G9_Answers.Ass2

קבוצה 9	תאריך הגשה 6.12.20	
שם	ת.ז	אימייל
חיים אזולאי	203374855	haimazu4@gmail.com
הודיה מקונן	204632988	hodayamekonen@gmail.com
אנסטסיה קוקין	321241192	nastyak6@gmail.com
בר כץ	315818567	barkatz0610@gmail.com
רינת סטודנץ	316237577	rinat20968@gmail.com
רועי עמר	203118666	roi.amar@e.braude.ac.il

שאלה 1:

1. תארו את תהליך הניתוח שביצעתם למרכיב: ניהול ביקור לא מתוכנן (מזדמן).
פרטו מה הם השאלות/הפרטים שהתייחסתם אליהם בתהליך הניתוח והחשיבה.
בתשובה התייחסו לקשרים ולמעברים 1. ממודל Use-case למודל תהליכי המיוצג בעזרת Activity Diagram, 2. ממודל התהליכים לקראת מימושו בתוכנה.

תשובה:

התהליך שביצענו עבור המרכיב : "ניהול ביקור לא מתוכנן (מזדמן)" הוא תהליך שבו למטייל מזדמן אין אינטראקציה ישירה עם המערכת שלנו ולכן,הגענו למסקנה כי עלינו לבצע תהליך "אישור כניסה לפארק" ע"י עובד בפארק כאשר מטייל מגיע לפארק ומבקש להיכנס ללא הזמנה מראש.

השאלות והפרטים שהתייחסנו אליהם:

- מי יפעיל את המערכת בסיטואציה זו.
- אילו פרטים ילקחו מהמטייל המזדמן וירשמו במערכת.
- מה המערכת צריכה לבצע כאשר מגיע מטייל מזדמן (לדוגמא בדיקת זמינות בפארק).

במודל הuseCase התייחסנו למטייל מזדמן כ- preOrederTraveler כאשר הוא מגיע ומבצע הזמנה חדשה.
הוא אינו שונה משאר השחקנים במודל, אך כאשר ביצענו מעבר למידול הזמנת המטייל המזדמן במודל ה- activity, הראינו בדרך יותר מפורטת כיצד התהליך מתבצע ע"י שחקן אחר שהינו עובד בפארק שמבצע עבור המטייל המזדמן את ההזמנה במערכת.
ביצוע ההזמנה יהיה יותר קצר מתהליך הזמנת ביקור מראש וידרוש מהמערכת לבצע בדיקת זמינות תפוסה בפארק ולקיחת פרטים (המערכת תבצע פחות פעולות).
במודל זה נמדל את המטייל המזדמן אחרת מכיוון שנתיב ההזמנה שונה משאר המטיילים שמבצעים הזמנה מראש. ביצוע המידול ב- activity של סוג מטייל זה יבוצע ב- partition שונה ויבצע אינטראקציה רק מול ה- partition של עובד בפארק ודרכו יתחיל המסלול של ביצוע ההזמנה למטייל מזדמן.

לקראת מימוש התוכנה נבנה מסך עבור העובד בפארק שאחד מתפקידיו יהיו לבצע הזמנות עבור מטיילים מזדמנים ותהייה אפשרות לבצע בדיקה מהירה לזמינות בפארק ובכך ליידע את המטייל המזדמן אם יכול להיכנס לפארק או לא.

שאלה 2:

2. בהרצאה הוגדרה ***Reusability*** כתכונה של תוצר של תהליך הפיתוח אשר משקפת את היכולת לבצע reuse בהקשר לתוצר זה. בהתאם להגדרה זו, תארו יישום של 3 התכונות המאפשרות לכם לשלב במערכת GoNature שאתם מפתחים קטעי קוד ומרכיבים אחרים שלא אתם כתבתם או תכננתם.
תארו בדיוק (ובהתייחסות ספציפית) ובפירוט את התכונות המאפשרות Reuse של אותם מרכיבים אשר בחרתם לשלב במערכת שלכם, תוך התייחסות בדוגמאות ספציפיות (לא 'עקרוניות' או 'כלליות') לדרישות הפונקציונליות של המערכת שתכננתם (התייחסות ספציפית בהקשר זה = התייחסות למרכיבים ספציפיים מתוך התיאור המילולי הראשוני של פעולת המערכת ששאתם מפתחים מהתחלת הסמסטר. לא כולל תהליך זיהוי משתמש).
אם יש מי מ-3 התכונות הנ"ל אשר לא באה לידי ביטוי ב-reuse שביצעתם - הסבירו את הסיבה לכך.

