C17 Ex02 JacobSobolev

תיאור עבודת הגשה:

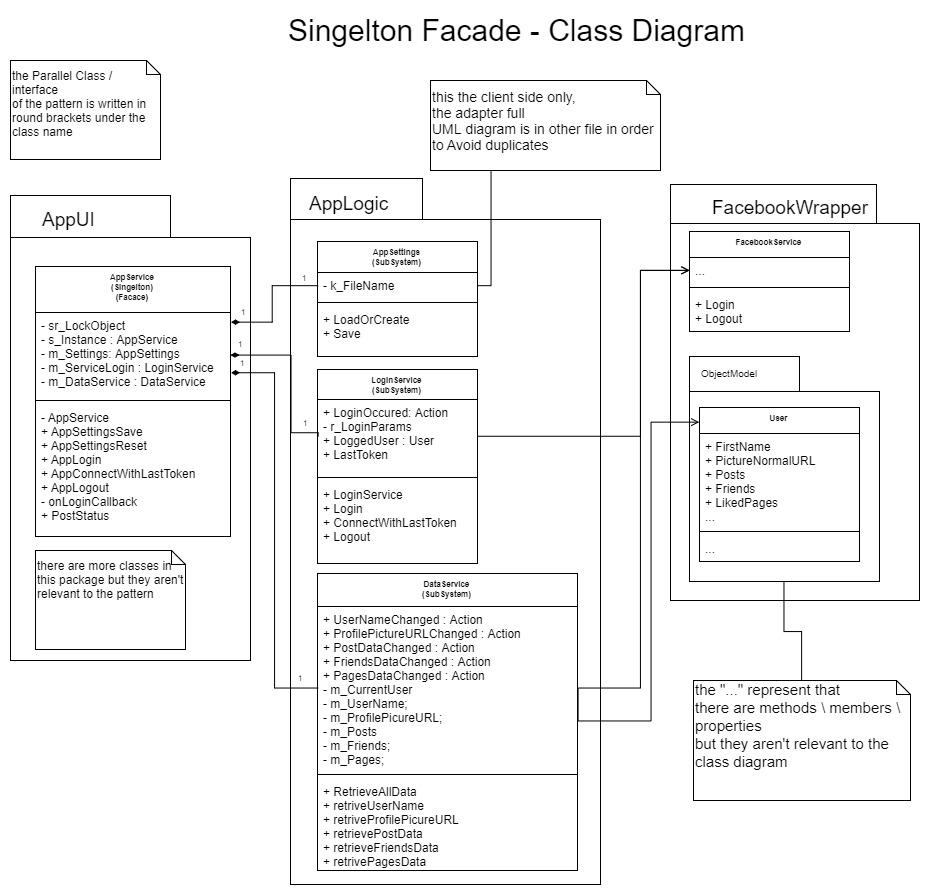
: א. רישמו הסבר קצר לגבי המימוש )איזו תבנית בחרתם, מדוע בחרתם בה / מה מטרת השימוש בה בתוכנית שלכם וכיצד היא מומשה

