי של המערכת (שעדיין איננו מנהל או בעל החנות) ל**מנהל-חנות**. למנהל חנות יש ממנה יחיד, שהוא אחד מבעלי החנות. למנהל מתחיל יש רק הרשאות לקבלת מידע – 4.9, 4.10.

Use case: Adding a new manager to a store.

Actor: User

Parameter: new manager username, store name

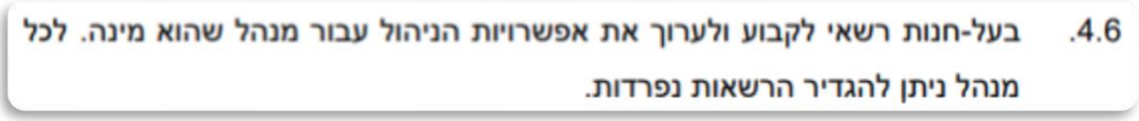
Precondition: The User exists in the system and owns a store.

Postcondition: A new manager is added to the store.

Textual description: The User clicks the "add new manager" on the store owner page.

Actions :

1. The user requests to add a new manager with the new manager username.
2. The system prompts the user for confirmation.
3. User approves.
4. If the username does not exist or is already a manager:
    - *i.* The system returns a descriptive error message.
    - *ii.* The system returns to stage 1.
5. The new manager is successfully added to the store.



4.6. בעל-חנות רשאי לקבוע ולערוך את אפשרויות הניהול עבור מנהל שהוא מינה. לכל
מנהל ניתן להגדיר הרשאות נפרדות.

Use case: Changing store manager managing options

Actor: store owner, store manager appointed by the store owner

Parameter: manager username, store name, managing option

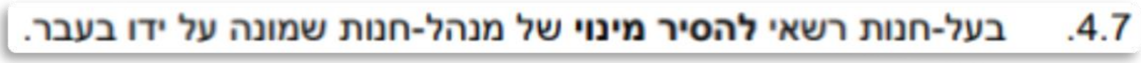
Precondition: The store owner is logged in the system and already appointed the store manager

Postcondition: The appointed store manager has his managing options changed by the store owner.

Textual description: The store owner edits one of his appointed store manager managing options.

Actions :

1. **Store owner requests to edit some store manager managing options for the given store.**
2. **If the store does not exist:**
   i. **Reject the store owner's request**
3. **If the store owner does not own the given store:**
   i. **Reject the store owner's request**
4. **If the store manager does not exist :**
   i. **Reject the store owner's request**
5. **If the given store owner does not assign the store manager:**
   i. **Reject the store owner's request**
6. **If the managing options are invalid:**
   i. **Reject the store owner's request**
7. **The request is accepted, and the store manager permissions update accordingly.**



4.7. בעל-חנות ראשי **להסיר מינוי** של מנהל-חנות שמונה על ידו בעבר.

Use case: Remove store manager

Actor: store owner, store manager appointed by the store owner

Parameter: manager username, store name

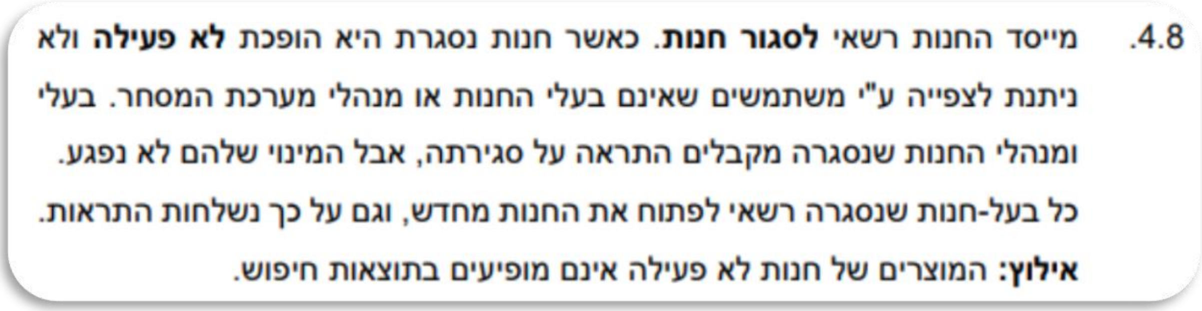
Precondition: The store owner is logged in the system and already appointed the store manager

Postcondition: Th previously appointed store manager is removed from his position as store manager.

