### תיאור קצר של הפיצ'רים שבחרנו לממש בתרגיל הקודם:

* Sort pages by Likes – ממיין את אוסף העמודים שהמשתמש אהב לפי כמות לייקים.
* Filter posts by date – מפלטר את אוסף הפוסטים שעל הקיר של המשתמש לפי טווח תאריכים שבחר.

### שימוש ב- MultiThreading:

בזמן בניית אובייקט UserFacade, יש שימוש ב- MultiThreading בזמן קריאה למתודות שמושכות מידע מאובייקט User של פייסבוק. כל שימוש במתודה FBLogic. CreateDataUnitCollectionFromOtherwiseIncompleteData מערב פניה לאובייקט מטיפוס User שמשכנו מפייסבוק ושמירת מידע זה בתוך אובייקט מטיפוס DataUnit שמהווה חלק מהתכונות של UserFacade. כל קראיה שכזו מהווה חלק משמעותי מהעומס שחל על האפליקציה לאחר התחברות. קריאה למתודה הזו ולדומות לה בעזרת Threads שונים הורידו בצורה משמעותית את זמני ההמתנה בזמן התחברות לאפליקציה, ותרמו לחוויית משתמש חלקה יותר.

### שימוש ב- DataBinding:

כאשר לוחצים על תמונת הפרופיל של המשתמש באפליקציה, נפתח טופס FormProfile שמציג מידע שמגיע מפרופיל המשתמש. כמעט כל המידע שמופיע בטופס הזה מגיע כתוצאה מ-DataBinding ומסונכרן עם המידע ששמור באפליקציה.

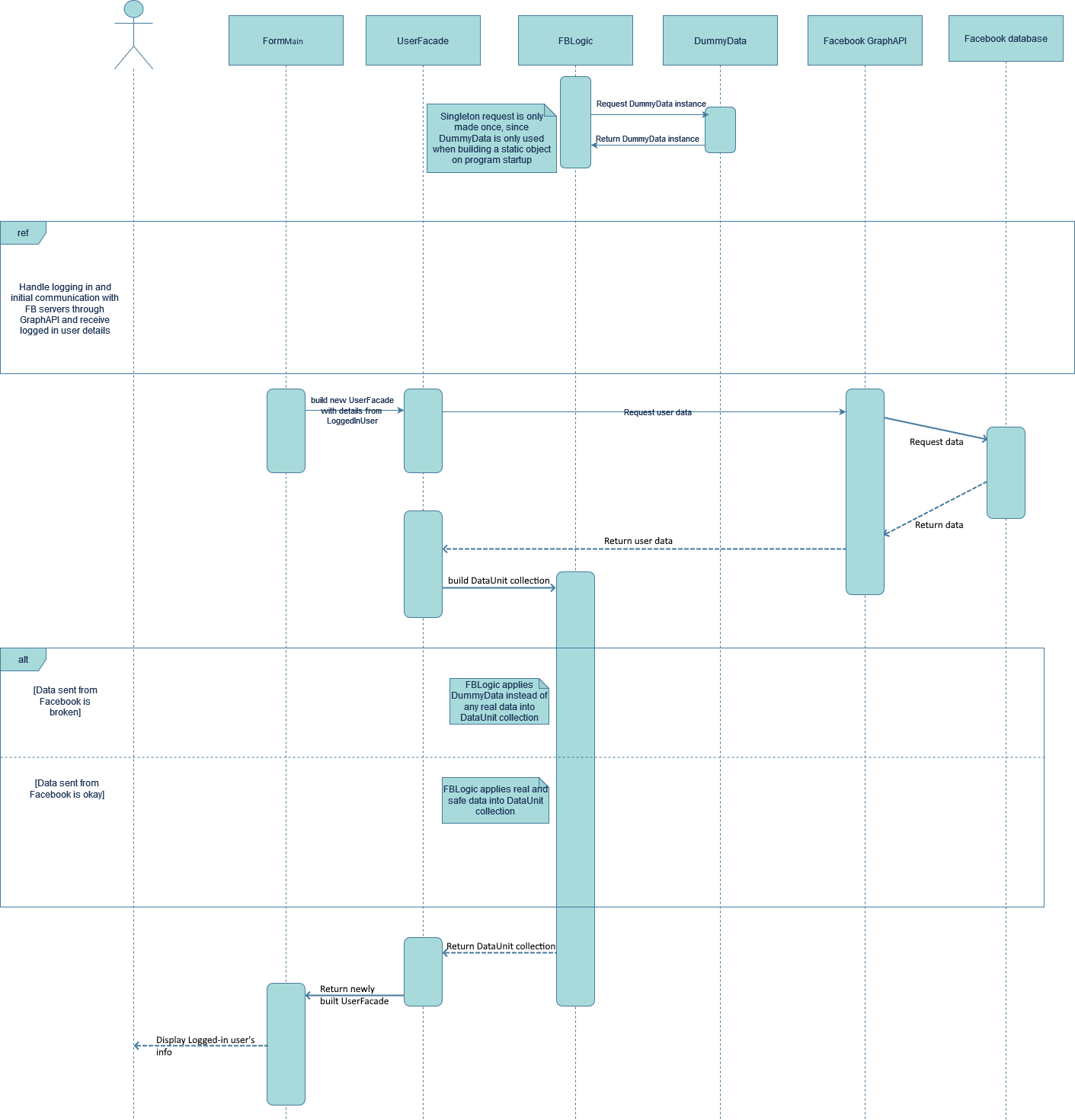
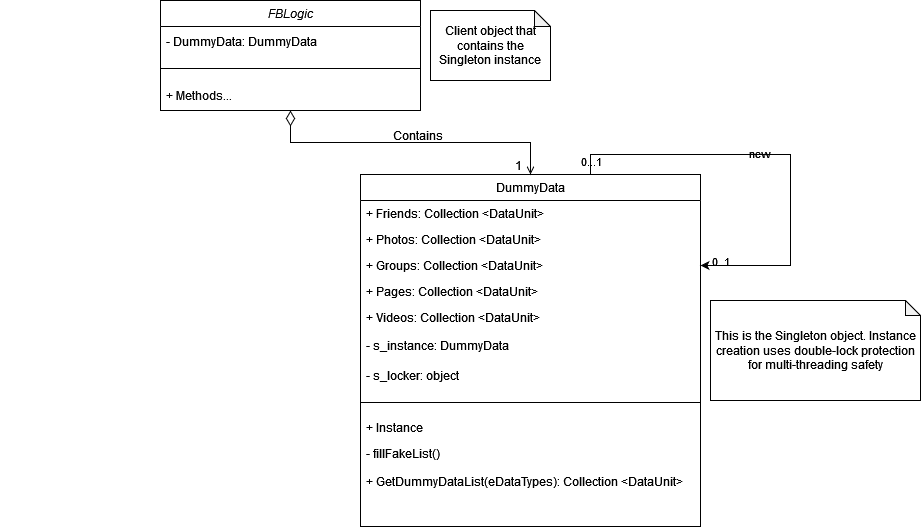
### תבנית מס' 1 – Singleton

