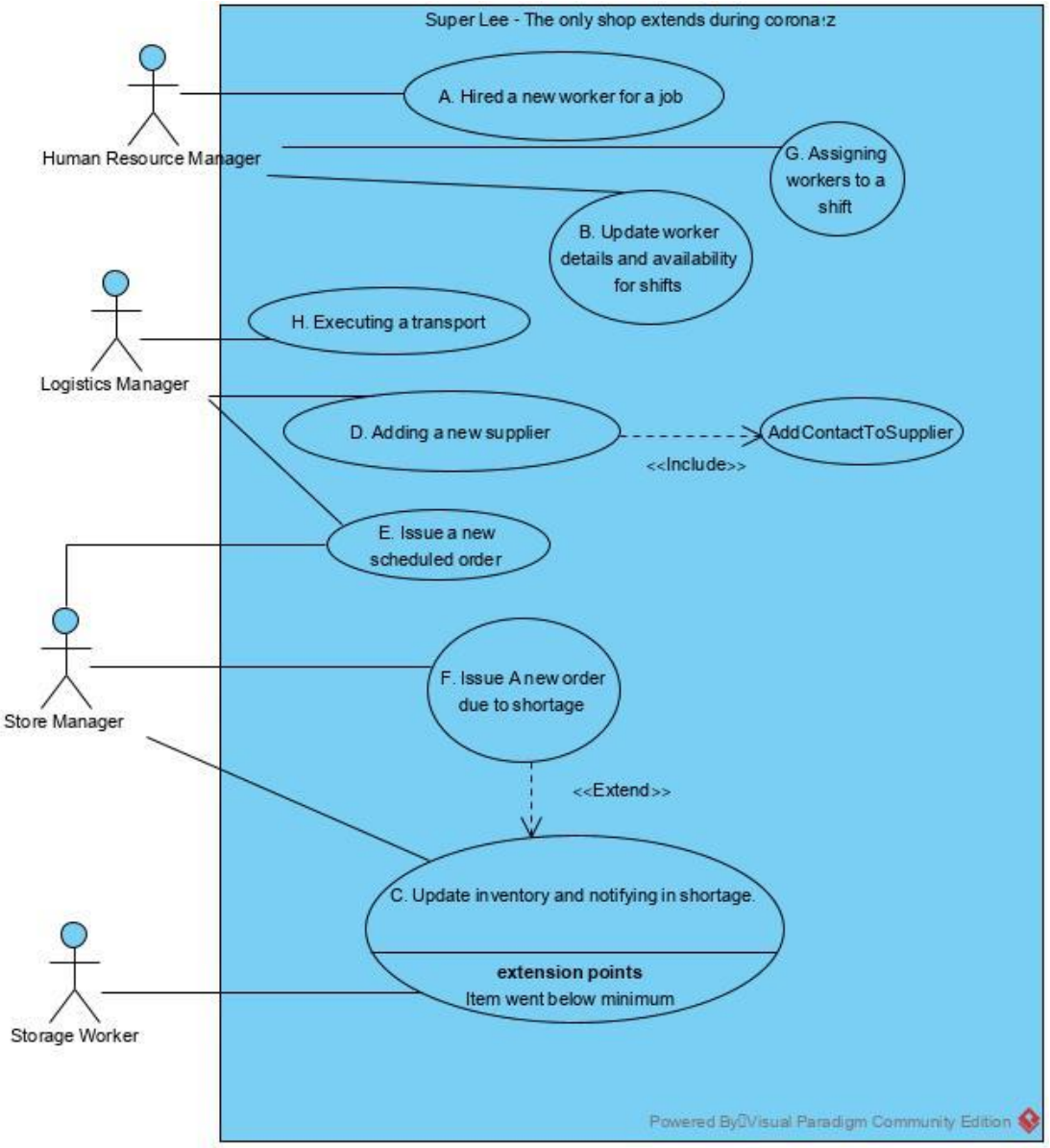
ניתוח התנהגותי



1. Use Case Diagram עבור כל נסיבות השימוש הבאות:

a. כניסת עובד חדש לתפקיד
b. עדכון פרטי עובד קיים וזמינות למשמרות
c. עדכון מלאי והתראה על חוסרים
d. הוספת ספק חדש
e. הוצאת הזמנה תקופתית מספק
f. הוצאת הזמנה מספק עקב חוסר
g. שיבוץ עובדים למשמרות
h. הוצאת הובלה

:Use Cases Description

שם תרחיש- Use Case Name:

הוצאת הזמנה תקופתית מספק

תיאור התרחיש- Textual description:

הזמנה של מוצרים שתצא בתדירות קבועה, בהתאם להגדרת הבקשה ע"י מחסנאי או אחראי מחסן. ניתן להגדיר את התדירות באמצעות ימים בשבוע ותדירות שבועית (כל כמה שבועות).