Textual description: The store owner removes the appointed store manager from managing his store

Actions :

1. **The store owner requests to remove one of his appointed store managers from his position.**
2. **If the store does not exist:**
   - *i.* **Reject the request**
3. **If the store owner does not own the given store:**
   - *i.* **Reject the request**
4. **If the store manager:**
   - *i.* **Does not exist – reject the request**
   - *ii.* **Not assigned by the given store owner – reject the request**
5. **The request is accepted, and the store manager is removed from his position.**



4.8. מייסד החנות רשאי **לסגור חנות**. כאשר חנות נסגרת היא הופכת **לא פעילה** ולא
ניתנת לצפייה ע"י משתמשים שאינם בעלי החנות או מנהלי מערכת המסחר. בעלי
ומנהלי החנות שנסגרה מקבלים התראה על סגירתה, אבל המינוי שלהם לא נפגע.
כל בעל-חנות שנסגרה רשאי לפתוח את החנות מחדש, וגם על כך נשלחות התראות.
אילוץ: המוצרים של חנות לא פעילה אינם מופיעים בתוצאות חיפוש.

4.8.1. Closing a store

Actor: Store founder, every other store owner and store manager (for the same store)

Parameter: store name

Precondition: The store founder is logged in the system, and the store is not already closed

Postcondition: The store closes and becomes "inactive.". Store items do not appear on the search results of the consuming system anymore. Every other store owner and manager receives a notification about the store closing, and their positions related to the store stay the same.

Textual description: The store founder closes the store

Actions :

1. **Store owner requests to close his store**
2. **If the store:**
   - i. **Does not exist – reject the request**
   - ii. **Already closed – reject the request**
3. **If the store owner does not own the given store:**
   - i. **Reject the request**
4. **The proposal is accepted, and the store closes.**
5. **The system sends relevant notifications to every other store owner and store manager (for the same store).**

4.8.2. Opening a closed store

Actor: Store founder, every other store owner and store manager (for the same store)

Parameter: store name

Precondition: The store founder is logged in the system, and the store is closed

Postcondition: The store opens and becomes "Active" . store items return to appear on the search results of the consuming system. Every other store owner and manager receives a notification about the store opening.

Textual description : The store founder opens the closed store

Actions :

1. **Store owner request to open his store**
2. **If the store:**
   - *i.* **Does not exist – reject the request**
   - *ii.* **Already open – reject the request**
3. **If the store owner does not own the given store:**
   - *i.* **Reject the request**
4. **The proposal is accepted, and the store opens.**
5. **The system sends fitting notifications to every other store owner and store manager (for the same store)**

4.9. בעל חנות יכול **לקבל מידע** על כל בעלי התפקידים בחנות והרשאותיהם.

Use case: Receive info about the store managers and owners

Actor: Store owner

Parameter: Store name

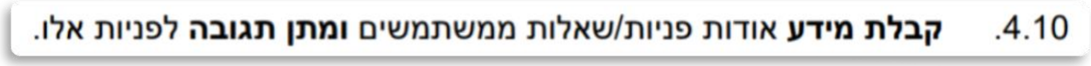
Precondition: The store owner is logged in the system

Post - condition : *None*

Textual description: The store owner is presented with all of the other store manager's and owner's info, including their permissions.

Actions :

1. **Store owner requests to receive information about every other employee in his store and their permissions.**
2. **If the store:**
   - i. **Does not exist – reject the request**
3. **If the store owner:**
   - i. **Does not own given store – reject the request**
4. **The request is accepted, and all of the information is sent to the store owner by the system.**



4.10. **קבלת מידע** אודות פניות/שאלות ממשתמשים **ומתן תגובה** לפניות אלו.

4.10.1. Receive info about messages and requests from consumers

Actor: Store owner

Parameter: Store name

Precondition: The store owner is logged in the system

Post - condition : None

Textual description: The store owner is presented with all the messages and requests from consumers.