* סיבת הבחירה / שימוש בתבנית:

כחלק ממימוש המחלקה הלוגית של האפליקציה, היה צורך לאובייקט שיחיל בתוכו מידע דמה שישומש במקום מידע שבור שמגיע משרתי פייסבוק, על מנת ליצור חווית משתמש חלקה יותר. האובייקט הזה ממומש כ- Singleton כי אין צורך לוגי ביותר מ-instance אחד ממנו, ובנוסף ישנו חיסכון במקום בכך שאין צורך ליצור אובייקט שכזה מחדש בכל שימוש שלו.

* אופן המימוש:

אובייקט מטיפוס DummyData ממומש כ-Singleton בזמן הבניה שלו, והמופע היחיד שלו הוא בתוך מחלקת FBLogic שמנהלת את הלוגיקה של כל האפליקציה. המימוש הסינגלטוני קורה בעזרת שימוש ב- Double check lock בזמן בניה ושליחת instance של DummyData.

* Sequence Diagram
* Class Diagram

### תבנית מס' 2 – Strategy

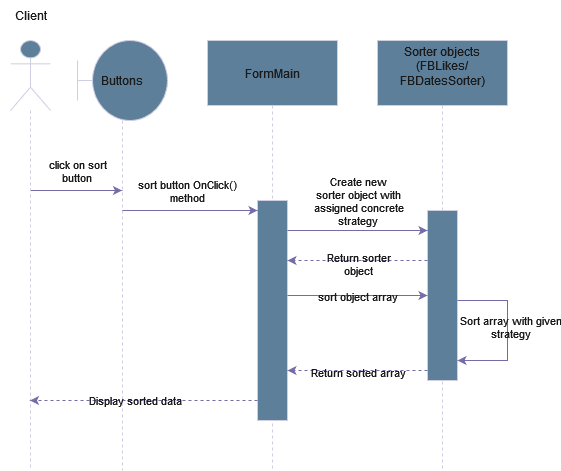
* סיבת הבחירה / שימוש בתבנית:

