

1. **היתרונות של מודל UML כעזר לתהליך התכנון:**

דיאגרמות UML מהוות סטנדרט שמשמש לתאר בצורה ויזואלית וחזותית תוכנה, בעיקר עבור תוכנה מבוססת פיתוח מונחה עצמים. הדיאגרמות עוזרות לארגן, לתכנן ולראות את התוכנה. בנוסף לכך, כיוון שדיאגרמות אלו מהוות סטנדרט פיתוח הרי שהן משומשות ומקובלות בתעשייה כדרך לתאר תוכנות ומשמשות בתור שפה משותפת בין מתכנתים. כמו כן יתרון נוסף הוא הנוחות שבקריאת הדיאגרמות והאפשרות לשימוש חוזר בחלקים נרחבים מהדיאגרמה בפרויקטים שונים.

* **תיאור חזותי של המידע:**

דיאגרמת Class היא בעצם תיאור חזותי של הקשרים בין המחלקות והישויות בתוך אפליקציות ותוכנות מחשב. מחלקה היא בעצם אובייקט תכנותי שמסדר משתנים ומתודות במקום אחד על פי הקשר ביניהם. בכדי להבין את הקוד והתוכנה, זה הכרחי להבין מה כל מחלקה עושה, המידע שהיא שומרת וכיצד זה בא לידי ביטוי בתמונה הגדולה ביחס לשאר המחלקות בתוכנה. על ידי הצגת המידע בדיאגרמה, ניתן לראות ולהבין בצורה פשוטה את הקשרים בין החלקים והמחלקות השונות בתוכנה.

* **נוחות קריאה:**

דיאגרמות UML הן מאוד קריאות, הן נועדו להיות מובנות על ידי מפתחים מתחומים שונים ועוזרות להסביר את הקשרים של התוכנה בצורה פשוטה ומובנת. באופן מסורתי, בכדי להבין תוכנה, על המפתח לקרוא את הקוד בצורה ישירה. כאשר מדובר בתוכנה גדולה ייתכן ומדובר במאות אלפים ואף מיליונים של שורות קוד. כאשר ניתן לחסוך את זמן ההבנה בהרבה על ידי התבוננות בדיאגרמות אשר ממחישות את הקשרים והפעולות המתבצעות.

* **שימוש חוזר:**

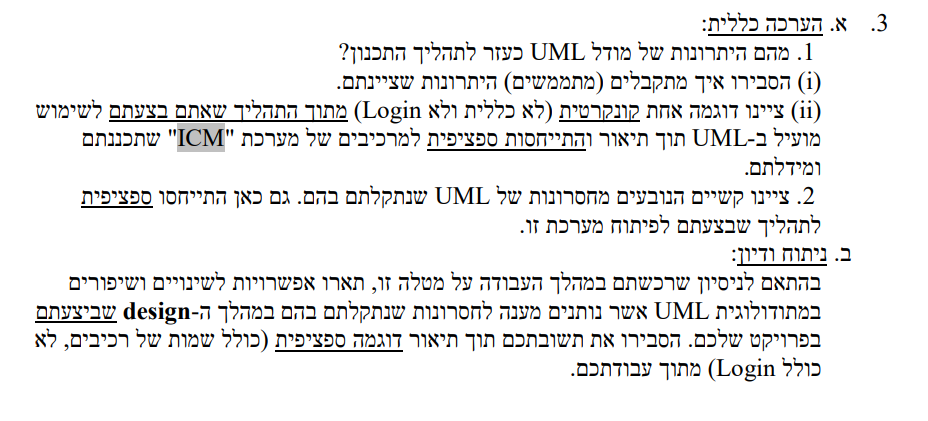
באמצעות דיאגרמות של Activity וSequence ניתן למדל ולהראות קטעי קוד שרצים ופועלים בתוכנה. מפתח התוכנה יכול לזהות ולראות קטעי קוד מיותרים או לחילופין קטעי קוד שחוזרים על עצמם, ובכך בעת תהליך המימוש יוכל לבצע שימוש חוזר (re-use) בקטעי קוד אלה במקום לכתוב את אותן פונקציות מחדש ובכך לחסוך זמן.

* **סטנדרט פיתוח:**

דיאגרמות UML מהוות בעצם סטנדרט או פרוטוקול פיתוח עבור פיתוח תוכנות בשפות מונחות עצמים. כאשר יוצרים מחלקות ואובייקטים עם קשרים ביניהם, דיאגרמת UML היא מה שמאפשרת לנו לתאר בצורה ויזואלית את קשרים אלו. כיוון שהדיאגרמות משמשות כסטנדרט הן ניתנות להבנה על ידי מפתחים רבים, ובפרט זה מאפשר למפתחים חדשים לקפוץ לתוך הפרויקט בקלות ולהיות מועילים ופרודוקטיביים מהיום הראשון בעבודה.

* **כלי תכנון:**

דיאגרמות UML עוזרות לתכנן את התוכנה לפני שמבצעים את שלב המימוש והתכנות. בכלי שאנו משתמשים ישנן דרכים לבצע יצירת קוד בצורה אוטומטית על בסיס הרכיבים שתוארו בדיאגרמה. עובדה זו יכולה לעזור לחסוך זמן בשלב המימוש של התוכנה. כמו כן ניתן לבצע שינויים בדיאגרמות בצורה פשוטה בעוד שלבצע שינויים ותיקונים בקטעי קוד יכולה להיות עבודה סיזיפית אשר גוזלת זמן רב. לכן מומלץ לבצע תכנון של הפרויקט באמצעות הדיאגרמות.



**יתרון ספציפי של הדיאגרמה:**

בפרויקט שלנו ישנו תהליך מרכזי וחשוב של הגשת בקשה וטיפול בבקשה. כאשר הטיפול בבקשה מורכב ממספר בעלי תפקידים שונים אשר כל אחד מהם מבצע שלב אחד בודד מתוך תהליך גדול. לאחר ניתוח הסיפור והדרישות שנובעות ממנו, הצלחנו בצוות לאבחן ולנתח שתהליך זה מהווה את לב ליבה של המערכת והפונקציונליות העיקרית שלה. כלומר ישנה חשיבות גבוהה ביותר להבנה של התהליך ומימושו בצורה שתענה על דרישות הלקוח וכמובן תאפשר ניצול אידאלי ומיטבי של משאבינו. כלומר מימוש שיאפשר לנו לבצע שימוש חוזר בקטעי קוד שונים ולפשט את הפונקציונליות לתתי פונקציות אשר כל אחת מהן עושה משהו בסיסי ופשוט ובכך לאפשר לנו קוד קריא יותר וגם נוח למימוש בסביבת עבודה של צוות כאשר העבודה הינה שיתופית.