רשימת שחקנים- List of actors:

- מחסנאי
- אחראי מחסן

תנאים מקדימים- Pre-conditions:

- רשימת המוצרים לא ריקה

תנאי סיום- Post-conditions:

- במידה וההזמנה הצליחה ההזמנה נשמרה במערכת

תרחיש הצלחה ראשי- Main success scenario:

- Inventory Worker or Suppliers Manager choose option "create new periodical order" in the appropriate menu
- He enters the necessary input:
    - List of Product ids and amounts
    - List of days of the week
    - Number for week period
- The system goes threw all the suppliers in the system and check if there is a supplier with the requested items
- From those that passed the items check, the system find the suppliers that their supplying days match the requested supplying days of the periodical order
- From the suppliers that passed both checks, the system checks which supplier has the lowest total price for the requested items in the requested amounts
- The system create a new order for that supplier with the listed items and amounts and records the new supplier order in periodical orders db with the days, items and weekly period.

Pseudo code

1. User type "create new periodical order
    i. 0166 5
    ii. 1234 8
    iii. 6985 20
    iv. Days: Sunday, Friday
    v. 2
2. -> CreateNewPeriodicalOrder(Map<Items,Amount> shoppingList, List<Days> req_days, weekPeriod)
3. List<SupplierID> allSuppliers=getAllSuppliers()
4. For(int i=0;i<allSuppliers.size();i++)
    a. if(!equals(shopingList.items,allSuppliers.at(i).items)

b. || !equals(req_days,allSuppliers.at(i).supplyDays))
    i. allSuppliers.delete(i);

5. Supplier minPriceSupplier;
6. Integer minTotalPrice;
7. For(int i=0; i< allSuppliers.size();i++)
    a. Integer currentPrice= allSuppliers.at(i).getTotalPrice(shoppingList.items());
    b. If(current<minTotalPrice)
        i. minTotalPrice=currentPrice
        ii. minPriceSupplier=allSuppliers.at(i)
8. int retDd=createNewPeriodicalOrder(minPriceSupplier, shoppingList, req_days, weekPeriod)
9. return retID

<u>– Main success scenario & Alternatives/Extensions-אלטרנטיבות/הרחבות</u>

At any alternatives or extensions point the user will get appropriate message that describe the error, and go back to the update products loop:

1. Wrong input.
2. There are no suppliers in system
3. The suppliers in the system doesn’t supply the total requested shopping list
4. The suppliers in the system doesn’t supply in the requested days
5. Something else went wrong in the order creation process.

Pseudo code

1. If (shoppingList.isEmpty() || req_days.isEmpty() || weekPeriod<=0)
    a. Print "wrong inout! Try again"
2. ## after the first for loop (filter)
3. If(allSuppliers.isEmpty())
    a. Print "No supplies in the system supply the requested items or in the requested days"
4. If(retID=-1)
    a. "Something went wrong"

שם תרחיש- Use Case Name:

ייצור הזמנה חדשה עקב מחסור.

תיאור התרחיש- Textual description:

כאשר הכמות של מוצר מסוים מגיעה לנקודת המינימום (המוגדרת בפרטי מוצר כלשהו), המערכת תנפיק הזמנה חדשה על מנת לקבל אספקה נוספת עבור אותו המוצר.

רשימת שחקנים- List of actors:

עובדי המחסן, ספקים

תנאים מקדימים- Pre-conditions:

כמות המוצר חוצה את נקודת המינימום המוגדרת לו.

תנאי סיום- Post-conditions:

עבור כל מוצר שהכמות שלו חצתה את נקודת המינימום המוגדרת לו, נוציא הזמנה ונשלח לספק.