Actions :

1. **Store owner requests to receive information about all messages and requests from consumers.**
2. **The request is accepted, and all of the information is sent to the store owner by the system.**

4.10.2. Reply to messages and requests from consumers

Actor: Store owner

Parameter: Store name, reply

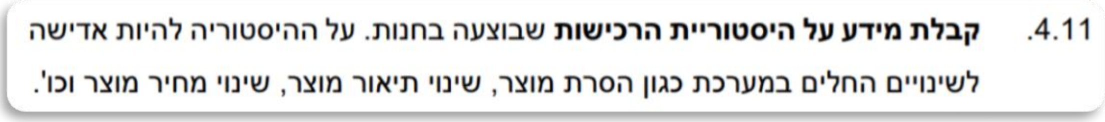
Precondition: The store owner is logged in the system

Post - condition : None

Textual description: The store owner is presented with all the messages and requests from consumers.

Actions :

1. **Store owner requests to reply to some message/request from consumers.**
2. **If the message/request:**
    - ***i.*** **Does not exist – reject the request**
3. **The request is accepted, and the message/request is replied to.**



4.11. **קבלת מידע על היסטוריית הרכישות** שבוצעה בחנות. על ההיסטוריה להיות אדישה
לשינויים החלים במערכת כגון הסרת מוצר, שינוי תיאור מוצר, שינוי מחיר מוצר וכו'.

Use case: Receive info about the store purchase history

Actor: store owner

Parameter: store name

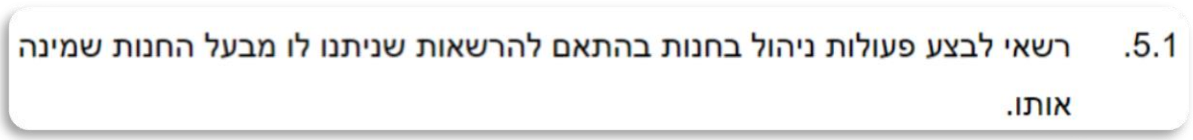
Precondition: The store owner is logged in the system

Post - condition : None

Textual description: The store owner is presented with all of the store purchase histories

Actions :

1. **Store owner requests to receive information about the store's purchase history.**
2. **If the store:**
    - ***i.*** **Does not exist – reject the request**
3. **If the store owner:**
    - ***i.*** **Does not own given store – reject the request**
4. **The request is accepted, and all of the information is sent to the store owner by the system.**



5.1. רשאי לבצע פעולות ניהול בחנות בהתאם להרשאות שניתנו לו מבעל החנות שמינה
אותו.

Use case: Performing store actions based on manager permissions

Actor: User

Parameters: store name, username

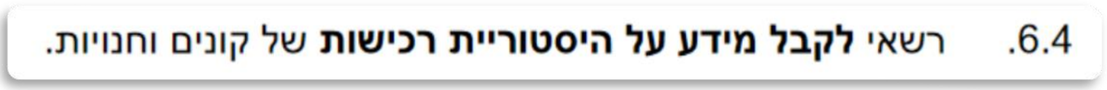
Precondition: User has manager permissions

Postcondition: The manager did the action based on his permissions

Textual description: An admin tries to do an action; if he got permission, the move would succeed, it will fail and tell him he doesn't have permission to do so.

Actions :

1. **The user does an activity in the store.**
2. **If the user has permissions to do the action:**
   - *i.* **The system will act.**
3. **If the user doesn't have the permissions to do the action:**
   - *i.* **The system will not perform the action**



6.4. רשאי **לקבל מידע על היסטוריית רכישות** של קונים וחנויות.

6.4.1. consumer's purchase history

Actor: User with Admin permissions

Parameter: Consumer's name

Precondition: selected consumer exists in the system

Postcondition: a list of a specific consumer purchase history(if it exists) is presented

Textual description: admin presses the button "see consumer purchase history" new window pop up with a text box, the admin fills the text box with a consumer name, the consumer's purchase list is presented on the screen.

