תרגיל 2 –קורס ג'אווה ואינטרנט

פרטי הגשה

שם: אלון בן הרוש

ת.ז: 312533698

מייל: alonbh5@gmail.com

קישור לרפוזטורי ב GITHUB : https://github.com/alonbh5/SuperDuperMarket

הערות: **בונוסים 1,2,6**

הסבר כללי על המערכת

הפרויקט מורכב מ-3 מודלים עיקריים :

- **engine** –ה"מנוע" של המערכת, מכילה את כל הלוגיה שלה ותפקידה לנהל את המערכת בהתאם לפקודות ה-UI, מכיל 3 תיקיות עיקריות:

1. תיקיית **engine** שמכילה את מצבור המחלקות שמרכיבות את המערכת (חניות, הזמנות, מוצרים וכו'). בתוכה נמצאת המחלקה **SuperDuperMarketSystem** שהיא היחידה שחשופה (public) ורק איתה מתנהלת מערכות ה-UI השונות.
2. תיקיית **exceptions** שמכילה את מצבור החריגות שהמערכת עלולה לזרוק לUI.
3. תיקיית **generatedClasses** שנוצרה ע"י JAXB ומכילה את כל המחלקות שקבצי ה-XML עושים בה שימוש.

- **mediatorClasses** – עוזרת לתקשורת נוחה בין המערכת ל-UI, מכילה מחלקות שמייצגות אובייקטים במערכת, מבלי לחשוף את האובייקטים של מערכת המנוע עצמם. כל המחלקות שם מכילות שדות (public final) וכן public c'tor. כאשר ה-UI מבקש מידע מסוים על אובייקטים במערכת, הוא יקבל אותם בצורה של אובייקטים ממחלקה זו, וכן עבור פונקציות מסוימות המערכת מבקשת מידע בדמות אובייקטים ממודול זה.

- **console-JavaFX** –UI מסוג Desktop App שנבנה באמצעות JavaFX, בה נמצאת פונקציית ה-Main, (שנמצאת במחלקה **RunGUI** – מחלקה שיורשת מApplictionn). מכילה בתוכה את מחלקת ה- **MainMenuController** (שאחת מהשדות הפרטיים שלה הוא מערכת ה-SuperDuperMarketSystem). היא היחידה שמתקשרת עם מנוע המערכת ועם שאר ה- Controllers האחרים במערכת. היא זאת שטוענת חלונות אלה ע"י מנגון ה FXMLLoader , ודואגת למלא אותם במידע רלוונטי (שניתן לה ע"י המערכת) ולהציבם במרכז המסך המרכזי בעת הצורך.

הסבר על המחלקות העיקריות

מודול console-JavaFX

- כל מחלקה במודול זה מייצגת סצנה כלשהי, ומכילה לפחות קובץ FXML וכן קובץ Controller רלוונטי אליו.

