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

* מציאת שידוך פוטנציאלי: (Find Match Feature)  
  הפיצ׳ר מאפשר למצוא התאמות עבור דייטים פוטנציאליים בשביל המשתמש על בסיס העדפותיו, כגון: מגדר וגיל.

בנוסף, הדייטים הפוטנציאליים ממויינים בסדר יורד על בסיס כמות דפים אהובים משותפים שיש בין המשתמש והדייט הפוטנציאלי.

* מבט כללי על חבר נבחר: (Friend OverView Feature)  
  הפיצ׳ר מאפשר לבחור חבר מרשימת החברים ולקבל את כמות האינטרקציות שלו עם הפוסטים של המשתמש, כגון: מספר לייקים והתגובות על פוסטים.  
  בנוסף, הפיצ'ר מראה את המשותף בין החבר למשתמש, כגון: רשימת החברים המשותפים עם המשתמש, דפים שהמשתמש והחבר אהבו, ספורט אהוב משותף ושפות משותפות.

הערה: כל פיצ'ר ממומש בטופס נפרד, אל הטפסים הללו ניתן להגיע דרך כפתורים ייעודיים בטופס הראשי באפליקציה.

### תבנית מס' 1 – Singleton

* סיבת הבחירה / שימוש בתבנית:

באפליקציה שלנו מימשנו מחלקה בשם UserManager, תפקידה הוא לנהל את המשתמש במערכת שלנו אל מול שרתי פייסבוק. המחלקה הזו מאפשרת פעולות התחברות והתנתקות מפייסבוק ובנוסף מחזיקה מידע מפייסבוק אודות המשתמש (ה-User) המחובר כעת למערכת.

מחלקות שונות במערכת עושות שימוש במחלקה UserManager, כדי לבצע התחברות או התנתקות, אך בעיקר כדי לשלוף נתונים על המשתמש המחובר למערכת.

כחלק מאפיון האפליקציה קיים משתמש אחד בלבד המחובר למערכת, לכן לא רצינו לאפשר יצירה של יותר ממופע יחיד של המחלקה (אם יהיה יותר ממופע 1 זה לא תקין). בנוסף רצינו לאפשר לרכיבים שונים במערכת כולל רכיבים עתידיים, לעשות שימוש במופע היחיד של המחלקה ביוזמתם.

התבנית Singleton Pattern עונה לצרכינו מהאפיון, לכן בחרנו לממש את המחלקה UserManager כמחלקה סינגלטונית.

* אופן המימוש:

בחרנו לממש באופן המלא של סינגלטון, על ידי המימוש הבא:

הסתרה של בנאי המחלקה, כלומר: private constructor כדי שלא יהיה ניתן לייצר מופעים מחוץ למחלקה.

