# קבוצה 4:

**יזהר אננייב** [Izhar\_ananiev@hotmail.com](mailto:Izhar_ananiev@hotmail.com)

**ליאור גל** [liorgal28@gmail.com](mailto:liorgal28@gmail.com)

**כפיר ביטון** [kfirbitonn@gmail.com](mailto:kfirbitonn@gmail.com)

**איתי עלמני** [itaialmani@gmail.com](mailto:itaialmani@gmail.com)

**רון בן-צבי** [benzvi.ron@gmail.com](mailto:benzvi.ron@gmail.com)

**הוגש בתאריך:** 17/12/2017

1. **תארו את תהליך התכנון שביצעתם לתהליך ביטול הזמנה ע"י הלקוח. פרטו מה הם הדילמות התכנוניות שהתייחסתם אליהם? מה היו החלופות ומה השיקולים לבחירת הפתרון שיושם? בתשובה התייחסו לעקרונות שנלמדו בהרצאות: ,Reuse ,Design ,Architectureו Design .patterns**

כאשר לקוח רוצה לבטל הזמנה, עליו להזין את מספר ההזמנה.

לאחר ההזנה, ראשית המערכת תבדוק האם ההזמנה קיימת במסד הנתונים, על ידי תקשורת עם ה-server.

בנוסף המערכת תבדוק האם ההזמנה המצוינת נוצרה על ידי המשתמש המבקש לבטלה – המערכת תשלוף את פרטי המשתמש מה-Context, ותשווה את פרטי המשתמש המקושר להזמנה למשתמש הקיים (Architecture Design). המחלקה Context מכילה את המידע הנעשה בו שימוש במספר מחלקות שונות, של כל Client. כך פתרנו את הדילמה כיצד לקשר בין הלקוח לבין ההזמנה שלו על מנת שלא ייווצר מצב שלקוח ביטל הזמנה של לקוח אחר.

כעת, המערכת תבדוק את ההפרש בין זמן ביטול ההזמנה לבין זמן המשלוח המבוקש ותבצע החזר כספי ללקוח לפיו הפרש הזמנים, על פי סיפור המערכת. הפיצוי מתבצע על ידי הוספה של הסכום המתאים אל חשבון התשלום של הלקוח. ניגש לחשבון הלקוח על ידי שליפת פרטי המשתמש הנוכחי ממחלקת ה-Context.

בשלב זה, המערכת תשנה את סטטוס ההזמנה ל-"canceled". על מנת להפיק דוחות של הזמנות, ולהציג בהם גם את ההזמנות המבוטלות, לא נמחק הזמנות שבוטלו מן המסד, אלא נשנה את סטטוס ההזמנה.

לבסוף, נעדכן את הנתונים השונים במסד הנתונים, ונציג הודעה בהתאם.

* 1. **ציינו עקרון של גישת התכן : Object-Oriented Design שאפשר לכם לשלב קטעי קוד ומרכיבים אחרים שלא אתם כתבתם או תכננתם. תארו בדיוק (ובהתייחסות ספציפית) ובפירוט את התכונות של אותם מרכיבים אשר אפשרו לכם לשלבם במערכת שלכם, תוך התייחסות בדוגמאות ספציפיות (לא 'עקרוניות)' לדרישות הפונקציונליות של המערכת שתכננתם (התייחסות ספציפית בהקשר זה = התייחסות למרכיבים ספציפיים מתוך התיאור המילולי הראשוני של המערכת ששאתם מפתחים מהתחלת הסמסטר).**

\* בתשובה לשאלתנו אנו מתייחסים **למימוש האב-טיפוס**.

מסמך התיאור המילולי של המערכת מכתיב לנו לאפשר למספר משתמשים שונים להיות מחוברים בו זמנית למערכת, בנוסף נאמר כי המערכת תהיה בעלת אופי תפעולי מבוזר כך שניתן יהיה לעבוד מתחנות עבודה בו זמנית על ידי משתמשים שונים.

על מנת לממש את הדרישות הללו השתמשנו בעקרון הרב צורתיות (פולימורפיזם)

עקרון זה מהווה חלק מרכזי בגישת התכן Object – Oriented Design ובא לידי ביטוי על ידי הגדרת מחלקות חדשות היורשות מהמחלקות הכלליות : AbstractServer, AbstractClient.