- MainMenu – מייצגת את המסך הראשי, בעזרת ה- Controller שלה מרימה התוכנית הראשית את הסצנה לבמה. בקרי הסצנה הראשית יהיו זמינים לאחר טעינת קובץ XML תקין (שיתבצע ב- Task כנדרש ע"י המנוע), בתחתית המסך ישנם מד התקדמות, ושורת חיווי להודעות השונות מהמערכת. מלבד בקרי ה-FXML, למחלקה מספר שדות פרטיים : ישנם מספר Properties בוליאניים שתפקידם לנהל את תוכנות ה-Disable של הכפתורים השונים, כמו כן מכילה את ה- Primary Stage , ואת מנוע המערכת כאשר היא היחידה שמתקשרת בינו לבין שאר המחלקות הגרפיות.
- Resources – מכיל את קבצי ה-CSS, וכן תיקייה עם כל התמונות השונות שהCSS מפנה אליהם.
- InfoMenuBuiler– בקר פשוט שכל תפקידו להציג מידע בצורה מוסדרת, יש לו מספר מטודות שימושיות כגון הוספת חנות לרשימה, הוספת הזמנה, בפועל מרים סצנה חדשה ע"י FXML ובקרים אחרים ו"מדביק" אותם אליו.
- StoresMenu – מייצג את הצגת חנות בודדת, מכיל מידע לגבי החנות כנדרש, כמו כן מכיל 3 TitledPane שבתוכם נמצא מידע על הזמנות,הנחות,ומוצרים על מנת לא למלא את המסך במידע מיותר.
- ShowItemsMenu – מייצג את הצגת כל המוצרים, זוהי טבלה שמכילה את כל המידע הנדרש על מוצר במערכת, לאחר יצירתו ישנה מטודה שמקבלת רשימה של מוצרים ומוסיפה אותם לטבלה, את הרשימה שולחת הMainMenu (אותה היא מקבלת מהמנוע).
- Order Menu – מציג הזמנה בודדת, מכיל את כל הפרטים הדרושים לפי סעיף G8, יודע לקבל הזמנה מסוג OrderInfo ולהציגה. עולה ע"י תפריט היסטורית הזמנות (ע"י InfoMenuBuiler), בהיסטורית הזמנות של חנות בודדת, וכן בסיכום יצירת הזמנה.
- MapMenu – מייצר מפה של המערכת, באמצעות GridPane דינמי, בעת יצירה מקבל את המיקומים המקסימליים (ומוסיף להם 1 כנדרש) ומייצר ממנה טבלה, כמו כן מקבל רשימה של כי מי שמימש את הממשק Coordinatable , עובר עליהם ומוסיף אותם למשבצת המתאימה (לאחר שמבצע שינוי חישובי קטן על מנת לגרום למצב שהמשבצת השמאלית, תחתונה ביותר תהיה מיקום (1,1))
- CustomersMenu - מייצג את הצגת כל הלקוחות, זוהי טבלה שמכילה את כל המידע הנדרש על לקוח במערכת, לאחר יצירתו ישנה מטודה שמקבלת רשימה של לקוחות ומוסיפה אותם לטבלה, את הרשימה שולחת הMainMenu (אותה היא מקבלת מהמנוע).
- InputPane – מסך של קבלת מידע מספרי מהלקוח, בעת יצירה מבקש לדעת האם ברצוננו לקבל **מספר שלם או עשרוני**, וכן מבקש Runnable לביצוע לאחר שהמשתמש הכניס קלט תקין (לפני שהוא שולח הוא מוודא שהוכנסו רק מספרים בעזרת Regex). כמו כן יכול להציג אנימציות (הסבר בסוף הדף)

- ChangeItemsMenu – מסך שינוי פרטי מוצר במערכת, מבקש לדעת תחילה איזה עבור איזה חנות ברצוננו לבצע הזמנה, לאחר שבחרנו מתאפשרת (Binding) לנו האפשרות לבחור סוג פעולה בדמות 3 כפתורי Radio, לאחר שבחרנו אחד מתאפשר לנו (גם כאן באמצעות Binding) האפשרות לבחור מוצר – מוצגת לנו טבלה על גבי המסך של כל מוצרי החנות, לחיצה כפולה על מוצר תגרום לביצוע הפועלה (אם ברצוננו להוסיף/לשנות מחיר יפתח תפריט לקליטת מספר).בעת ביצירה מקבל רשימה של חניות המערכת בדמות StoreInfo שכבר מכילים מידע על כל המוצרים שנמצאים בהם, "החלפת" המסכים השונים מבוצע ע"י העמסה על גבי StockPane נסתר.
- CreateOrderMenu – מסך יצירת הזמנה, מבוצע בסדר כפי שנדרש, לאחר כל שלב מתאפשרת לנו השלב הבאה ע"י Binding, כאן החלטתי (על מנת למנוע מצב של ניסיון לחבל בפעולת המערכת..) לנטרל את כל האפשריות לשינוי סוג חנות/לקוח/סוג הזמנה לאחר שהמשתמש כבר הזין לפחות מוצר אחד שברצנו לקנות (אם המשתמש רוצה הוא יכול ללחוץ על יצירת הזמנה מחדש..). מכיל מספר קבצי FXML עבור שלב לקיחת המידע הבסיסי, הצגת חניות, הצגת הנחות והצגת סיכום הזמנה. בעת יצירה הוא מקבל קישור לתפריט הראשי MainMenu שם יוכל לבקש לקבל מידע מהמנוע (בעקיפין), וכן לפנות אליו שסיים לאסוף את המידע . "החלפת" המסכים השונים מבוצע ע"י העמסה על גבי StockPane נסתר.
- AddDiscountMenu – בונוס , מייצג מסך של הוספת הנחה למערכת, מכיל comboBox של החנות אליה רוצים להוסיף הנחה, מבקש שם של הנחה (מתעלם מרווחים ודורש לפחות 2 תווים "אמיתיים"), מבקש מוצר בסיס לקניה וסוג הנחה, לאחר מכן מציג את כל המוצרים אותם ניתן להוסיף למוצרים שניתן לקבל, אין כאן אפשרות לבחור בסוג Irrelevant שכן המנוע דואג לסמן אותו ככה במידה והתקבל מוצר בודד (רשום במסך כהערה).

