חברי הקבוצה:

שלומי עמר	308438084	shlomia1074@gmail.com
אופיר נוימן	313539546	tuphr2234@gmail.com
אלון איבשין	205843899	alon.ivs94@gmail.com
אופיר אברהם ווקנין	206487274	ofirvaknin55@gmail.com
ליאור קרן	318549185	liorkeren15@gmail.com

שאלות ותשובות:

.1

1. תארו את אופן השימוש במודל Use Case בעבודתכם (מבחינה מהותית – לא טכנית): הסבירו את מקומו ותרומתו של המודל לתהליך הפיתוח הכולל של המערכת בעזרת דוגמאות **פרטניות** (ספציפיות) מהמערכת "GoNature " (לא Login).

תשובה:

השימוש ב-use case מעניק סיכום מקיף על הפונקציונליות על מה שהמערכת שלנו מספקת ומאפשרת למשתמשים בה ולא על איך היא מממשת את הפונקציונליות הזאת.

דיאגרמת use case נותנת מבט כללי, מבט "מלמעלה" על הפונקציונליות של המערכת מבלי להיכנס לעומק הפרטים.

- הדיאגרמה כתובה במילים שכל אדם יכול להבין, כלומר גם אנשים ללא רקע בתכנות (לדוגמה: לקוחות של המערכת, לקוחות קצה וכו') יכולים להביט בדיאגרמה ולקבל מושג כללי על מה המערכת מספקת.
  - ❖ לדוגמה בדיאגרמה שלנו, ניתן ע"י הסתכלות מהירה וללא שום ידע בתכנות לראות שמטייל יכול לבצע הזמנה של ביקור בפארק או שעובד שירות יכול לבצע רישום של משתמשים למערכת.

- כמו כן, use case model יכול לספק לנו נקודת התחלה טובה להיבטים אחרים בשלבי הפיתוח של המערכת, כמו הערכת עלות בניית המערכת , תכנון הפרויקט, הכנות לבדיקות והכנת דוקומנטציה למשתמשים.
  - ❖ לדוגמה ע"י הסתכלות על הדיאגרמה ניתן להבין כי יש רישום של מנויים למערכת, כלומר יש צורך גם בשמירת מידע זה לצורך התחברות, קבלת הנחות ועוד, לכן ניתן כבר להבין שיש צורך ככל הנראה ב-database שימוש לנו מידע זה.

כלומר ע" מודל ה-use case אנו מקבלים תמונה רחבה של מה נחוץ לפרויקט וכך ניתן גם להעריך עלויות עתידיות.

- ❖ דוגמה נוספת, אם העובד שירות מבצע רישום של מנוי למערכת, ואמרנו שיש צורך בשמירת המידע הזה ב-database, ניתן כבר מידית להתחיל לתכנן test cases עבור הרישום ולתכנן מה אנו מצפים שיקרה ממצבים ומקרים שונים בתהליך ההרשמה.

- בנוסף, use case model מאפשר לנו לתאר את רצף האירועים המרכזי ובנוסף רצפים אלטרנטיביים אחרים.
  - ❖ לדוגמה, ניתן לתאר את רצף האירועים המרכזי של הזמנת ביקור כך:

כניסה למערכת -> לחיצה על הזמנת ביקור -> מילוי פרטים -> ניתן לבצע הזמנה -> מעבר לתשלום -> מילוי פרטים -> תשלום -> הצגת אישור הזמנה ושליחת אישור ההזמנה במייל ובפלאפון.

וניתן לתאר גם רצף חלופי אפשרי:

כניסה למערכת-> לחיצה על הזמנת ביקור -> מילוי פרטים -> לא ניתן לבצע הזמנה -> בחירת מועד חלופי -> מעבור לתשלום -> מילוי פרטים -> תשלום -> הצגת אישור הזמנה ושליחת אישור ההזמנה במייל ובפלאפון.

.2

2. תארו בפירוט איזה מרכיבים פונקציונליים ספציפיים של האופיין של מערכת "GoNature" (כפי שמתואר במסמך "Semester Project") לא הצלחתם לבטא בעזרת מודל UC ? מה הסיבה (או הסבר) לאי הצלחה זו? מה מאפיין את המרכיבים האלה?

