**שאלות פתוחות**

1.תארו את תהליך הניתוח/תכן ראשוני שביצעתם למרכיב: ביצוע בחינות (ממוחשב.) קוים מנחים לתשובה: פרטו מה הם השאלות/הפרטים שהתייחסתם אליהם בתהליך הניתוח והחשיבה, וכן התייחסו לקשרים ולמעברים .1 ממודל Use-case למודל תהליכי המיוצג בעזרת Diagram ,Activity

.2 ממודל התהליכים הדינמיים לקראת מימושו בתוכנה.

פתרון:

תהליך ניתוח מרכיב הביקור הלא מתוכנן התחלק לשלושה חלקים מרכזיים: -

חלק ה מי?

חלק ה מה?

חלק ה איך?

**בחלק המי-**

בחלק הזה ניסנו להבין מי לוקח((actor חלק במרכיב ביצוע הבחינות.

לפי מודל הUse Case ניתן לראות שהתלמיד הוא לוקח חלק ברכיב זה

-הנבחן מקבל סיסמא בעל פה מהמורה ונכנס למערכת בכדי להיכנס לבחינה

**עכשיו לחלק המה:**

נשאל כעת מה צריך לממש במרכיב ביצוע הבחינות. זה בעצם המעבר בין הUSE CASE ל Activity diagram

בשלב זה ניסינו להבין מה צריך להתרחש במרכיב ביצוע הבחינות,

בשביל לבצע בחינה:

1)תנאי מקדים – המורה העלתה בחינה לביצוע והעבירה בעל פה את הסיסמא לתלמידים הנבחנים.

2)הנבחן מקיש סיסמא ונכנס לבחינה

א)אם הסיסמא נכונה המשתמש עובר ל3

ב)אחרת הוא מקבל הודעת שגיאה וחוזר ל2

3)הנבחן מקיש תעודת זהות

א)אם ת"ז תואמת למשתמש המחובר למערכת, והנבחן מקבל גישה לסמן תשובות והמערכת מתחילה למדוד זמן

ב)אם ת"ז שגוייה המשתמש מקבל הודעת שגיאה וחוזר ל3

4)תיאור סיום הרכיב:

א) נגמר הזמן ההגשה ונסגרת האופציה להגיש את הבחינה

ב) הנבחן מגיש את הבחינה לפני תום הזמן ע"י לחיצה על "הגש".

**חלק האיך:**

שלב זה מתאר את המעבר משלב המודל תהליכים למימוש בפועל.

בשלב זה ניסנו לחשוב איך נממש את כל מה שתואר בחלק הקודם.

השתדלנו לפרק זאת לתהליכים אטומיים ככל שאפשר:

-כדי לבדוק האם הנבחן הקיש תעודת זהות נכונה-עלינו לשלוח למערכת את מספר תעודת הזהות ולבצע השוואה:

* הנבחן מקיש תעודת זהות ולוחץ "אישור"
* הקליינט ישלח בקשה לסרבר לקבל מידע של אותו משתמש המחובר
* בעזרת שאילתות מה-DB נשלוף את המידע
* הסרבר מחזיר את המידע לקליינט ומפעיל את הטיימר ומעדכן שאותו הנבחן ניגש לבחינה
* והקליינט מעודכן ונפתחת לו גישה לסמן תשובות

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 ספציפיים קונקרטיים (לא גנריים)מתוך התיאור המילולי הראשוני של פעולת המערכת ששאתם מפתחים מהתחלת הסמסטר. לא כולל תהליך Login או זיהוי משתמש.)

אם יש מי מ3- הדרישות הנ"ל אשר לא באה לידי ביטוי בreuse- שביצעתם - הסבירו את הסיבה לכך.

פתרון:

בהרצאה הוגדרו שלוש תכונות של Reusability:

-availability ,Understandability ,Flexibility

ניתן לבצע reuse בכלים שהוכחו את עצמם ויש גישה אליהם וכך בעצם פותר מהמתכנת לרשום קוד מאפס ולקבל ישר קוד פונקציונלי וגורם לייעול זמנים.

תכונות אלו באות לידי ביטוי בפרויקט שלנו כך:

Availability-

אנו משתמשים ב3 כלים מרכזיים בפרויקט CEMS שנכתבו מראש וכבר מוכחים בתעשייה שהם עובדים באופן תקין:

OCSF- מממש לנו את ארכיטקטורת לקוח -שרת

JDBC- ממשק שמנהל לנו את ההתעסקות עם הDataBase (שאילתות, שליפות מידע,ועדכון וכו')

javaFx-ספריות קיימות שבעזרתן אני בונים GUI למערכת שלנו

Understandability-

הכלים שציינו מתוחזקים ומעודכנים מפעם לפעם והקוד רשום עם תיעוד ולכן הוא קריא וקל להבנה.

וניתן למצוא הרבה מדריכים ברחבי האינטרנט של כיצד להשתמש בהם מכיוון שהם מאוד מפורסמים ופרקטים.

Flexibility-

JAVAFX ו OCSF כתובים בצורה שניתן לבצע extends למחלקות או לבצע Override לשיטות.

כלומר ניתן להשתמש בקוד הקיים ולשנות אותו במידת הצורך.

דוגמאות ספיצפיות:

- כל פעם שסוגרים את המערכת ) סוגרים קליינט(, מופעלת השיטה connectionClosed)תכונת ה(- Availability שנותנת לנו אפשרות להריץ דברים אחרונים לפני שהקליינט נסגר. לנו יש את האפשרות לכתוב אותה מחדש (תכונת ה-(Flexibilityכדי שתעשה מה שאנו צריכים. כנ"ל לגבי הפעלה וסגירה של הסרבר עם השיטות serverStarted וServerStopped. –

לאורך כל השימוש במערכת שלנו, הקליינט מדבר עם הסרבר והסרבר עונה לקליינט. לדוגמה, כאשר מורה רוצה להוסיף שאלה עלייה להזין את פרטי השאלה ותשבותיה אשר יועברו מהקליינט באמצעות השיטות handelMessageFromClientUI -> SendToServer<-Aссept.