מודול engine

- תיקיית **engien** עבור ניהול המערכת, הסבר קצר על רוב המחלקות בה:

- Coordinatable – ממשק שמייצג אובייקט שאחד משדותיו הם נקודה על המפה, מחייב לממש פונקציה Point getCoordinate. למערכת יש מפה שמתאימה נקודה לאוביקט Coordinatable (כרגע רק עבור חניות, בהמשך עבור הזמנות ולוקחות).
- FileHandler – משומש ע"י המערכת לגישה וקבלה של קובץ ה - XML.
- LoadXmlTask – Task שטוען למערכת את כל הפרטים מה-XML, בהתחלה ה-UI קוראת למטודה UploadInfoFromXML עם נתיב הקובץ, שם המנוע יוצר את הTASK ושולח חזרה לUI (למטודה bindTaskToUIComponents) את הTASK על מנת לחווט את כל המידע הרלוונטי (מקשר את אזור ההודעות להודעות של ה-TASK, וכן את מד ההתקדמות לTAS) וכן להגדיר מה לעשות במידה והTASK הצליח ומה לעשות במידה ולא.
- Item – מייצג פריט בסיסי, בעל מסדר סידורי, שם ואופן קנייה (enum).
- Person – מייצג אדם, יש לו **מספר סידורי** ייחודי ושם
- Customer – מרחיב את Person, בנוסף ממש את הממשק Coordinatable שכן יש לו מיקום במערכת, ובנוסף יש לו מפה של היסטורית הזמנות שביצע.
- ProductInSystem – מייצג פריט שמור במערכת , מכיל הפנייה לפריט הבסיס היחיד **item** (וכך מקבל את כל פרטיו), בנוסף מכיל נתונים שמתעדנים ע"י המערכת כמו **מספר חניות שמוכרות** את הפריט במערכת, מספר **פעמים שנמכר** במערכת, וכן נכון לחלק זה של התרגיל **הפנייה לחנות שמוכרת את הפריט במחיר הזול ביותר** (מתעדכן ביצירת המערכת מקובץ הXML, וכאשר מתבצע שינוי כלשהו על הפריט). זאת במטרה לייעל את ההזמנה הדינאמית בבונוס התרגיל (ניסיתי להימנע כרגע מלחפש כל פעם את הפריט הזול ביותר, עלול להשתנות בהתאם להמשך התרגיל).
- ProductInStore - מייצג פריט שמור בחנות , מכיל הפנייה לפריט הבסיס היחיד **item** (וכך מקבל את כל פרטיו), כמו כן שם נמצא המחיר שלו **בחנות**, כמות **הפעמים שהמוצר נמכר וכן מצביע לחנות** שמוכרת אותו.
- ProductInOrder – מייצג פריט בתוך הזמנה , מכיל הפנייה לפריט סופי מסוג **ProductInStore** (וכך מקבל את כל פרטיו), כמו כן כאן נמצאים הכמות שנקנה בה בהזמנה והמחיר של הפריט X כמות, בנוסף ערך בוליאני האם הגיע מהנחה או לא.
- Store – מייצגת חנות במערכת, בעלת **מספר סידורי** ייחודיי, שם מיקום, רווח ממשלוחים וPPK. בנוסף מתוך מחשבה על יעילות חיפוש מהירה יש לה Map של היסטורית הזמנות מסוג **<Order, מס' הזמנה>**, וכן Map של מוצרים בחנות מסוג **<ProductInStore, מס' מוצר>** וסט של הנחות שהחנות מציעה. מממשת את הממשק Coordinatable,
- Order – מייצג הזמנה במערכת, מכילה **מספר סידורי** ייחודי, את שאר פרטי ההזמנה כמו לקוח, תאריך וסכומי קנייה , כמו כן מכילה **סט של חניות** שהשתתפו בהזמנה, **סט של מוצרי ProductInOrder** (שכן אין חשיבות לסדר בהזמנה ואין טעם לשמור מספר חניות/מוצרים זהים) בנוסף היא מממשת את הממשקים **Coordinatable**.
- SuperDuperMarketSystem – המערכת הראשית איתה עובדת ה-UI, יש לה מספר פונקציות עזר private אבל הרוב כן חשופות ל-UI, מכילה שדה בוליאני