Actions :

1. **Admin requests a consumer's purchase history by his details.**
2. **If the consumer's details are incorrect:**
   i. **System request to correct the incorrect details.**
3. **Else :**
   i. **The system presents the consumer's purchase history**

6.4.2. store's purchase history

Actor: User with Admin permission

Parameter: Store name

Precondition: selected store exists in the system

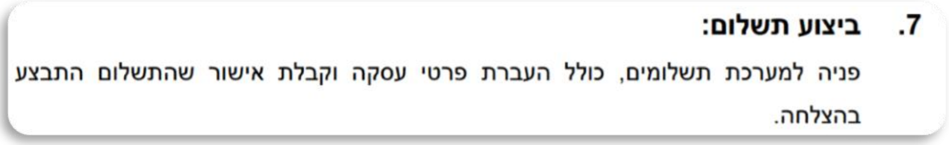
Postcondition: a list of a specific consumer purchase history(if it exists) is presented

Textual description : admin presses the button "see store purchase history", a new window pop up with a text box, the admin fills the text box with a store name, the store purchase list is presented in the screen.

Actions :

1. **Admin requests a purchase history of a specific store by it's details.**
2. **If the store details are invalid:**
   i. **The system requests to correct the invalid details.**
3. **Else:**
   i. **The system presents the store's purchase history**

External Systems use cases



7. ביצוע תשלום:

פניה למערכת תשלומים, כולל העברת פרטי עסקה וקבלת אישור שהתשלום התבצע בהצלחה.

Use case: Payment using an external system

Actors: System, external payment system

***Parameter*:** amount to charge

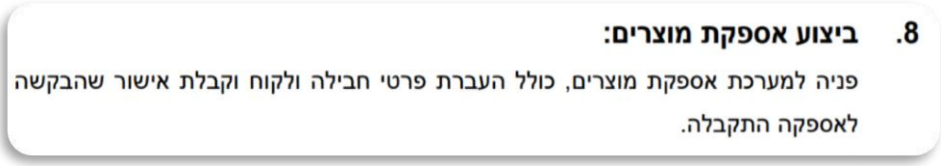
Precondition: The product that is desired to be bought exists in the system

Postcondition: User purchase has been made successfully / failed, the purchase has added to user's assets.

Textual description: Our system doesn't implement the act of charging money. Therefore, we delegate the payment method to an external system.

Actions :

1. The user pays on the content of his shopping cart.
2. A redirection to the external system is made with the user's purchase details.
3. External system returns whether the payment process has been made successfully.
4. if the payment process failed :
    a. The system presents a descriptive error message.
    b. Return to stage 1
5. The System presents successful purchase message.



8. **ביצוע אספקת מוצרים:**

פניה למערכת אספקת מוצרים, כולל העברת פרטי חבילה ולקוח וקבלת אישור שהבקשה לאספקה התקבלה.

Use case : Shipping using an external system

Actors : System, external shipping system

Parameter : shipping details

Pre - condition : User made a purchase

Post – condition : Shipping confirmation document is presented to the user

Textual description : Our system doesn't implement the act of shipping purchased products to the consumers. Instead, we delegate the action of shipping products to an external system.

Actions :

1. System has made a request to the external shipping system with the purchase details.
2. External shipping system returns whether the shipping request has been made successfully.
3. if shipping process failed :
   a. System presents a descriptive error message.
4. Shipping confirmation document is presented to the user

Acceptance Tests

Action	Data	Expected Result
2.3 Registration to the system	Non existing username, valid password	Successful registration
	Existing Username	Unsuccessful registration
2.4 Login to the system	Existing username & matching password	Successful login
	• Existing username with incompatible password • Non existing username	Unsuccessful Login
2.5 Receiving information about stores & products in certain stores	Existing store	correct information about the store & products in it
	Non – existing store	Null, which means no info
2.6.1 Search products	Existing product in store, with valid category	list of suitable products
	Category doesn't exist	Empty list (no suitable products)
2.6.2 Filter desired products	Keyword (filter) valid & suitable for products	list of suitable products

	Keyword doesn't suit any product	Empty list (no suitable products)
2.7 Add products to the shopping basket of a store	All details are valid (store, products in store, purchase)	successful addition to user's shopping basket
	No sad path	
2.8.1 Editing personal shopping cart	Shopping cart exists and contains products	Valid shopping cart content
	New shopping cart	Empty shopping cart
2.8.2 Edit shopping cart (adding was checked, checking removal)	Shopping basket and products details are valid	shopping cart has been updated, successful removal
	Invalid details (product's GUID)	shopping cart does NOT update, Unsuccessful removal
2.9.1 Purchase products	Purchase details are valid	Purchase performed successfully
	Purchase details are invalid	Unsuccessful purchase
3.2 Creating a new store	Store details are valid	Successful creation
3.7 Viewing personal purchase history	Purchase history exists	personal shopping history is returned
4.1.1 Adding a product to a	User is store owner	The products have been added

