**סקר איכות התיכון של VirtuClass**

ביצענו סקר עם ד"ר ראובן יגל, ודנו במספר נושאים:

1. איכות התיכון.
2. עמידות התיכון לשינויים.
3. פתרונות לבעיות שעלו

במהלך הסקר עלתה הנקודה שצריך לפרק מחלקות גדולות לתתי מחלקות (SRP), שלכל מחלקה יהיה תפקיד משלה. כך ימומשו עקרונות SOLID שלכל מחלקה אחריות אחת, ושלכל מחלקה תהיה רק סיבה ***אחת*** לשינוי. כרגע הקוד עמיד לשינויים, אך אם נמשיך לכתוב כמו שכתבנו את מחלקת ה LoginMediator , הקוד יתנפח, ימלא הרבה תפקידים שונים במחלקות בודדות, יהיה לא קריא (בעיקר אם נרצה להשתמש בו בעתיד), יהיה קשה לשנותו ויהיו הרבה סיבות לשנותו ולבסוף גם יהיה קשה לבצע עליו טסטים.

כמו כן, עלתה הנקודה שיש להוריד את המורכבות הציקלומטית ע"י הורדה של if מקוננים, ושינויים ל switch case. בתוך ה switch case נכניס פונקציות שונות כך שיהיו קצרות (עקרון SOFA).

**מה אנו מתכוונים לעשות:**

קודם כל, נפתור את בעיית המחלקות הארוכות ע"י חלוקה למחלקות קטנות (בעיקר בעיה של ה (LoginMediator. לאחר מכן נפתור את בעיית המורכבות הציקלומטית ע"י חלוקה לשיטות רבות וקצרות.

נקפיד על שמות משתנים טובים וקריאים, על תיעוד, ועל טסטים שוטפים *ועצמאיים*. כל שיטה צריכה להכיל קצת ארגומנטים, ואסור לקבל כפרמטרים flags. ע"י כך נמנע ריחות רעים בקוד.

**תיכון שכבר ביצענו**

הפרדנו את האפליקציה ל 3 שכבות ברמת ה client ו 2 שכבות ברמת ה server, כאשר ב client ה 3 שכבות הן: Gui,Logic,Connection ואצל ה server: Logic,Connection.

הפרדה זו עוזרת בהפיכת הקוד לקריא, ומחזקת הפוקוס של כל גורם וגורם. דבר זה מביא ללכידות (Cohesion) גבוהה, שתעלה את הסיכוי שהקוד יהיה שמיש בעתיד. כמו כן לכידות משאירה את המורכבות נמוכה.

דבר שני שביצענו הוא יישום סינגלטונים לFakeDB ול Synchronizer (אחראי לקשר הראשוני של הלקוח-שרת). זה מקטין את הצימוד, מייפה ומפשיט את הקוד.

כמו כן הורדנו תוכנה ייעודית שמטרתה לגלות ריחות רעים בקוד ולשפר את איכותו, להלן התוצאות כרגע (רובן חיוביות):

<https://github.com/ilayze/VirtuClass/tree/master/SDF/Notes/bad%20smells%20-%20code%20review%20images>