תרחיש הצלחה ראשי - Main success scenario

1. Inventory worker choose option "update inventory" after stocktaking in the shop.
2. For each product:
    2.1 Worker update quantities.
    2.2 If the product got below minimum amount:
        2.2.1 Mark the product as missing. (Issue 'itemOrder' object)
        2.2.2 Add this product to shortage order.

3 The system passing immediately the shortage order to suppliers.
4 Suppliers issue an order.

Pseudo code

10. User type "update inventory" → UpdateInvWorker ()
11. While (item_id != 0)
    2.1 item.updateMyQuantities (quantityMissStock, quantityMissShop, '-')
    2.2 if(item.checkMinimumQuant ())
        2.2.1 itemOrder ← issueOrderForShortageItem ()
        2.2.2 shortageOrder ← addItemToShortageOrder(itemOrder)
12. Inv2supCtrl.placeNewShortageOrder(shortageOrder)
13. myOrderAndProductManagment.createRegularOrder (order)

Alternatives and extensions:

At any alternatives or extensions point the user will get appropriate message that describe the error, and go back to the update products loop:

6. Wrong input.
7. Item id isn’t exist.

8. Place shortage order didn’t succeed.

Pseudo code

```
1. If input != 'id quantityStock quantityShop'
        8.1 Print "wrong input! Type again"
2. If id in input is not exist
        8.2 Print "Item isn’t exist in the inventor! Type again"
3. Result res = placeNewShoratgeOrder ()
        3.1 If(res.isFailure ())
                    3.1.1    Print res.getMessage ()
```

H. הוצאת הובלה

Textual Description:

כאשר ישם ספקים אשר לא מספקים סחורה לסניפים צריך להוציא הובלות לספקים אלו כדי להביא את הסחורה מהספק לסניף.

List of actors:

אחראי הובלות

Pre-conditions:

סניפים להובלה אליהם, ספקים לקחת את הסחורה, משאית לבצע בה את ההובלה, נהג שינהג במשאית, מחסנאי אחר יקבל את הסחורה בסניף.

Post-conditions:

יצאה הובלה לסניף אחר מתבצעת בתאריך הנבחר, עם נהג שפנוי להובלה ועובד באותה משמרת, ומחסנאי ההנמצא בסניף לקבלת ההובלה.

Main success scenario:

1. האחראי הובלות בוחר באופציה הוצאת הובלה שגרתית ("Routine transport")
2. לאחר מכן הוא בוחר אזור לספקים לביצוע הובלה מאופציות של האזורים המודפסות לו למסך.
3. מהספקים הקיימים באותו איזור הוא בוחר ספק אחד או מספר ספקים.
4. המשתמש בוחר אזור לחנויות לביצוע ההובלה מהאזורים שהמערכת הדפיסה למסך.
5. מהחנויות הקיימות באותו אזור המשתמש בוחר חנות או מספר חנויות לביצוע ההובלה.
6. המערכת מבקשת מהמשתמש לבחור יום להובלה וסוג משמרת (בוקר/ ערב).
7. המערכת מדפיסה למסך את המשאיות הפנויות באותו תאריך ומשמרת והמשתמש בוחר משאית להובלה.
8. המערכת מדפיסה למסך את רשימת הנהגים העובדים, פנויים באותו היום והמשמרת ובעלי רשיון נהיגה מתאים למשאית, המשתמש בוחר נהג להובלה.
9. לאחר מכן המערכת מבקשת מהמשתמש להכניס את רשימת הפריטים אותם הוא ירצה להעביר מכל ספק לכל אחת מהחנויות.
10. לאחר מכן ההובלה נרשמה בהצלחה.

Pseudo code

```
1. user type “Routine transport” -> Regular_stock_transport()
2. transportation_service.get_area_for_suppliers()
3. suppliersArea <- user input
4. site_service.getSuppliersbyarea(suppliersArea)
5. supplieresID [] <- user input
6. transportation_service.get_area_for_stores()
7. storesArea <- user Input
8. site_service.get_Stores_By_specific_area(storesArea)
9. storesID [] <- user input
10. date <- user input
11. shift <- user input
12. trucks_service.getFreeTrucks(date, shift)
13. truck <- user input
14. drivers_service.getDriverToTrucks(truckId, date, shift)
15. driver <- user input
16. for (storeId in storesID):
17.     for (supplierid in supplieresID):
18.         itemList<Quantity, itemName > <- user input
19. transportation_service.createRegularTransportation()
```