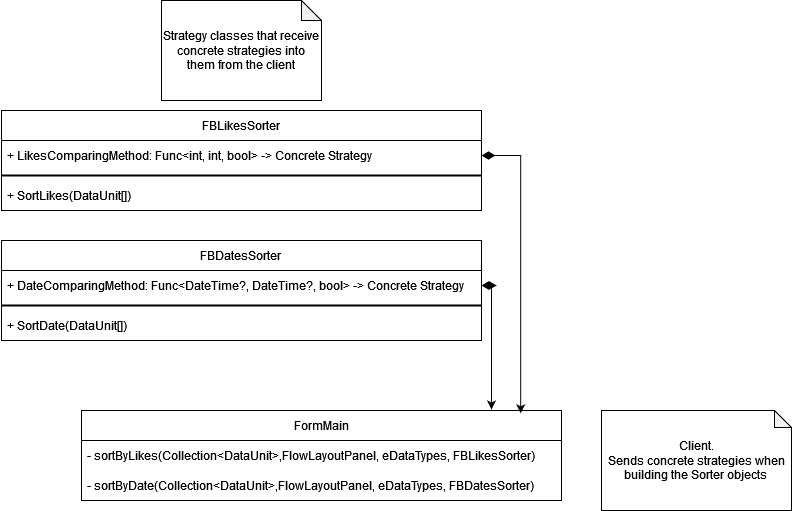
על מנת להוסיף שני פיצ'רים נוספים לאפליקציה שהם מיון אובייקטים לפי מספר לייקים או לפי תאריך, השתמשנו ב-Strategy כדי לממש את צורות המיון השונות, ולכיוונים שונים.

* אופן המימוש:

המחלקות FBLikesSorter ו- FBDateSorter מכילות בתוכן property מטיפוס Func בשם LikesComparingMethod או DateComparingMethod שמקבל לתוכו מתודה שתהווה שיטת ההשוואה שהמחלקה תשתמש בה. בנוסף ישנה מתודה בשם SortLikes/SortDate שמקבלת מערך של אובייקטים מטיפוס DataUnit (הטיפוס הכללי עבור עצם עם מידע באפליקציה כולה) וממיינת את המערך לפי שיטת ההשוואה שמביאים למחלקת המיון.

