שיטות הנדסיות לפיתוח מע' תכנה, תרגיל 2:

קבוצה: 13

חברי הקבוצה: נתיב לנגל 315680058 nativlengel11@gmail.com

נדב מאיר 305781452 nadav491@gmail.com

תומר אבירם 206066052 tomerav83@gmail.com

שאלה 1:

עבור תהליך ביצוע הבחינות באופן ממוחשב שמערכת AES שלנו, התייחסנו לשלושה היבטים עיקריים והם –

1. ההיבט הארכיטקטוני Architectural Patterns

בחרנו לבנות את שלד התכנה, בעזרת מודל שלוש השכבות, ובכך בעצם לאפיין את הבסיס לארכיטקטורת המערכת, שתאופיין על ידי שלושה מרכיבים – ממשקים גרפיים, ממשקי בקרה, וממשקי מידע.

את המערכת כולה ניתן לחלק באופן אבסטרקטי לשלוש האפיונים האלו, ובפרט את התהליכים הפנימיים כדוגמת תהליך ביצוע הבחינה באופן ממוחשב.

את תהליך זה בחרנו להציג במערכת על ידי כל שלושת השכבות, כאשר כל אחת מהן באה לידי ביטוי באופן הבא –

הממשקים הגרפים – משמשים את כל השחקנים הרלוונטיים במערכת, (מורה, תלמיד..), כאשר ביצוע הבחינה עצמה מתנהל באמצעות ממש גרפי-ויזואלי, שאותו ממלא התלמיד.

ממשקי הבקרה – הם אלו שינהלו את כל העברת המידע בין המחלקות השונות, ויהוו מעין תיווך בין שכבת הוויזואליות לבין שכבת המידע, כדוגמה – העברת טופס המבחן לאחר מילוי באמצעות הממשק הגרפי, למערכת לשם בדיקה ומתן ציון, אפשרות להעברה למורה למתן אישור סופי, ובחזרה לתלמיד במקרה הצורך, כל ההתנהלות בין המחלקות והישויות השנונות עם המערכת, מתבצעת באמצעות הבקרים האלו.

ממשקי המידע – ככל מסד הנתונים בו יישמר כל המידע הנחוץ, ובפרט מחלקות מסוימות אשר יכללו מידע רלוונטי, לדוגמה – טפסי הבחינה לאחר מילוי של התלמיד ולפני, מאגרי השאלות והמבחנים, והדוחות הסטטיסטים לאחר קבלת הציונים.

1. דפוסי העיצוב - Design Patterns

בכדי לתאר את המערכת כולה ככלל, ואת תהליך ביצוע הבחינה בפרט, בנינו דיאגרמות שונות, כדוגמת דיאגרמת המחלקות, ובתוכה מעיין מבט – packaging-view, ודיאגרמת הפעולות (activity diagram), כאשר כל אחת מהן מציגה באופן שונה את אותם ההיבטים, הבחירות, הדילמות והפתרונות, שבאמצעותם נממש את התהליך עצמו.

1. עיצוב – Design ו שימוש חוזר Reuse

העיצוב הבולט הוא עיצוב השרת-לקוח, client-server, כאשר כל לקוח מתחבר לשרת המערכת שהוא בעצמו מתחבר למסד הנתונים, לכל לקוח אפיונים משלו, לדוגמה מספרו, והרשאות הגישה שלו במערכת ושרת המשותפים לכולם.

השימוש החוזר בא לידי ביטוי, בבעיית בניית תכנת שרת-לקוח, שהיא נפתרה על ידי שימוש בפתרון קיים – ocsf.

באופן כללי, בהמון תהליכים, ובפרט עבור תהליך ביצוע הבחינה באופן אוטומטי, דיי היה ברור לכולנו איך ייראה התהליך, במובן הרחב והכולל, ואפילו במובן הצר של ירידה לפרטי תכנות, כך שלא נתקלנו בדילמות, או קשיים שהצטרכנו למצוא להם פתרונות חלופיים.

השיקולים לבחירת הפתרון, והדרך בה בחרנו לייצג אותו, ולממש אותו, הם פשטות, יעילות, ותקינות, כלומר – הדרך הכי פחות מסובכת, הן מבחינת מימוש והן מבחינת איכות המימוש(יעילות) כך שהתוצאה תהיה תקינה.

שאלה 2

1. ראשית התכונה הבולטת ביותר היא ביסוס – שרת לקוח, ושרת –מסד נתונים.