כלומר השתמשנו למעשה במחלקות מופשטות המאפשרות לנו להשתמש בקוד כללי הפועל על סוגים רבים ושונים של מושגים הדומים להן, נציין בנוסף כי גם בתרשימי ה – UML שלנו מבוצע פוליפורמיזם, לדוגמא: בתרשים המחלקות שלנו (Class Diagram), ניתן לראות כל כל מחלקה שהיא למעשה Entity (ישות) יורשת מהמחלקה User שהיא למעשה מחלקה מופשטת (Abstract class).

על ידי שימוש בעקרון הפוליפורמיזם שילבנו למעשה קטעי קוד קיימים (מתודות שונות) שהיו נחוצים לנו לצורך עבודה נכונה עם מסד הנתונים שלנו ובנוסף נציין כי שימוש בעקרון הפוליפורמיזם המצויין בתרשים המחלקות יסייע לנו רבות בשלב מימוש המערכת שלנו, בפרט כאשר נרצה לממש את כלל הישויות הקיימות במערכת זר-לי.

* 1. **ציינו לפחות עקרון אחד של OOD שלא בא לידי ביטוי במודלים שלכם - והסבירו למה.**

\*בתשובה זו התייחסנו **לתרשימי ה- UML**.

העקרון שלא בא לידי ביטוי בתרשימי ה – UML שלנו הוא עקרון ההורשה.

משמעותו של עקרון זה היא כאשר מחלקה יורשת ממחלקה אחרת היא מעניקה לאובייקטים שלה גם את המימוש הפנימי של המחלקה האחרת.

מטרתו העקרית של עקרון ההורשה הוא הרחבה של מחלקות קיימות על ידי הוספת מתודות למחלקה היורשת ובמקרה הצורך גם ביצוע דריסה של מתודות הקיימות של מחלקת האב.

עקרון ההורשה לא בא לי ביטוי במודלים שלנו מכיוון שאין לנו צורך לבצע הורשה של מחלקות במערכת זר לי, "ההורשה" שאנו עושים (כפי שמתואר בתרשים המחלקות) היא למעשה הורשה ממחלקות אבסטרקטיות (כפי שצויין בסעיף א') וזהו למעשה מימוש של עקרון הפולימורפיזם.

מכיוון שאין לנו באמת צורך ליצור אובייקט מסוג User שהרי כל User הוא :

לקוח, מנהל חנות, מוכר בחנות, מנהל רשת, עובד שרות לקוחות וכדומה.

* + 1. **הערכה כללית:**
    2. **מהם היתרונות של מודל UML כעזר לתהליך התכנון? ציינו דוגמה אחת קונקרטית (לא כללית) מתוך התהליך שאתם בצעתם לשימוש כזה ב- UMLתוך תיאור והתייחסות ספציפית למרכיבים של מערכת "זר-לי" שתכננתם ומידלתם.**

יתרונתיו של מודל ה UML כעזר לתהליך התכנון הם:

* התמקדות בתחום הידע של המשתמש
* התאמה לדרך החשיבה של המשתמש
* תרשימים ידידותיים יותר למשתמש

היתרון העקרי שבא לידי ביטוי בתהליך המידול הנו ביצוע התאמה לדרך החשיבה של המשתמש.

כפי שידוע תהליך המידול כולו החל למעשה לאחר קבלת מסמך דרישות מהלקוח, במסמך זה מפורט למעשה מהם הדרישות שהלקוח רוצה ממערכת המידע כלומר, מהם הפעולות שהמערכת מספקת עבור המשתמש. מסמך הדרישות כפי שניתן לנו הוא בדרך החשיבה של המשתמש, לא מפורטים בו נתונים טכניים של כיצד לממש דברים במערכת (אילו מחלקות, איזו שפת תכנות, איזה מסד נתונים וכד').

ניתן לשים לב שתהליך המידול שלנו וכלל תרשימי ה UML השונים הם קריאים למדי ,ורמתם מותאמת לדרך החשיבה של המשתמש.

לדוגמא אופן ביצוע תהליך הLogin מתואר בפשטות רבה (משתמש מזין : שם משתמש וסיסמא, המערכת בודקת את הנתונים ומחזירה הודעה מתאימה), שזהו תיאור שללקוח קל מאוד להבין.