תשובה:

ה- Use Case מתאר פונקציונליות אשר משתמשים יכולים לבצע במערכת, לדוגמה:

- הזמנת ביקור,
- התחברות/הרשמה
- שינוי מספר מבקרים מקסימלי

אך ה- Use case נכשל בהצגת הפונקציונליות שהמערכת עצמה מבצעת באופן אוטומטי, לדוגמה:

- המערכת יודעת לבטל אוטומטית הזמנות שלא אושרו בזמן
- המערכת שומרת הזמנה למי שבראש רשימת ההמתנה למשך שעה
- המערכת יודעת להעביר את ההזמנה לבא ברשימת ההמתנה באם האדם הקודם ברשימת ההזמנה לא אישר את ההזמנה.
- המערכת יודעת לבצע רישום כניסה ויציאה של מבקרים
- המערכת יודעת לשייך את כמות המבקרים הקשורים למספר מזהה ספציפי

כמו כן, **ה-Use Case אינו מציג דרישות/הגבלות של המערכת** מכיוון שהוא שם דגש על השחקנים במערכת ועל הפעולות שהם יכולים לבצע במערכת, לדוגמה:

- "אותו משתמש לא יכול להיות מחובר למערכת בו זמנית יותר מפעם אחת".
- "יכולים להיות מספר משתמשים שונים המחוברים בו זמנית למערכת".

בנוסף **ה- Use Case נכשל בהצגת סדר האירועים ותזמונם**, לדוגמה:

- מנהל הפארק יכול להציע שינויים, אך מנהל המחלקה צריך לאשר זאת לפני ששינויים אלו יכנסו לתוקף. לא ניתן לדעת לפי המודל שיש רצף בין הפעולות.
- מבקר אשר ביצע הזמנה צריך לאשר את ההזמנה יום לפני הביקור. לא ניתן לדעת זאת לפי המודל. לפי המודל ניתן לדעת שלמבקר יש אפשרות לבצע אישור הזמנה וביטול הזמנה, אך לא מתואר מתי הוא יבצע את אישור ההזמנה.

דבר אחרון, **ה- Use case לא מציג לנו מערכות חיצוניות באופן אופטימלי**, כמו מערכת הקורא כרטיסים ומערכת המידע של העובדים. לפי המודל לא ניתן לדעת כיצד המערכות החיצוניות מתממשקות עם המערכת שלנו, לא ניתן לדעת איך המערכות האלה בנויות, מי מתחזק אותן ועוד.

.3

3. בתשובות 1 ו-2 תיארתם יתרונות ומגבלות שונות של מודל UC. הציעו דרכים להתגבר על המגבלות שציינתם, ונמקו למה הצעותיכם נותנות מענה למגבלות אלה.
הסבירו את תרומתה של הגישה שאתם מציעים כאן לפתרון אותן המגבלות שתיארתם תוך התייחסות ישירה למערכת "GoNature " ובסיוע דוגמאות **פרטניות** (ספציפיות) מהמערכת.

תשובה:

ניתן להתגבר על המגבלות שהעלנו על ידי הוספת תרשימים נוספים ל- UML שלנו.

על ידי שימוש **ב- Class diagram בשילוב עם Activity diagram ו - Sequence diagram** ניתן לפתור את הבעיות שהצגנו.

ב- **Class diagram** לתאר את כל המחלקות והאובייקטים שמשתתפים במערכת שלנו, ניתן לתאר את מחלקות ה-Control במערכת ואילו פעולות הן יכולות לבצע וניתן לתאר את החלק הגרפי של המערכת – מה קורה בכל layout במערכת שלנו.

ב-**Activity diagram** ניתן לראות את זרימת העבודה של המשימה, את הפעולות שהמערכת עושה בכל משימה ספציפית, את הבדיקות שהיא מבצעת, אילו שחקנים משתתפים בכל משימה, ואילו חלקים ספציפיים של המערכת משתתפים במשימה.

בעזרת ה- **Sequence diagram** ניתן לתאר את **האופן** שבו המערכת מבצעת את המשימה הספציפית. התרשים מציג לנו את רשימת המסכים (layouts), הישויות והשירותים הלוגיים שאותו תהליך משתמש.

בעזרת שלושת הדיאגרמות האלו, ניתן "לכסות" על המגבלות שהצגנו, לדוגמה:

- ההגבלה כי "אותו משתמש לא יכול להיות מחובר למערכת בו זמנית יותר מפעם אחת" אינה באה לידי ביטוי ב-Use Case אך ניתן למדל הגבלה זאת ב-Activity diagram וגם ב- sequence diagram מפני שבדיאגרמות אלו ניתן להציג את הבדיקות שהמערכת שלנו מבצעת.
כמו כן, יהיה ניתן לראות ב- Class diagram כי ל- Control שיהיה אחראי על ביצוע Login תהיה אפשרות להפעיל פונקציה שבודקת האם המשתמש מחובר כבר במחשב אחר.
- אמרנו כי ל- Use Case יש בעיה להציג את רצף האירועים והצגנו את הדוגמה כי מבקר אשר ביצע הזמנה צריך לאשר את ההזמנה יום לפני הביקור.
בעזרת Activity diagram ניתן להציג את הרצף הנכון של תהליך זה.