לא מימשנו בעצמנו את התכונות האלו, אלא השתמשנו בתכונות קיימות כדוגמת ocsf ו mysql connection 5.1.

תכונה נוספת באמצעותה נוכל לבטא את עיקרון ה reuse היא הדוחות הסטטיסטים – נוכל להשתמש באלגוריתמים קיימים לשם חישוב הדוחות והנתונים, מבחינת ההיבט המתמטי.

1. העיקרון הראשון לדוגמא, של תכונות שרת-לקוח, ושרת- מסד נתונים, היא תכונה המצריכה זמן רב, מאמצים רבים, ולמידה רבה, בהמון נושאים המתקשרים לתכונה זו, שלא נחשפנו אליהם כלל, ועד כה לא נתקלנו בהם במהלך התואר, והקורסים השונים שעברנו עד כה.

עבור תכונה זו, עקרון ה reuse הוא אפילו קריטי, שהרי בנייה של כל התכונות האלו היא בעצמה מערכת שלמה, המתחלקת לתתי מערכות, שאילוצי הזמן כלל לא מאפשרים לממש באופן עצמי וללא שימוש חוזר.

שאלה 3:

א.

1. מודל ה uml אפשר לנו בעצם לתאר בצורה הטובה ביותר את בניית המערכת שלנו, המבוססת על תכנות מונחה עצמים.

הדיאגרמות השונות, הן כלי עזר המאפשר לבטא תכונות ואפיונים בדרכים שונות, המתאימות יותר לדרך החשיבה של המשתמש, ובצורה ידידותית כאשר תמיד נוכל למצוא דיאגרמה שבאמצעותה נוכל לבטא בפירוט ויותר לעומק, חלקים שונים במערכת, ולשפר את ההבנה ואת התנהלות התהליכים הפנימיים במערכת הכוללת, כדוגמת activity diagram, ובנוסף מעיין מבטים שונים על המערכת למשל, packaging view , 3-layers class diagram.

1. למשל, אם נסתכל רק על דיאגרמת המחלקות במודל שלוש השכבות, לא כל כך נוכל להבין איך מתבצע תהליך ביצוע הבחינות באופן ממוחשב(אוטומטי), אך באמצעות ה activity diagram יכולנו לתאר זאת בצורה קצת יותר פרטנית ומובנת.
2. בסעיף א' הצגנו יתרונות של ריבוי הכלים, העיצובים והדיאגרמות במודל זה, אך הדבר הוא גם חסרון - קשה לבטא את הקשר בין כל הדיאגרמות השונות, שהרי לכל אחת מהן מבט ייחודי משלה על התוכנה אותה היא מייצגת, ולעיתים הריבוי הזה עלול לסרבל הבנה מבחינת ארכיטקטורת התכנה בשלבי התכן והעיצוב שלה, במיוחד במערכות מורכבות.

ב. נבחר את נושא תיאור התהליכים והפונקציות של המערכת, בדגש על ריבוי הדיאגרמות והמבטים השונים אותם נאלצנו לבחון.

וכדוגמה ספציפית נבחר למשל את תהליך הנפקת הדוחות ועדכונן, זהו מעיין תהליך רקע המתעדכן בכל פעם שמבחן קיבל ציון שאושר על ידי המורה, לאחר בדיקה אוטומטית של המערכת, אך את הנקודה הזו היה נורא קשה לבט בכל הדיאגרמות השונות שבאמצעותן עצבנו וביטנו את המערכת.

תהליכי מערכת כפרט הם משהו שדיי קשה לבטא באמצעות מודל זה, בעיקר במערכות מורכבות וגדולות, אך גם במערת aes שלנו נתקלנו בכמה, והפתרון שאנו מציעים הוא מעיין איחוד של דיאגרמות לכדי דיאגרמה אחת שאולי מבטאת קצת פחות לעומק איך עובדת המערכת, אך מקנה הבנה מעמיקה יותר מבחינת התהליכים הפנימיים הנקשרים והתלויים זה בזה, ובנוסף הוספת קשרים בין האובייקטים בדיאגרמות, שיבהירו בדיוק את המטרה הזו, כלומר – מעיין קשרים ואילוצים "background updated" או "depends on" או "awaits for", בכדי להבהיר בצורה הטובה ביותר, ועד כמה שאפשר, את מימושם, ומטרתם של תהליכי המערכת הפנימיים.