* Sequence Diagram



* Class Diagram

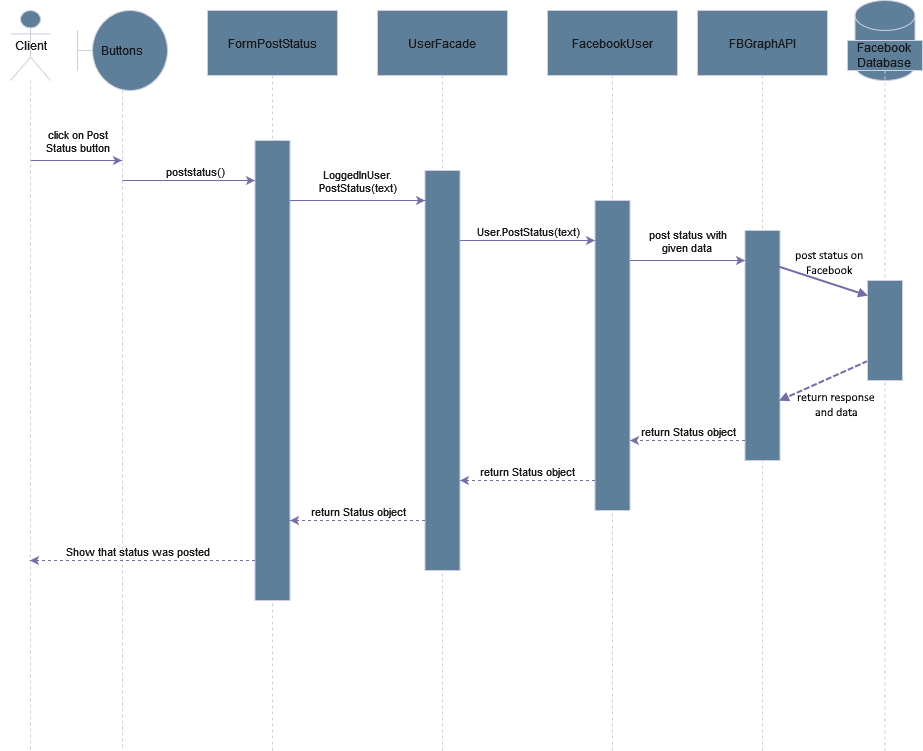
### תבנית מס' 3 – Facade

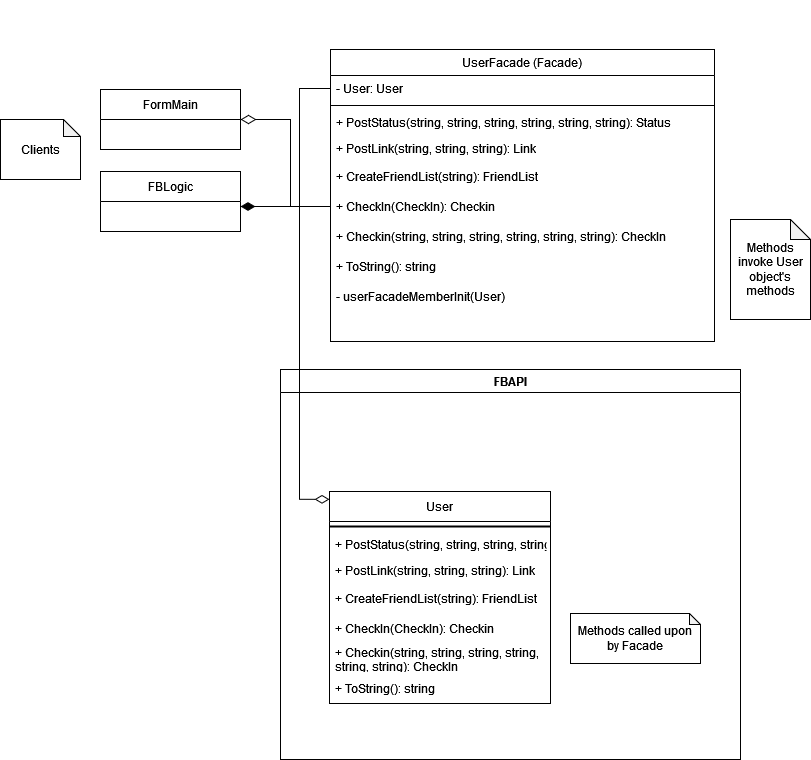
* סיבת הבחירה / שימוש בתבנית:

הרבה מאוד מהמידע שמתקבל משרתי פייסבוק דרך FBGraphAPI מגיע כ-null, אם זה בגלל הרשאות שלא קיימות או בעיות בחיבור האינטרנט. הופעה של מידע שבור דורשת התחשבות מיוחדת מצד הלקוח, ובדיקה וטיפול מיוחד בכל מקרה קצה רלוונטי הנוגע למידע הזה. כל ההתייחסות הזאת עלולה להוביל לחווית משתמש לא חלקה ולא נוחה, משום שחלקים משמעותיים מהאפליקציה לא יהיו פתוחים בפני המשתמש בגלל החוסר בהרשאות האלו. על מנת לתקן את זה, בחרנו להכניס לתוך אובייקט המשתמש מידע דמה שיעמוד במקום המידע שמגיע משרתי פייסבוק. ובכדי להסתיר את התהליך הזה מהלקוח, בחרנו להשתמש בתבנית Façade. המחלקה מממשת את כל המתודות שאובייקט משתמש צריך, בזמן שהיא מטפלת במידע שבור. לכן המחלקה UserFacade שמממשת את תבנית זו, משמשת כשילוב בין תבנית Façade ותבנית Adapter.

* אופן המימוש:

המחלקה UserFacade מממשת את אותן מתודות שיש באובייקט User שניתן לקבל מפייסבוק, ובכל קריאה למתודות האלו דרך UserFacade יש קריאה לאותן מתודות בתוך אובייקט User מקורי שנשמר כ-property בתוך UserFacade.