תשובה:

התכונות שהוגדרו בהרצאה עבור REUSABILITY הן: זמינות, גמישות והבנה.
נתאר כיצד שלושת התכונות הללו מיושמות במערכת שלנו ומאפשרות שילוב של קוד מוכן:
מערכת GO-NATURE אמורה לפעול בארכיטקטורת שרת - לקוח על מנת לאפשר ממשק משתמש נוח ושמירת נתונים בצד השרת. מכיוון שמדובר בארכיטקטורה נפוצה וגנרית ניתן להשתמש ב- framework קיים ולבצע REUSE במקום לפתח קוד מחדש.
לשם כך אנו זקוקים למחלקות אשר מממשות ארכיטקטורת שרת - לקוח ואת התקשורת ביניהן. בחלק זה במערכת, בחרנו להשתמש בקוד מוכן העונה לנו על הצרכים. כלומר, החלטנו כי אנו מבצעים REUSE בארכיטקטורת OCSF שנחשפנו אליה במהלך הקורס. בחרנו לעשות זאת מכיוון שקוד זה היה זמין, מתועד היטב, הוסבר באופן מפורט והומלץ ע"י צוות הקורס. כמו כן, הקוד מכיל את כל הפונקציונאליות המרכזית של ארכיטקטורת שרת - לקוח ולכן השתמשנו בו.

נתאר כיצד ארכיטקטורת OCSF באה לידי ביטוי אל מול הדרישות הפונקציאונליות:

- **גמישות:**
ארכיטקטורת OCSF מאפשרת גמישות בכך שניתן לרשת ממנה תכונות מרכזיות ולהתאימן למערכת שלנו. למשל, יש לנו את המחלקות chatClient ו- EchoServer שיורשות ועושות שימוש ממחלקות OCSF שאנו משתמשים בהן בתהליך ביצוע ההזמנה לשם העברת המידע.
ה- REUSE בא לידי ביטוי בכך שהפונקציות handlemessagefromserver , handlemessagefromclient ניתנות לשינוי בהתאם להודעות שנרצה להעביר.
השינוי שנבצע יהיה בפונקציות אלו ופונקציות נוספות שנוסיף בהתאם למערכת שלנו, ובכך ניתן להגיד שהמחלקות ב- OCSF הן אדפטיביות.
בשימוש בארכיטקטורה זו, אם בעתיד נרצה לשדרג את המערכת- לא נדרש לכתוב את כל המודול שאחראי על שרת-לקוח מחדש, אלא רק להתאים את ה- OCSF מחדש. בהתייחסות ספציפית לדרישות מהסיפור כחלק מהגדרת הפרויקט:
  - "בגרסה השנייה תפותח אפשרות גישה למערכת מכל מקום באמצעות רשת האינטרנט (WWW) והרשת הסלולרית".
  - " עליכם לתכנן ולפתח את המערכת כך שתהליך המעבר העתידי לשלב השני (גרסה שניה) יהיה יעיל וחלק במידה מירבית".

- **<u>הבנה:</u>**
הבנת ארכיטקטורת OCSF מאפשרת לנו ככותבי המערכת להשתמש באופן ברור ותקין בארכיטקטורת שרת – לקוח. בעזרתה אנו יכולים להבין כיצד עלינו לבצע את התיאום בין שני הצדדים ובכך לממש על המערכת שלנו.

- **<u>זמינות:</u>**
ארכיטקטורת OCSF הינה קוד פתוח (חינמית), היא זמינה להתאמה בכל מחשב וכל פרויקט אשר נבצע.

2 התכונות האחרונות לא נגזרות מתוך התיאור המילולי הראשוני של פעולת המערכת.
תכונות אלה הן חלק מתהליך ה-REUSE ובאו לידי ביטוי בשימוש כאשר ביצענו את תהליך ה- REUSE עצמו. כלומר, טרם הטמעת ארכיטקטורת OSCF קראנו את הקוד ואת התיעוד ובחרנו כיצד לשלבו במערכת. כמו כן, במהלך הלמידה בקורס הוסבר לנו שקוד זה זמין עבורנו לטובת מימוש המערכת.

<u>שאלה 3-א':</u>

3. א. <u>הערכה כללית</u>:
   1. מהם היתרונות של מודל UML כעזר לתהליך התכנון?
      (i) הסבירו איך מתקבלים (מתממשים) היתרונות שציינתם.
      (ii) ציינו דוגמה אחת <u>קונקרטית</u> ממוקדת (לא כללית ולא Login) <u>מתוך תהליך הניתוח והתכן שאתם בצעתם</u> לשימוש מועיל ב-UML תוך תיאור <u>והתייחסות ספציפית</u> למרכיבים של מערכת "GoNature" שתכננתם ומידלתם.
   2. ציינו קשיים הנובעים מחסרונות של UML שנתקלתם בהם. גם כאן התייחסו <u>ספציפית</u> לתהליך שבצעתם לפיתוח מערכת זו.