הסרבר יטפל באמצעות connection לdatabase באמצעות שאילתות האם השאלה קיימת בdatabase המתאים ואם כן יחזיר את האובייקט המתאים אחרת יחזיר הודעת שגיאה.

בעזרת השיטה handleMessageFromClient שמספק לנו ה-OCSF ,הסרבר מזהה כי יש פנייה חדשה. (תכונת ה-Availability ) בנוסף נשים לב כי handleMessageFromClient זוהי שיטה אבסטרקטית, כלומר אין לה מימוש ואנו חייבים לממש אותה כרצוננו. (תכונת ה-Flexibility. )אז הסרבר מזהה שיש לו בקשה מקליינט, במקרה זה הוספת שאלה, הוא מפעיל את השיטה המתאימה שמוסיפה שאלה חדשה לdatabase

לאחר ההוספה הסרבר מחזיר תשובה לקליינט, במקרה שההוספה הצליחה. ההודעה מגיעה לשיטה handleMessageFromServer שנמצאת בקליינט (תכונת ה (Availability וגם אותה אנו צריכים לכתוב כרצוננו שתתאים למערכת שלנו. )תכונת ה(Flexibility

3 א. הערכה כללית:

.1 מהם היתרונות של מודל UML כעזר לתהליך התכנון?

(i)הסבירו איך מתקבלים (מתממשים) היתרונות שציינתם.

(ii) ציינו דוגמה אחת קונקרטית ממוקדת )לא כללית ולא Login )מתוך תהליך הניתוח והתכן שאתם בצעתם לשימוש מועיל בUML- תוך תיאור והתייחסות ספציפית למרכיבים של מערכת "CEMS" שתכננתם ומידלתם

פתרון:

יתרונותיו של מודל ה UML כעזר לתהליך התכנון הם:

* חלוקה גרפית של המחלקות השונות.
* מאפשר יצירה של דיאגרמות ברורות וקלות להבנה אשר ניתן להעביר ולהציג לאנשים אחרים כהסבר למערכת.
* נותן אפשרות לצפות את תהליך הפיתוח ולהתכונן בהתאם, דבר שיקצר את זמן הפיתוח.
* בעזרת ACTIVITY ו- SEQUENCE ניתן להראות רצף של אירועים, כלומר, אילו מתודות מתרחשות ובאיזה סדר.
* מאפשר למדל תוך כדי שימוש בחלק מעקרונות הOOP-, מה שמבטיח תכנון ועבודה נכונה.

היתרונות שציינו באים לידי ביטוי בכך- בעזרת ה UML של הפרויקט שלנו ניתן להבין ביתר קלות מה הפונקציונליות של המערכת שלנו, אילו מחלקות יהיו במערכת, ואיך נממש את הפונקציונליות של המערכת. בנוסף בעזרת התרשימים של ה-Activity וה-Sequence שעשינו, יש לנו מושג איך לממש את הפונקציונליות בקוד עצמו.

דוגמא לשימוש יעיל במערכת שלנו של הUML במערכת שלנו הוא בניתוח בניית בחינה. בעזרת הUML אנו יודעים אילו ישויית לוקחות חלק בתהליך זה, ובACTIVITY תארנו את הפעולות שמתבצעות במערכת ותיזמונם כתוצאה מפעלות של המשתמש ביציאת בחינה :בחירת קורס, בחירת שאלות, מתן ניקוד לשאלות, בדיקת תקינות, ושליחת נתונים לdatabase

Sequence תארנו איך אנו מבצעים את הפעולות שהגדרנו ששיכות לבניית בחינה.

כעת כשנרצה לרשום את הקוד בפעול שיממש את הפונקציונליות של בניית בחינה יש לנו תמונה מסודרת בעזרת דיאגרמות הUML ויש הבנה יותר ברורה מה בדיוק צריך לכתוב.

.2 ציינו קשיים הנובעים מחסרונות של UML שנתקלתם בהם. גם כאן התייחסו ספציפית

לתהליך שבצעתם לפיתוח מערכת זו.

פתרון:

אחד הקשיים במודל ה-UML הוא שאין אפשרות למדל את ה-GUI, ולכן עדיין אין לנו מידע מספיק לגבי איך המערכת תוצג למשתמשים, למשל המסך של בניית מבחן.

הפתרון שאנו השתמשנו בו הוא הכלי Scene builder בו הרכבנו את החלונות שלנו למשתמשים למטלה של האב טיפוס (בממשק אדם-מחשב), ובכך הבנו יותר טוב מה עלינו לממש.

ובנוסף דיאגרמות הUML דורשות הרבה מאוד זמן, והבנה של איך הדיאגרמה צריכה להיות שבזמן הזה היה ניתן לרשום את הקוד עצמו.

ב. 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, לא כולל (Login מתוך עבודתכם.

אנחנו משתמשים במתודולוגית UML 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.

שינוי אפשרי במתודולוגיית UML היא לקחת את הUML בגישה יותר בסיסית וכקו מנחה. כלומר לא להשקיע כל כך הרבה זמן ביצירת ה UML ובפירוט רב, אלא לייצר את אותם תרשימים אך בצורה בסיסית יותר תוך כדי הסבר מילולי. זה יביא לכך שנבזבז פחות זמן ביצירת ה UML ונוכל להשקיע את הזמן הזה בכתיבת קוד בפועל.