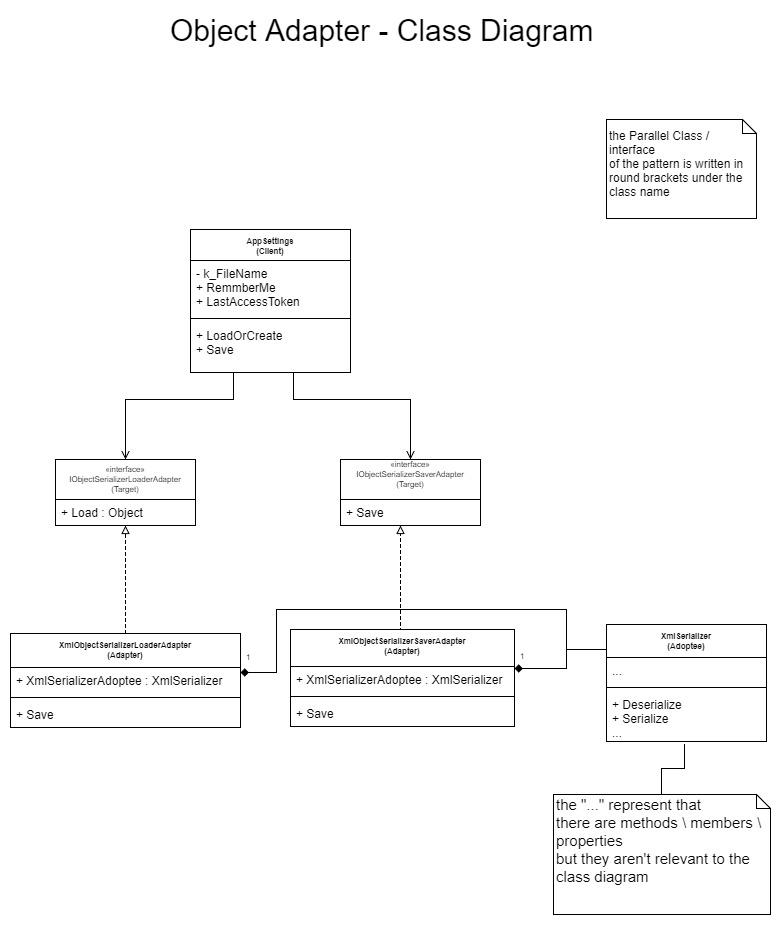
1. התבניות שבחרנו הם: Singleton Façade Adapter
   1. Singelton – עקב העובדה שחילקנו את את התוכנית לתתי רכיבי UI אשר דורשים גישה לאותה מחלקה יחידה אשר כוללת מידע יחיד ועקבי לאורך כל משך חיי התוכנית ופונקציונאליות אז ראינו לנכון לממש Singelton.  
      מטרת השימוש של המחלקה בתוכנית היא לאפשר גישה למידע יחיד ועקבי ופונקציונאליות לכלל רכיבי הUI במערכת.  
      הsingleton מומש ע"י double-check locking singleton
   2. facade – רצינו ליצור אפליקציה אשר מפרידה את שכבת הUI ושכבת הלוגיקה והשכבה הלוגיקת מורכבת מתת שכבות לוגיות אז ראינו לכנון לממש Facade (בסנגון מודל ה MVVM).  
      מטרת השימוש היא לאפשר גישה לכלל ריכיבי הUI מול שכבה יחידה אחת אשר כוללת את תת השכבות הלוגיות הנדרשות לפעולה של האפליקציה.  
      אופן המימוש: ארזנו את כלל רכיבי הלוגיקה בDLL נפרד ויצרנו שכבת ביננים אשר נגישים לכלל רכיבים הUI.
   3. adapter – בחרנו להשתמש בadapter בגלל שהיה צורך לשמור ולטעון אובייקטים מורכבים לקבצים ורצינו לאפשר לעבוד עם מחלקה פולימורפית עם שם של קובץ ואובייקט בלבד בלי קשר למימוש איך מתבצעת השמירה והטעינה.  
      מטרת השימוש בתוכנית זה לשמור ולטעון אובייקט מורכב לקבוץ ומקובץ בכדי לשמור מידע אשר נוצר עקב פעילות המשתמש באפליקציה.  
      המימוש התבצע ע"י Object Adapter (Gang-of-4)