<u>תשובה:</u>

1) <u>נמנה את היתרונות של ה- UML:</u>
   - דיאגרמות פשוטות להבנה
   - שפה אחידה – סטנדרט יחיד ליצירת אפיוני מערכת
   - מעבר קל מדיאגרמה למימוש
   - התאמה לדרך החשיבה של המשתמש

i. היתרונות מתקבלים מכך שה- UML מאפשר הסתכלות מזוויות ראייה שונות על המערכת שנרצה לפתח, מאפשרת לראות את התמונה הכוללת של המערכת וממחישה את מבנה המערכת והקשרים המשותפים. בשימוש ב- UML ניתן לראות ויזואלית את רכיבי המערכת ושיטה זו קלה יותר להבנה מתיעוד במסמכים. בנוסף במידה ואנו עובדים בצוות גדול או כמה צוותים בעזרת שימוש ב- UML אנו מוודאים כי כולם מדברים "באותה השפה", ויש פחות מקום וסיכוי לפרשנות אישית שגויה.

ii. ה- UML הועיל לנו להבנת התהליך של ביצוע ההזמנה. התבקשנו להבדיל בין סוגי המבקרים, ובעזרת שימוש ב-UML פירטנו את השלבים השונים.
נפרט דוגמאות בהתאם:
   - דיאגרמת המחלקות איפשרה לנו באמצעות תכונת ההורשה להבדיל בין סוגי המבקרים השונים, להבחין בין ההרשאות שלהם, התכונות השונות ואילו פעולות הם יכולים לבצע במערכת.
   - בדיאגרמת ה- sequence הצגנו את הפונקציות הספציפיות במערכת, תיארנו את הזרימה בין המחלקות והמתודות השונות כולל המידע שיוחזר מרגע כניסת משתמש (המבקר) לחלון המתאים במערכת ועד להחזרת המידע דרך מעבר בשרת ב- DB וחזרה למסך בצד הלקוח. תהליך זה מתאר את כל השלבים לביצוע ההזמנה וכך אנו יכולים ללמוד כיצד יראה הקוד בפירוט.

2) <u>קשיים שנבעו מחסרונות ה-UML שנתקלנו בהם :</u>

- <u>ריבוי דיאגרמות</u> :
  - ❖ נדרשנו לבצע את דיאגרמת ה- package, זוהי דיאגרמה שמתבססת על דיאגרמת ה-class כך ששילובן היה אמור להיות טבעי ופשוט, אך לא היה קל לשלב ולהבין את החלוקה לקטגוריות.
  - ❖ ריבוי דיאגרמות גורם לריבוי דעות ואם מידולן מתבצע ע"י אנשים שונים ייתכן ויהיו התנגשויות בין המידולים השונים.
  - ❖ ריבוי דיאגרמות עלול לגרום לכך שרוב ההתעסקות בפרויקט יהיה תכנון הפרויקט ולא ביצוע הפרויקט בפועל (לא תואם תצורת פיתוח אג'ילית).
  - ❖ ריבוי דיאגרמות עשוי לגרום לכך שכאשר המתכנת ייגש לממש את המערכת, הוא לא יבין מאיפה להתחיל (מרוב דובים לא רואים את היער).
- <u>התעסקות רבה בעיצוב</u> :
  עיצוב הדיאגרמות והבאתן למצב אסתטי וקריא לקח זמן ניכר שהיה יכול להיות מושקע בקוד יעיל ונקי יותר.

<u>שאלה 3-ב':</u>

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

<u>תשובה:</u>

<u>ריבוי דיאגרמות</u> :
בהתאם לנסיון שרכשנו במהלך העבודה על הפרויקט, ראינו שאנו היינו רוצים לשפר את תהליך תיאור ההתרחשויות בדיאגרמה מרוכזת יותר. למשל מידלנו עבור 4 דיאגרמות את תהליך הזמנת הביקור. במהלך המידול ראינו שהייתה חזרתיות בין הדיאגרמות ולכן היינו מציעים לשלב בין דיאגרמת ה- activity ל- sequence לכדי דיאגרמה אחת ובכך ליצור קשר מובן וברור יותר.
בנוסף, לא ניתן לשחזר את הסיפור המלא מדיאגרמה אחת ותמיד נצטרך להוסיף מידע מדיאגרמות נוספות.