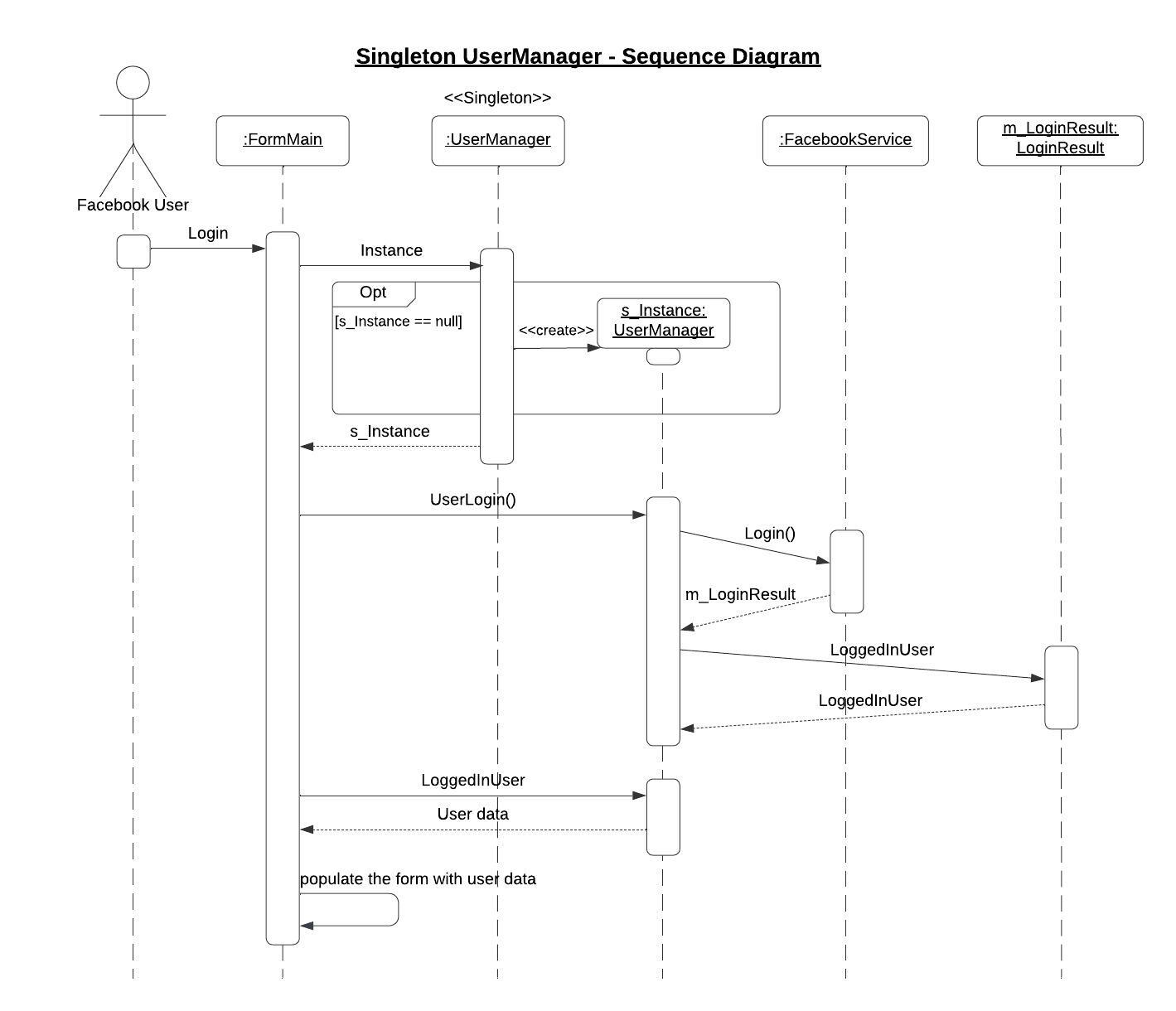
הגישה למופע של המחלקה נעשה באמצעות private static רפרנס למופע היחיד של המחלקה, שנקרא s\_Instance.

הגדרת המחלקה כ- sealed כדי לעזור בהתלבטות שלא יהיה ניתן לרשת מהמחלקה.

המימוש הוא Thread safe על ידי שימוש במנעול שעוטף את שלב יצירת האובייקט.