( ב. רישמו איזו מחלקה בקוד שלכם מייצגת איזה "שחקן" בתבנית

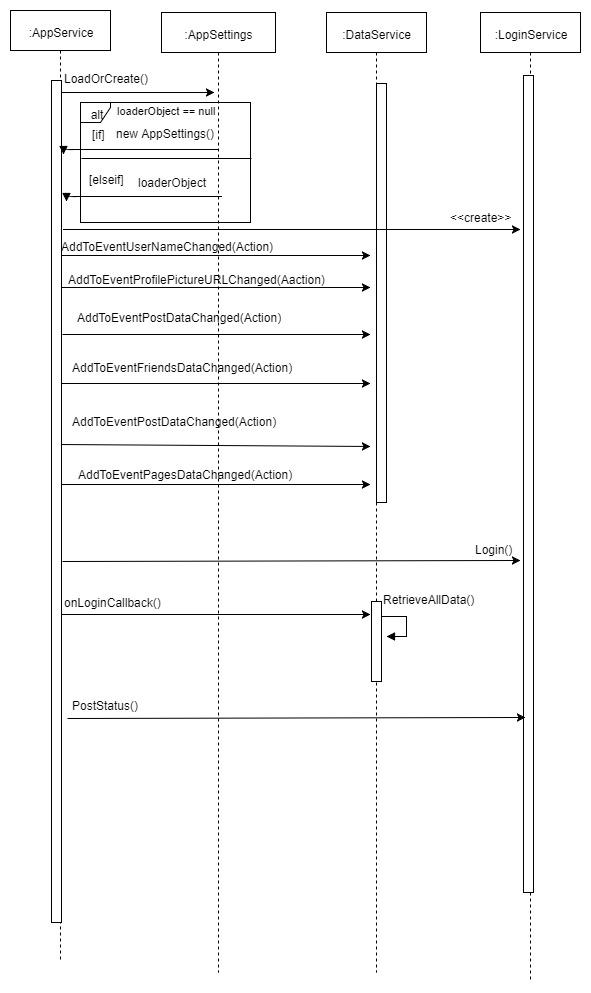
1. Singleton -
   1. המחלקה AppService מייצגת Singleton
2. Façade -
   1. LoginService - מייצגת את ה SubSystem
   2. AppSettings - מייצגת את ה SubSystem
   3. DataService - מייצגת את ה SubSystem
   4. AppService – מייצגת את ה Façade
   5. כלל רכיבי ה UserControl מייצגים את הClient אך חשוב לשים לב שהרכיב גם Singleton ואין פה קומפיזציה אלא הרכיב ניגש אליו.
3. Adapter-
   1. AppSettings – מייצגת את הClient
   2. IobjectSerializerSaver – מייצגת את ה Target
   3. IobjectSerializerLoader – מייצגת את ה Target
   4. XmlObjectSerializerSaver – מייצגת את Adapter
   5. XmlObjectSerializerLoader – מייצגת את הAdapter
   6. XmlSerializer – מייצגת את ה Adoptee

ג.

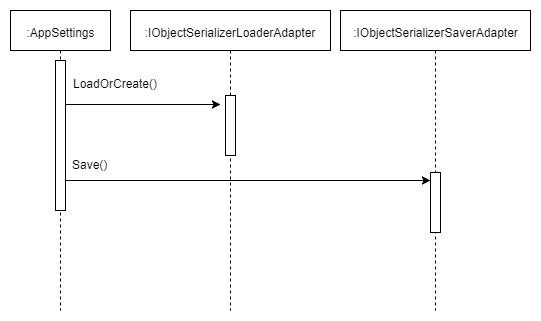




Singelton Façade – Sequence Diagram:



Object Adapter – Sequence Diagram:



עבודה אסינכרונית – בכלל רכיבי הUI באפליקציה נעשה שימוש בעבודה אסינכרונית.  
טעינת המידה לתוך רכיבי הUI מתבצעת ע"י רישום שם כל UserControl לשינוי במידה, ברגע שמתבצע השינוי ברכיב המידע מופעל האירוע ומתבצע העדכון בThread חדש.  
פעולות מחיקת פוסט והוספת פסוט גם כן מתבצעות כן, רכיב הUserControl נרשם לאירוע סיום הפעולה אשר בסופה הוא מראה הודעה למשתמש, הפעולה מופעלת ע"י Thread חדש ע"י אינטרקציה של המשתמש לUI. כל זאת נעשה בכדי למנוע משכבת הUI לקפוא ולתת חייות משתמש יותר טובה בנוסף לכך שימוש בThread ים לאכלוס רכיבי הUI מזרזת את האכלוס הכלל של כלל הרכיבים.

1. UserControlMain -
   1. עדכון שם המשתמש – שם המתודה updateUserName
   2. עדכון תמונת הפרופיל – שם המתודה updateProfilePictureThread
2. UserControlBasicActions -
   1. עדכון רשימת הפוסטים – שם המתודה updatePostList
   2. עדכון רשמית החברים – שם המתודה updateFriendsList
   3. ביצוע פוסט – שם המתודה makePost
3. UserControlPostsActions
   1. עדכון הdata source של הפוסטים – שם המתודה updatePostListDataSource
   2. מחיקת פוסט – שם המתודה deletePosts
4. UserControlPagesTable
   1. עדכון רשימת הדפים המשתמש עשה לייק – updateListViewPages
   2. ביצוע פילטר על רשימת הדפים - filterDataViewBasedOnInputText

Data Binding – בעבודה בוצע שימוש ב-Data Binding ב-UserControlPostAction.