store		successfully.
	User is not the store owner	Authorization error is thrown
4.1.2 Removing a product from a store	User is store owner	The products have been removed successfully.
	User is not the store owner	Authorization error is thrown
4.1.3 Editing a product in a store	User is store owner & product details are valid	The products have been updated successfully.
	1. User is not the store owner 2. Products details are invalid	Authorization / validation error (respectively) is thrown
4.3 Adding a new owner to a store	The user is the store owner & the new nominated owner details are valid (including that he is not a seller already)	The User has been added as a store owner.
	1. The owner is a seller 2. The user is not the	The User has not been added as a store owner.

	store owner 3. The nominated owner is not registered	
4.4 Remove store owner	1. The current user is a store owner 2. Existing store owner to delete 3. Store owner to be deleted is a subordinate of the user	The wanted store owner and his subordinates are deleted
	1. The current user is a store owner 2. Existing store owner to delete 3. Store owner to be deleted is **NOT** a direct Nominee of the user	The wanted store owner and his subordinates are **NOT** deleted
4.5 Adding a new manager to a store	The user is the store owner & the new nominated owner details are valid	The User has been added as a store manager.

	(including that he is not a seller already)	
	1. The owner is a seller 2. The user is not the store owner 3. The nominated owner is not registered	The User has not been added as a store manager.
4.6 Changing store manager managing options	+ Existing store owner + Existing store manager + Valid managing options	Success: The managing options for the store manager changed according to the new chosen managing options
	+ Existing store owner + Existing store manager - invalid managing options	Validation error is thrown
4.7 Remove store manager	Existing store owner & Existing store manager	Success: The previously appointed store manager is removed from his position as store manager.
	Non existing store manager	The removal could not been made.
4.9 Receive info about the	Existing store owner/manager	Success: The requested info is presented

store managers and owners	Non existing store owner/manager	Authorization error is thrown
4.11 Receive info about the store purchase history	Existing store owner & store details is valid	Success: The requested info is presented
	Non existing store owner	Authorization error is thrown
5.1 Performing store actions based on manager permissions	User that has manager permissions to do a certain action	Successful action
	User that doesn’t have manager permissions to do a certain action	Authorization error
6.4.1 Consumer's purchase history	Valid consumer details	Successful purchase history is returned
	Non existing consumer	Empty list is returned
6.4.2 Store's purchase history	Valid store details	Successful purchase history is returned
	Non existing store	Empty list is returned
7 Payment using an external system	External payment system collapses	purchase fails!
8 Shipping using an external system	External supply system collapses	purchase fails!

Glossary

Term	Description
Discount policy	A set of conditions that must be met for a Discount to be applied successfully.
Discount Type	A discount type is one of the following types: • Composite (conditional) discount type. • Simple discount type.
Discount	A deduction from the usual cost of product, can be applied to a shopping basket, specific product, or a category.
Guest	An unregistered entity.
Manager permissions	A subset of "Store basic management permissions" that were received by the owner that assigned the user to be a manager.
Notification	An instant message that can be received by a user.
Owner permissions	Permissions include: • Assign new sellers to the store he is the owner of. • An owner has all store basic management permissions.
Product	A unique entity that belongs to a store and can be purchased by users.
Product offer	An offer made by a user on a specific product. Can be approved, declined, or countered by any owner of the store that contains the specific product.
Store policy	A set of conditions that must be met for a purchase to be successful.
Purchase type	A purchase type is one of the following types: • Instant purchase type. • Bidding purchase type

Seller	A seller is either an Owner or a Manager.
Shopping basket	A user's collection of store related products that the user is interested in purchasing.
Shopping cart	A user's collection of shopping baskets.
Store	Collection of products and functionalities regarding purchasing products.
Store basic management permissions	Permissions include: • Adding products. • Deleting products. • Editing products. • Viewing other sellers of the store.
Store founder	The store owner that has opened the store.
Store manager	A user that has been assigned to be a manager by another owner. A manager has "Manager permissions" over the store.
Store owner	A user that has opened a store or was assigned to be an owner by another owner.
System Admin	The admin of the system.
User	A registered entity.