תשובות לשאלות פתוחות
מטלה 2 – קבוצה 10

email	תעודת זהות	שם
mor.shmuel@e.braude.ac.il	315040980	מור שמואל
dor.david.shabat@e.braude.ac.il	316575620	דור שבת
lior.zucker@e.braude.ac.il	316375187	ליאור צוקר
yuval.rozner@e.braude.ac.il	207756552	יובל רוזנר

שאלה 1:

תארו את תהליך הניתוח/תכן ראשוני שביצעתם למרכיב: בדיקת מבחנים.
קוים מנחים לתשובה:

- פרטו מה הם הפרטים / השאלות שהתייחסתם אליהם בתהליך הניתוח והחשיבה והתשובות הספציפיות שנתתם להן.
- התייחסו לקשרים ולמעברים:

1. ממודל תרחישי שימוש Use-case למודל תהליכי מפורט המיוצג בעזרת Activity Diagram,.
2. ממודל הפעילויות והתהליכים לתיאור המימוש בתוכנה (Design).

1. ממודל ה Use Case ל- Activity Diagram:
כאשר כתבנו את ה UC, תיארנו את התקשורת בין המערכת שלנו למשתמשים שלנו על ידי תרשים זרימה Flow Of Events.
כאשר יצרנו את מודל ה Activity, הרחבנו את פירוט תרשים הזרימה שתואר ב UC בכך שציינו מי לוקח חלק בכל פעולה (באמצעות swimlanes).

2. ממודל הפעילויות והתהליכים לתיאור המימוש בתכנה:
כאשר סיימנו לכתוב את ה Activity and Sequence Diagram היה לנו יותר קל להבין ולגשת ליצירת תיאור מימוש ומבנה התכנה. מכיוון שהבנו אילו רכיבים במערכת מבצעים ולוקחים חלק בפעילויות מסוימות ומה צורת התקשורת ביניהם, יכולנו לתכנן את המחלקות שאותם נממש באופן מדויק.
תיארנו מבנה מפורט של המחלקות וסידרנו אותו לפי סוגים כפי שלמדנו (BCE).

שאלות שהעלנו בתהליך הניתוח של המרכיב '**בדיקת מבחנים**' ליצירת Activity Diagram:

- **מתי מתבצעת בדיקת המבחן במערכת?** בעת בהגשת (submission) המבחן על ידי הסטודנט, מתבצעת בדיקה אוטומטית של המבחן.
- **היכן נשמר הציון של המבחן?** הציון נשמר בשדה grade באובייקט ה StudentTest ומשם מועבר לDB באמצעות שאילתה ממחלקת ה DBcontroller.
- **איזו מחלקה "אחראית" על ביצוע בדיקת המבחן?** המחלקה האחראית לביצוע המבחן הינה StudentTestController אשר מהווה controller עבור פונקציונאליות ה"טופס" מבחן הספציפי עבור סטודנט מסוים.
- **היכן מתבקשת המרצה לאשר את הציון?** בתפריט הראשי של המרצה מתאפשר לה לבחור באפשרות approve grades אשר פותח עבורה מסך ייעודי (מחלקה בשם AprroveGradesGUI) ובה רשימת ציונים של מבחנים שלה שהתבצעו ואפשרות לאשר אותם.
- **איך נשמר הציון בDB?** (כאשר מאושר\ לא מאושר). ברגע אישור הציון על ידי המרצה, הציון מועבר ל DB על ידי המחלקה DBcontroller.
- **כיצד המרצה מוסיפה הערות לסטודנט לגביי הציון?** בתפריט אישור הציונים ניתנת למרה גם האפשרות לשנות את הציון ולהוסיף הערה לסטודנט.

- **איך הסטודנט יוכל להיחשף לציון וההערות מהמרצה?** בתפריט הראשי של הסטודנט מתאפשר לו לבחור באפשרות show grades אשר פותח עבורו מסך ייעודי (מחלקה בשם ShowGrades) ובה רשימת ציונים של מבחנים שביצע כולל הערות מהמרצה.
- **כיצד ייוודע לסטודנט על שינוי הציון שלו?** עם אישור הציון המערכת מודיעה לסטודנט בהודעת אימייל ובמסרון SMS שהציון שלו זמין במערכת.