<u>Alternatives and extensions:</u>

1. האחראי הובלות בוחר באופציה הוצאת הובלה שגרתית ("Routine transport")
2. לאחר מכן הוא בוחר אזור לספקים לביצוע הובלה מאופציות של האזורים המודפסות לו למסך.
3. במידה ובחר איזור שלא קיים רשימת האיזורים מתבטלת ההובלה והוא חוזר למסך הראשי.
4. מהספקים הקיימים באותו איזור הוא בוחר ספק אחד או מספר ספקים.
5. במידה ובחר ספק שלא קיים ברשימת הספקים מתבטלת ההובלה והוא חוזר למסך הראשי.
6. המשתמש בוחר אזור לחנויות לביצוע ההובלה מהאזורים שהמערכת הדפיסה למסך.
7. במידה ובחר איזור שלא קיים רשימת האיזורים מתבטלת ההובלה והוא חוזר למסך הראשי.
8. מהחנויות הקיימות באותו אזור המשתמש בוחר חנות או מספר חנויות לביצוע ההובלה.
9. במידה ובחר חנות שלא קיימת ברשימת החנויות מתבטלת ההובלה והוא חוזר למסך הראשי.
10. המערכת מבקשת מהמשתמש לבחור יום להובלה וסוג משמרת (בוקר/ ערב). 11. במידה וקיבלנו input לא חוקי מהמשתמש או במידה ולא קיים מחסנאי במשמרת שנבחרה בכל החנויות, המערכת מבקשת לבחור תאריך ומשמרת מחדש.
12. המערכת מדפיסה למסך את המשאיות הפנויות באותו תאריך ומשמרת והמשתמש בוחר משאית להובלה.
13. במידה וקיבלנו input לא חוקי מהמשתמש או במידה ולא קיימות משאיות פנויות בתאריך והמשמרת, המערכת מבקשת לבחור תאריך, משמרת מחדש ומשאית.

14. המערכת מדפיסה למסך את רשימת הנהגים העובדים, פנויים באותו היום והמשמרת ובעלי רשיון נהיגה מתאים למשאית, המשתמש בוחר נהג להובלה.
15. במידה וקיבלנו input לא חוקי מהמשתמש או במידה ולא קיימים נהגים אשר עובדים באותו היום, פנויים להובלה, ויש להם רשיון נהיגה מתאים למשאית, המערכת מבקשת לבחור תאריך, משמרת מחדש, משאית ונהג.
16. לאחר מכן המערכת מבקשת מהמשתמש להכניס את רשימת הפריטים אותם הוא ירצה להעביר מכל ספק לכל אחת מהחנויות.
17. במידה וקיבלנו input לא חוקי של מספרי פריטים ושמות המערכת תבטל את ההובלה ותחזור למסך הראשי.
18. לאחר מכן ההובלה נרשמה בהצלחה.

Pseudo code

```
user type “Routine transport” -> Regular_stock_transport()
transportation_service.get_area_for_suppliers()
suppliersArea <- user input
if(transportation_service.get_area_for_suppliers() not contains suppliersArea)
    print - “wrong input” throw exception;
site_service.getSuppliersbyarea(suppliersArea)
supplieresID [] <- user input
for (supplierid in supplieresID):
    if(ite_service.getSuppliersbyarea(suppliersArea) not contains supplierid)
        print - “wrong input” throw exception;
transportation_service.get_area_for_stores()
storesArea <- user Input
if(transportation_service.get_area_for_stores() not contains storesArea)
    print - “wrong input” throw exception;
site_service.get_Stores_By_specific_area(storesArea)
storesID [] <- user input
for (storeId in storesID):
if(site_service.get_Stores_By_specific_area(storesArea) not contains storeId)
        print - “wrong input” throw exception;
while(true)
    date <- user input
    shift <- user input
    if(date not Date | shift not (Morning| Evening)
        print - “wrong input” continue;
    trucks_service.getFreeTrucks(date, shift)
    if(trucks_service.getFreeTrucks(date, shift) is empty)
        print - “dont have truck to the day” continue;
    truck <- user input
    if(trucks_service.getFreeTrucks(date, shift) not contains truck)
```

```
            print - “wrong input” continue;
        drivers_service.getDriverToTrucks(truckId, date, shift)
        if(drivers_service.getDriverToTrucks(truckId, date, shift))
                print - “dont have driver to the day” continue;
        driver <- user input
if(drivers_service.getDriverToTrucks(truckId, date, shift) not contains driver)
            print - “wrong input” continue;
for (storeId in storesID):
    for (supplierid in supplieresID):
        itemList<Quantity, itemName > <- user input
    if(Quantity not number)
            print - “wrong input” throw exception;
transportation_service.createRegularTransportation().
```