* Sequence Diagram



* Class Diagram

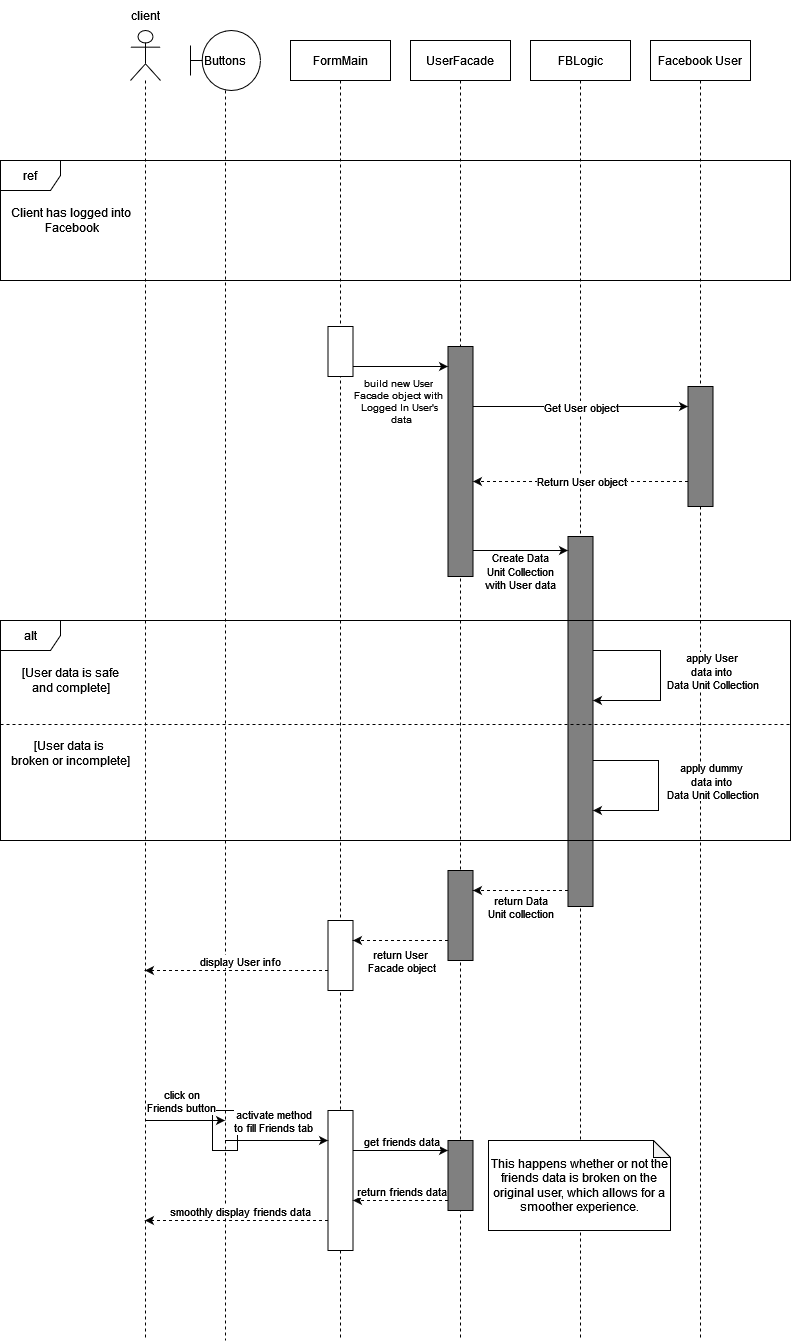
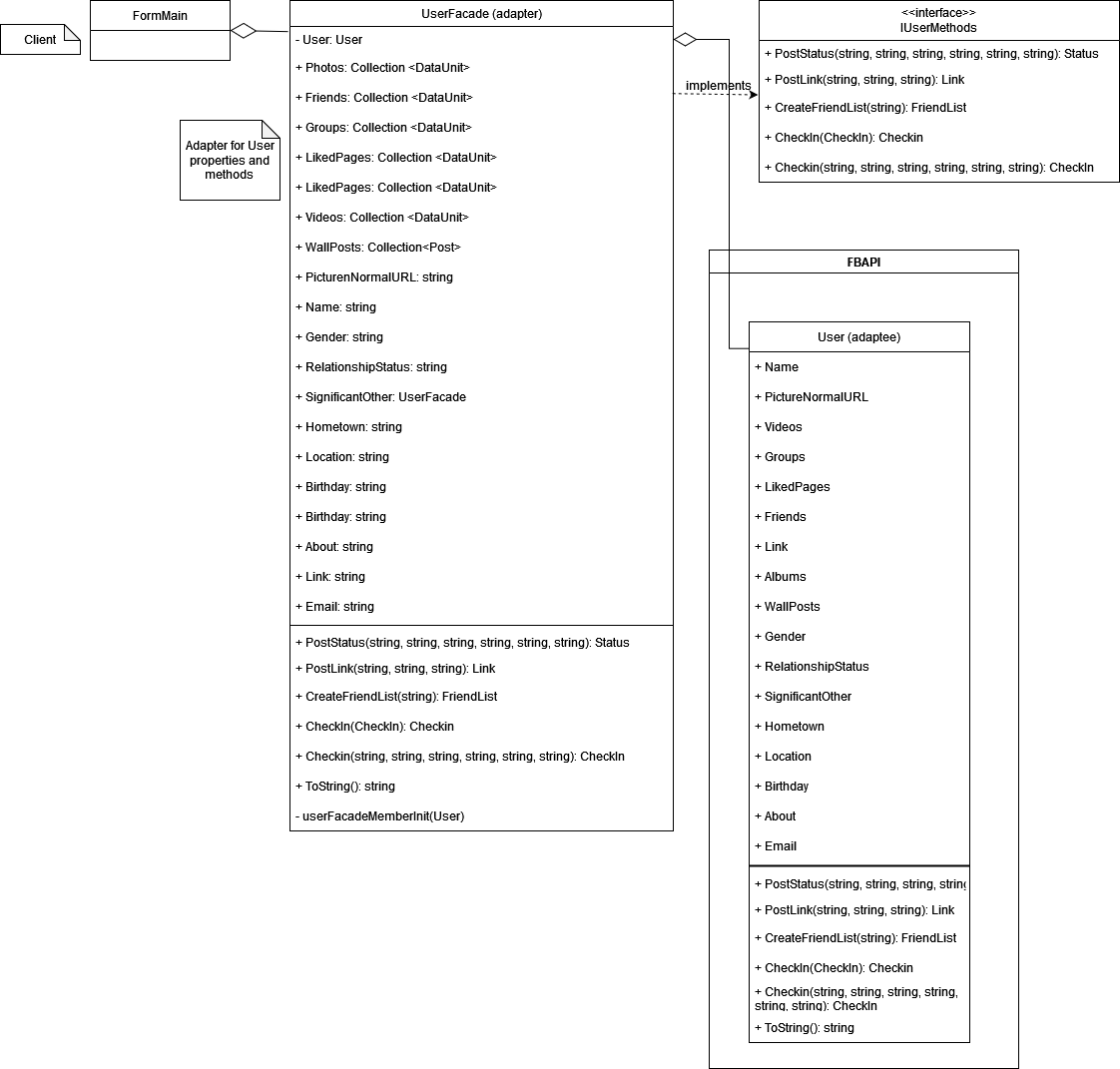
### תבנית מס' 4 – Adapter