דוגמא נוספת לכך היא ביצוע הזמנה, גם במקרה זה תיאור התהליך הינו פשוט ככל הניתן (המשתמש בוחר מוצר, לוחץ על ביצוע הזמנה, לאחר מכן בוחר אופן תשלום, ולבסוף את סוג המשלוח), ניתן לראות שבמקרה זה התהליך ברמת המימוש הוא יותר מסובך ועליו לכלול הרבה מאוד בדיקות ומקרי קצה, אך בתהליך המידול שלנו אנו מראים זאת בדרך יותר מפושטת מכיוון שללקוח אין אינטרס לדעת כיצד המערכת מבצעת בפועל את ההזמנה מבחינתו חשוב רק שההזמנה אכן בוצעה והתשלום הועבר באופן תקין.

* + 1. **ציינו קשיים הנובעים מחסרונות של UML שנתקלתם בהם. גם כאן התייחסו ספציפית לתהליך שבצעתם בפרויקט זה.**

חסרונותיו של מודל ה UML כעזר לתהליך התכנון הם:

* ריבוי מודלים וכלים.
* מורכבות הקשר בין המודלים.
* קושי בשימוש במערכות מורכבות.

הקשיים בהם נתקלנו בתהליך המידול שלנו הנובעים מחסרונותיו של מודל ה UML הנו ריבוי המודלים הרב שנדרשנו להשתמש בהם והקושי בשימוש במודל הנ"ל במערכת מורכבת כמו מערכת זר-לי.

לאורך תהליך המידול שלנו השתמשנו בחמישה כלים ודיאגרמות שונות הנכללים במודל ה UML.

הדבר הקשה עלינו מהבחינה שקיים קשר בין דיאגרמות שונות וכך לכל אורך התהליך יש צורך לבדוק אם בעת מידול דיאגרמה חדשה לא נפגע המידול של הדיאגרמות הקודמות לה,  
(Class Diagram, Activity Diagram ,Sequence Diagram)

מה שמצריך עבודה רבה יותר ולעיתים אף חזרה לאחור לצורך תיאום אופן המימוש בין הדיאגרמות השונות.

בנוסף, מערכת זר-לי הנה מערכת מורכבת המכילה פעולות רבות ותרחישים שונים דבר המקשה מאוד על תהליך המידול מכיוון שיש צורך לקחת בחשבון את כלל הפעולות השונות של המערכת והדבר לעיתים עלול לגרום לדיאגרמות השונות להיות עמוסות במידע וכתוצאה מכך הן הללות להיות לא קריאות וקשות מאוד להבנה על ידי הלקוח או על ידי גורם חיצוני שלא היה נוכח בתהליך המידול.

1. **ניתוח ודיון:  
   בהתאם לניסיון שרכשתם במהלך העבודה על מטלה זו, תארו אפשרויות לשינויים ושיפורים במתודולוגית UML אשר נותנים מענה לחסרונות שנתקלתם בהם במהלך ה-** design**שביצעתם בפרויקט שלכם. הסבירו את תשובתכם תוך תיאור דוגמה ספציפית (כולל שמות של רכיבים) מתוך עבודתכם.**

במתודולוגיה שבה השתמשנו אנחנו משתמשים בסוג של "מפל מים" כאשר ביצוע של דיאגרמה אחת מהווה תלות בביצוע דיאגרמה אחרת.

לדוגמה, על מנת לממש תרשים רצף עלינו לממש (אפילו רק גרסה ראשונית) של תרשים מחלקות, מתוך תרשים הרצף עולות הרבה תהיות אשר באות לידי ביטוי גם בתרשימים שהתבצעו לפני.

דוגמא נוספת הנה מימוש זרימת האירועים בכל Use-Case,במהלך עבודתנו על תרשים הרצף הבנו שהיה צורך בשינויים קטנים אך כאלו שיכלו לעשות שינוי גדול באופן הבנת הקורא.

אנו היינו מבצעים עבודה בשיטה שונה במקצת, שבה ישנה האופציה לעבוד על דברים בו זמנית.

כמובן שאין חד משמעית נכון מכיוון שלכל מתודולוגיה ישנן נקודות החוזק ונקודות החולשה שלה, לכן זה מאוד סובייקטיבי.