G. שיבוץ עובדים למשמרת

Textual Description:

בעת יצירת משמרת, נצטרך לשבץ עובדים למשמרת בכדי שהסופר יתנהל כמו שצריך

List of actors:

אחראי עובדים

Pre-conditions:

קיום של סניף לשבץ אליו את העובדים. קיים עובד אחד לפחות במערכת שתפקידו מנהל ושהוא יכול לעבוד באותה משמרת וזה לא מתנגש עם האילוצים שלו.

Post-conditions:

המשמרת נוספה בהצלחה עם עם לפחות עובד אחד שתפקידו מנהל. תאריך המשמרת תקין ועדיין לא קרה.אין התנגשויות בין אילוצי העובדים לבין סוג ותאריך המשמרת. אין כפל עובדים(לא יכול להיות שאותו עובד משובץ באותה משמרת פעמיים)

Main success scenario

1. המשתמש בוחר בהוספת משמרת חדשה
2. המשתמש בוחר את תאריך המשמרת ואת סוג המשמרת (בוקר/ערב)
3. המשתמש יבחר את מנהל המשרת
4. המשתמש יבחר את כל עובדי המשמרת
5. לאחר מכן המשמרת תתווסף בהצלחה לרשימת המשמרות

Pseudocode

1. date ← user input
2. shiftType ← user input
3. manager SN ← user input
4. workersSNs ← user input
5. SystemInterface.createShitt(date,shiftType,managerSN,workersSNs)
6. ShiftControleer.createShift(date,shiftType,managerSN,workersSNs)
7. validate_and_parse(date)
8. valdate_and_parse(shiftType)
9. validate(managerSN)
10. validate_and_parse(WorkersSNs)

Alternatives and extensions:

1. המשתמש בוחר בהוספת משמרת חדשה
2. המשתמש בוחר את תאריך המשמרת. במידה והמשתמש מכניס ערך ריק. תרוץ לולאה עד אשר יוכנס ערך תקין
3. המשתמש יבחר את מנהל המשמרת. במידה ולא הוכנס ערך שהוא מספר תרוץ לולאה עד אשר הערך הוא יהיה מספר
4. המשתמש יבחר את כל עובדי המשמרת.
5. במידה והתאריך עבר או שכבר קיימת משמרת בתאריך זה תוצג הודעה מתאימה למשתמש
6. במידה והמשתמש בחר בעובד שלא קיים או שכבר משובץ לאותה משמרת תוצג הודעה מתאימה למשתמש

Pseudocode

1. date ← user input
2. while(date == “”)
    a. print(Invalid input, please try again)
    b. date ← user input
3. shiftType ← user input
4. while(shiftType == “”)
    a. print(Invalid input, please try again)
    b. shiftType ← user input
5. manager SN ← user input
6. while(tyoeOf(manager SN) != Integer)
    a. print(Invalid input, please try again)
    b. manager SN ← user input
7. workersSNs ← user input
8. SystemInterface.createShitt(date,shiftType,managerSN,workersSNs)
9. ShiftControleer.createShift(date,shiftType,managerSN,workersSNs)
10. if(shiftType is not typeOf(enum.shiftTypes)
    a. print(“Invalid shift type”)
11. if(shiftType is not typeOf(enum.shiftTypes)
    a. print(“Invalid shift type”)
12. if(date is valid date format)
    a. print(“Invalid date format”)
13. if(managerSN in not in managers list)
    a. print(“Invalid manager SN”)
14. for SN in workersSNs input
    a. if SN is not in workersSN list
        i. print(“There is not worker with serial {SN} number
    b. if worker starts to work after the shit date
    i. print(Worker {SN} start woking after)
    c. if date and shiftType in workers.constrains
        i. print(“{SN} can work due to his constrains)
15. createNewShiftSuccsessfuly.