שאלה 2:

בהרצאה הוגדרה Reusability כתכונה של תוצר של תהליך הפיתוח אשר משקפת את היכולת לבצע reuse בהקשר לתוצר זה. בהתאם להגדרה זו, תארו בדיוק (ובהתייחסות ספציפית) ובפירוט איך באות לידי ביטוי 3 הדרישות ליישום מוצלח של Reusability בהקשר של אותם מרכיבים שלא אתם כתבתם או תכננתם ובחרתם לשלב במערכת שלכם באמצעות, Reuse תוך התייחסות בדוגמאות ספציפיות (לא 'עקרוניות' או 'כלליות') לדרישות הפונקציונליות של המערכת שתכננתם (התייחסות ספציפית בהקשר זה = התייחסות למרכיבים פונקציונליים ספציפיים קונקרטיים (לא גנריים) מתוך התיאור המילולי הראשוני של פעולת המערכת ששאתם מפתחים מהתחלת הסמסטר). לא כולל תהליך Login או זיהוי משתמש. במקרה של OCSF יש להתייחס למרכיבים הפונקציונליים הספציפיים למערכת המפותחת בפרויקט זה.
אם יש מי מ- 3 הדרישות הנ"ל אשר לא באה לידי ביטוי ב- reuse שביצעתם - הסבירו את הסיבה לכך.

בבניית האב טיפוס ומימוש הקוד ביצענו reuse בקוד קיים הנקרא OCSF . קוד זה ביצע עבורנו את החיבור בין ה- client ל- server .

3 הדרישות ליישום מוצלח של Reusability :

1. **Availability** – קוד ה- OCSF הוצג לנו לראשונה בתרגול 6 בו ביצענו חיבור בין client ל- server וכך למדנו להשתמש בקוד זה. בנוסף קוד ה-OCSF מוצג בחיפוש באינטרנט והוא נגיש לכולם , ולכן תכונת הנגישות היא תכונה חזקה העוזרת בחיבור בין client ל- server .

2. **Understandability** – קוד ה- OCSF מתועד לפרטי פרטים , כל מתודה בו וכל שורת קוד מתועדת ומובנת להפליא , שינוי פונקציונליות במתודות קיימות בו הוא שינוי הנעשה בצורה קלה ונוחה מכיוון והקוד עצמו מובן ובכל חלק הפונקציונליות ואופן השימוש מתועד ומוסבר .
דוגמא ספציפית בפרויקט שלנו היא שימוש בתכנות ההורשה בכך שבנינו מחלקות client ו server אשר יורשות מהמחלקות המובנות של OCSF- AbstractClient ו AbstractServer בהתאמה.

3. **Flexibility** – בקוד ה- OCSF קיימת מתודה הנקראת sendToAllClient מתודה זו הכרחית למימוש קוד ה-OCSF מכיוון והקוד גמיש ונוח מתודה זו הוספה כדי לממש שליחת הודעה ל clients וכדי שמתכנתים אשר משתמשים ב-OCSF יוכלו לבצע שליחת הודעות ללקוחות מסוימים בצורה נוחה ואינם יצטרכו לחקור ולגלות איך עושים זאת לבד . בפרויקט שלנו השתמשנו במתודה זו ליצירת מתודה שנקראת sendToclient() אשר שולחת הודעה ללקוח יחיד שמתודה זו הכרחית בפרויקט שלנו ונבנה בעיקרה על הפונקציה הראשית בקוד של OCSF. מתודה זו שיצרנו בפרויקט שלנו, משמשת אותנו עבור "החזרת" מענה ללקוח מהשרת באופן בלעדי במקום להעביר את המענה לכת הלקוחות.

שאלה 3:

א. הערכה כללית:

1. מהם היתרונות של מודל UMLכעזר לתהליך התכנון?
   - הסבירו איך מתקבלים (מתממשים) היתרונות שציינתם.
   - ציינו דוגמה אחת קונקרטית ממוקדת (לא כללית (גנרית) ולא)Loginמתוך תהליך הניתוח והתכן שאתם בצעתם לשימוש מועיל ב- UMLתוך תיאור והתייחסות ספציפית למרכיבים של מערכת "CEMS" שתכננתם ומידלתם.

   הערה: אין להסתפק בסופרלטיבים כלליים כמו: "מתאר", "עוזר להבנה", "מועיל", "משפר", וכו'. יש להעמיק ולהסביר את היתרונות, ולתאר בדוגמה ספציפית.
2. ציינו קשיים הנובעים מחסרונות של UMLשנתקלתם בהם.
   גם כאן התייחסו ספציפית לתהליך שבצעתם לפיתוח מערכת זו.

ב. ניתוח ודיון:

בהתאם לניסיון שרכשתם במהלך העבודה על מטלה זו, תארו אפשרויות לשינויים ושיפורים במתודולוגית UMLאשר נותנים מענה לחסרונות שנתקלתם בהם במהלך ה-design שביצעתם בפרויקט שלכם. הסבירו את תשובתכם תוך תיאור דוגמה ספציפית (לא כללית/גנרית, כולל שמות של רכיבים, לא כולל)Loginמתוך עבודתכם.

הערכה כללית:

1. **יתרונות של מודל הUML** :

I. **פישוט ויזואלי**: בעת בניית דיאגרמות ה UML קיבלנו פישוט ממשי וברור של נראות המערכת, כאשר חילקנו את המערכת למחלקות ב class diagram קיבלנו הבנה מעמיקה על היחסים בין המחלקות השונות בפרויקט ועל נראותם במערכת. כלל הדיאגרמות שבנינו יצרו לנו הבנה מפושטת על המערכת כולה וגרמו להבנה על אפיון כל מחלקה ,דרך מימושה ותפקידה במערכת.