"נעול" שיישתנה רק לאחר שיטען קובץ XML תקין, השדות האחרים שלה כולם מפות מתוך מחשבה שלכל פריט במערכת יש מסדר סידורי ייחודי וכך יהיה ניתן לגשת אליו ב-$\theta(1)$.

- **<מס' מוצר, ProductInSystem>** עבור מוצרי המערכת השונים.
- **<מס' חנות, Store>** עבור החניות השונות.
- **<מס' הזמנה, Order>** עבור מוצרי המערכת השונים.
- **< Point,Coordinatable>** עבור ייצוג מפה להמשך, כרגע מכיל רק חניות.

- Discount - מייצג הנחה, להנחה יש שם , סוג הנחה (eNum), ProductYouBuy וסט של ProductYouGet.
- ProductYouBuy – מייצג מוצר שאתה צריך לקנות על מנת לקבל הנחה, יש לו בפנים את מוצר הבסיס Item, וכמות אותה צריך לקנות
- ProductYouGet – מייצג מוצר שאתה יכול לקבל מההנחה, יש לו בפנים את מוצר הבסיס Item, מחיר לפריט אחד בהנחה וכמות של מוצרים אותה ניתן לקבל.

- תיקיית **exceptions** עבור החריגות שהמערכת עלולה לזרוק לUI. בחרתי לסמן את כל החריגות כ-Checked על מנת לזכור לטפל בחריגות ברמת ה-UI, רשימת החריגות בקצרה, רובם נזרקים בקבלת קובץ ה-XML:

- DuplicateItemIDException – התגלו 2 פריטים עם אותו מסדר סידורי
- DuplicateItemInStoreException – חנות מוכרת 2 פריטים זהים
- DuplicatePointOnGridException – חנות/הזמנה מנסה לגשת למיקום תפוס
- DuplicateStoreInSystemException – התגלו 2 חניות עם אותו מסדר סידורי
- ItemIsNotSoldAtAllException -יש מוצר במערכת שלא נמכר בכלל
- NegativePriceException – התקבל מחיר שלילי או אפס.
- NoValidXMLException – אין קובץ XML תקין במערכת, אן שהמערכת נעולה.
- PointOutOfGridException – התקבלה נקודה מחוץ לטווח המוגדר
- StoreDoesNotSellItemException – יש חנות שלא מוצר שום פריט
- StoreItemNotInSystemException – חנות מנסה למכור מוצר שלא במערכת
- WrongPayingMethodException – התקבלה אופן קנייה לא תקין (בקובץ)
- CustomerNotInSystemException-התקבל לקוח שלא נמצא במערכת
- DuplicateCustomerInSystemException– התקבל לקוח עם אותו ID פעמיים.
- ItemIsTheOnlyOneInStoreException– מנסים למחוק מוצר אחרון בחנות.
- IllegalOfferException– התקבל סוג הנחה לא תקין.
- NegativeQuantityException– התקבלה כמות לא תקינה בהנחה.
- NoOffersInDiscountException– התקבלה הנחה ללא הצעות.
- OrderIsNotForThisCustomerException– מנסים להוסיף הזמנה להיסטורית הזמנות של לקוח כאשר ההזמנה לא בוצעה על ידי אותו לקוח.

מודול mediatorClasses

כפי שתואר לעיל, מודול זה מחזיק מספר מחלקות שמייצגות את האובייקטים השונים של המערכת בגרסה פשטנית יותר, המחלקות מייצגות פריטים, חניות והזמנות כאשר רוב השדות בהם הם פרימיטיביים או אוסף של מחלקות אחרות מתיקייה זו, כמו כן כולם public final ובעלי public ctor. יש 2 מטרות עיקריות עבור מחלקות אלו:

1. שליחה מה-UI למנוע המערכת – חלק מהמטודות במערכת המנוע דורשות לקבל אובייקטים ממודול זה, למשל ביצירת הזמנה על מערכת ה-UI לשלוח **אוסף של פריטים** רצויים להזמנה, המערכת תדע לקבל את המידע ולהמיר אותו לפריטים הרלוונטיים אצלה, וזאת מבלי לחשוף ל-UI את הפריטים עצמם ובכך שומרת על כך שתפקיד ניהול המידע נשאר רק לה. כמו כן תדע להוציא חריגות מתאימות במידע וקלט לא תקין הגיע.
2. שליחה מהמנוע ל-UI – עבור כל מטודות קבלת המידע של המערכת (getters למיניהם), המערכת תייצר עותק שמייצג את המידע הרלוונטי ותשלח אותו ל-UI לעשות בה כרצונה, בכך נמנע מצב שה-UI תוכל לפנות למידע רגיש במערכת – שכן היא מקבלת רק העתק פשטני.

מימוש הבונוסים

- בונוס 1 – אנימציות: ממימוש הבונוס נעשה כנדרש כאשר מוסיפים מוצר חדש להזמנה, המימוש נכתב במחלקה InputPane, שכברירת מחדל לא מבצעת אנימציות, ועל מנת להפעילם היא דורשת את המיקום אליו האנימציה תלך, וערך בוליאני האם לבצע אנימציות (כפתור הצ'ק במסך הראשי) .כאשר מוסיפים כמות של מוצר המוצר "מתעופף" לתחתית המסך.
- בונוס 2 – החלפת SKIN: במסך הראשי ישנו ComboBox איתו ניתן להחליף ל-3 סוגי סקינים שונים, בפועל כל שינוי מחליף את קובץ ה-CSS לזה שנבחר, השינויים הם ברקע בראשי , תמונות המפה וההנחות, גופן ומראה של כפתורים.
- בונוס 6 -הגדרת מבצעים: במסך הראשי בכפתור האחרון ישנה בחירה להגיע לתפריט "הוספת מבצע", בוא ניתן לבחור את החנות שאליו רוצים להוסיף מוצר, ולהגדיר את כל פרטי המוצרים של ההנחה. מנוע המערכת מקבל את המידע בדמות DiscountInfo ומייצר הזמנה חדשה בחנות (זורק חריגות במידה ויש), כברירת מחדל אם התקבל רק מוצר אחד לבחירה – סוג ההנחה יהיה IRRELEVENT .