* סיבת הבחירה / שימוש בתבנית:

השתמשנו בתבנית Adapter ביחד עם תבנית Façade, והסיבה לכך מתוארת בעיקר בהסבר לשימוש בתבנית Facade. בגלל שנדרשת אדפטציה של מידע שייתכן שמגיע שבור מפייסבוק, השתמשנו בתבנית Adapter על מנת להעביר בצורה חלקה חלקים משמעותיים מאובייקט המשתמש המגיע מפייסבוק לתוך האפליקציה.

* אופן המימוש:

מחלקת UserFacade מממשת ממשק IUserMethods שמכיל בתוכו את אותן מתודות שיש באובייקט User שמגיע מפייסבוק, ובנוסף לכך, היא מכילה כ-Properties חלק מהמידע שמהווה properties של משתמש מפייסבוק. המידע הזה מורכב מכל מיני אובייקטים שונים, ולכן, כדי לפשט את כל הדבר, חלקים מסוימים מהמידע הזה מתורגמים לאובייקטים מטיפוס DataUnit שמכיל בתוכו את המידע הבסיסי והרלוונטי על אותם חלקים. בנוסף, במקרה ויש מידע חסר או שבור, אז יש שימוש במחלקת DummyData שמכניסה מידע דמה לתוך אובייקט המשתמש, כדי שוב להוות מין גישור חלק בין המשתמש לפייסבוק. כל ההתייחסויות האלו מהווים אדפטציה של אובייקט משתמש שמגיע מפייסבוק למשתמש שמאפשר עבודה חלקה בלי שום התייחסות למידע שבור או חסר.

* Sequence Diagram
* Class Diagram