II. **פיתוח אחיד**: כאשר פיתחנו את האב טיפוס תוך כדי הסתמכות על דיאגרמות ה-UML היה לנו קו מנחה ברור ואחיד לגבי צורת הפיתוח והמחלקות הנמצאות במערכת כך שכל חברי הצוות שאפו להיצמד לצורת התכנון שמפורטת בדיאגרמה ובכך לשמור על פיתוח אחיד ושמירה על המחשבה המעמיקה שהשקענו ביצירת הדיאגרמה והחזון שלנו איך תראה המערכת , בצורה זו קיבלנו תוצר איכותי יותר ונתקלנו בפחות תקלות.

III. **שפה משותפת**: דיאגרמות הUML היוו עבורנו שפה אוניברסלית משותפת בין חברי הצוות בזמן פיתוח המערכת ועוד לפני כאשר נתקלנו בחילוקי דעות הדיאגרמות פישטו לנו את המחשבה בנוגע לבעיות ועזרו לנו למצוא שפה משותפת בנינו כך שההסברים בין חברי הצוות היו מכווני מטרה ובשפה שכל חברי הצוות הסכימו עליה והבינו אותה.

מימוש היתרונות שציינו לעיל :

I. **פישוט ויזואלי** : יתרון זה מומש אצלנו בעיקר בדיאגרמת ה-class כאשר כתבנו ומימשנו במפורט את כלל המחלקות וההקשרים בניהם וכך ראינו בצורה פשוטה את עתיד העבודה שלנו ואת כלל המערכת .

II. **פיתוח אחיד ושפה משותפת** : כאשר פיתחנו את האב טיפוס נעזרנו בדיאגרמת ה- activity וב- flow of event שגרמו לכלל חברי הצוות להבין ולאחד את צורת המחשבה לגבי המערכת ורצף האירועים איזה מתודות בכל מחלקה ומה תפקידה של כל מחלקה.

דוגמא קונקרטית :

עבור תהליך "יצירת וצפיית דוחות סטטיסטיות" (מצד HoD) נעזרנו בדיאגרמת ה-activity כדי להבין את המחלקות השונות שעלינו לממש בחלק זה , בנוסף איזה מתודות עלינו לממש עבור תהליך זה מהשלב בו ראש המחלקה מעוניין ביצירת דו"ח או בצפייה בדו"ח קיים, בחירת נתוני הדו"ח לפי סיווג (מרצה, סטודנט או קורס) ועד לשמירתה במסד הנתונים.

2. **קשיים הנובעים מחסרונות** :

I. **חוסר בייצוג מערכות חיצוניות** – במערכת "CEMS" יש מנגנון המיובא ממערכת זרה הדואג לייבוא מידע אישי של המשתמשים בה אל מסד הנתונים , בדיאגרמות שיצרנו אין שום פירוט והתייחסות למערכות זרות במערכת ולכן הדבר מקשה על הבנת מנגנונים זרים ואופן ביצועם.

II. **חוסר במידע** – דיאגרמות הUML אינן מפרטות מידע ספציפי על הדוחות שהמערכת מפיקה בעיקר במידע הסטטיסטי המוצג בדוחות , כגון ממוצע ציונים , חציון , התפלגות ציונים. בנוסף הדיאגרמות אינן מציגות את דרך החישוב של אותם נתונים.

III. **מקרי קצה** – הדיאגרמות אינן מפרטות מקרי קצה לדוגמא המקרה בו ראש המחלקה לא נכנס לאשר הארכות זמן בזמן והארכה נעשית לא רלוונטית ויש להוריד אותה מהמערכת .

ניתוח ודיון :

רעיונות לשיפור מתודולוגית דיאגרמות ה-UML :

1. **חוסר בייצוג מערכות חיצוניות** – הצעתינו היא להרחיב את התרשימים הרלוונטיים כגון תרשים ה- class שיציג גם מידע בנוגע למערכות חיצוניות , כמו שתיארנו בסעיף הקודם שאין ייצוג למנגנון שמייבא מידע אישי על המשתמשים נציע להוסיף לדיאגרמה את המחלקות המציינות את האינטגרציה בין שני המערכות (מערכת " CEMS " , מערכת חיצונית) , את הקשר בניהן לבין שאר המחלקות בדיאגרמה כגון שימוש במידע מהמערכת החיצונית למערכת שלנו.

2. **חוסר במידע**- הצעתינו היא ליצור דיאגרמות מינוריות לכל מקרה במערכת בו יש חישוב מסוים, המציגות את אופן החישוב ומקושרות לאותה מחלקה הרלוונטית בה קורה החישוב בתוך דיאגרמת ה-class .