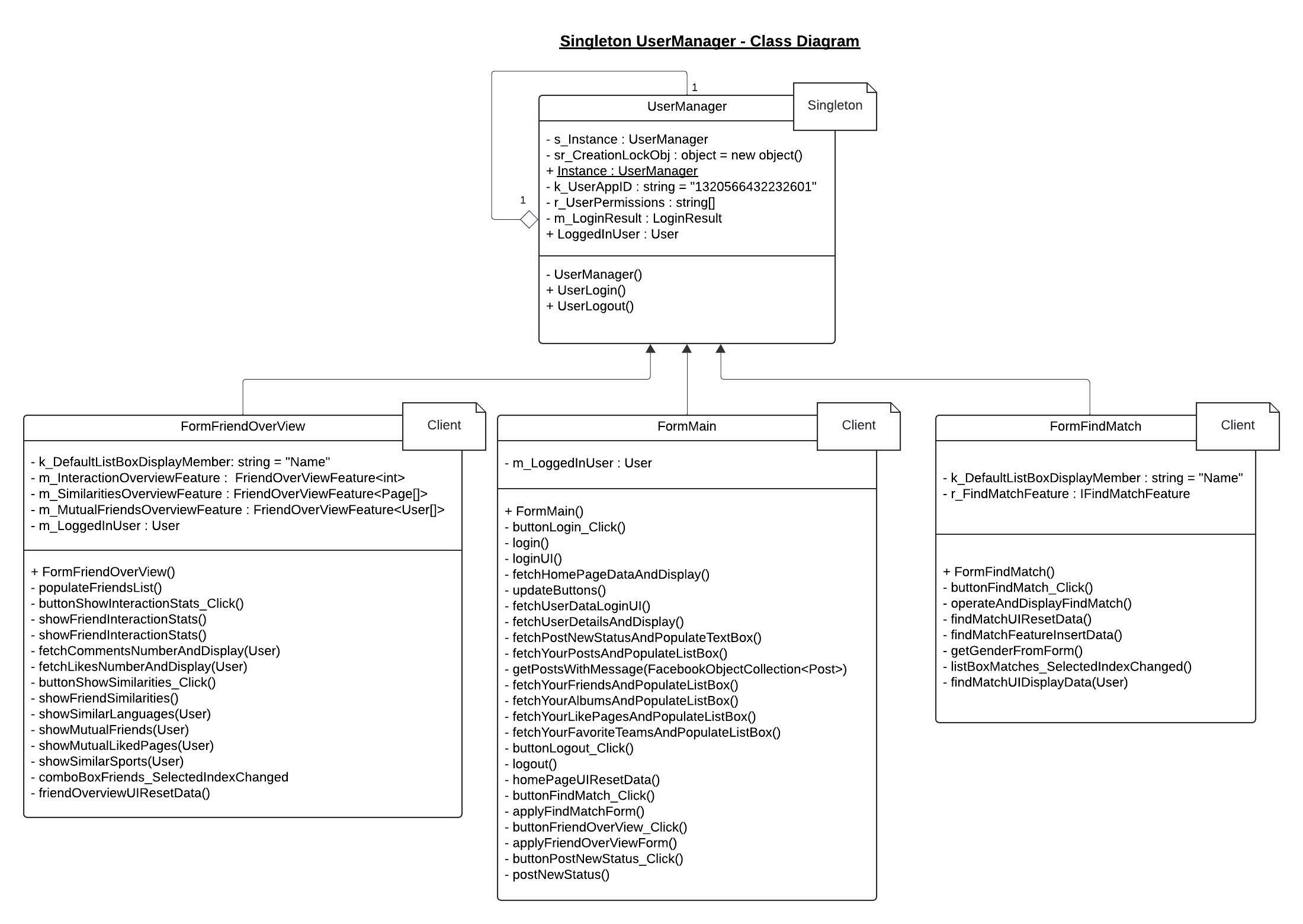
שימוש ב- Double check lock שהוא פטרנס יישומי כדי לחסוך פעולה יקרה של נעילה.

* Sequence Diagram

של האינטרקציה המורכבת ביותר.

* Class Diagram

**Singleton:** UserManager

**Clients:** FormMain, FormFindMatch, FormFriendOverView

### תבנית מס' 2 – Factory Method

* סיבת הבחירה / שימוש בתבנית:

[תיאור הסיבה / הצורך בשימוש בתבנית במערכת שלכם]

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram

[sequence diagram שמציגה את התהליך שקשור לתבנית והאינטראקציה בין המחלקות]

* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם

### תבנית מס' 3 – Proxy

* סיבת הבחירה / שימוש בתבנית:

[תיאור הסיבה / הצורך בשימוש בתבנית במערכת שלכם]

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram

[sequence diagram שמציגה את התהליך שקשור לתבנית והאינטראקציה בין המחלקות]

* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם

### תבנית מס' 4 – Strategy

* סיבת הבחירה / שימוש בתבנית:

[תיאור הסיבה / הצורך בשימוש בתבנית במערכת שלכם]

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram

[sequence diagram שמציגה את התהליך שקשור לתבנית והאינטראקציה בין המחלקות]

* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם