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

**Friends With Common Interest**

פיצ'ר זה מציג את רשימת החברים איתם למשתמש יש עמודי פייסבוק, להם המשתמש והחבר שלו עשו "לייק". ברשימוה מוצגים כל החברים שיש להם לפחות עמוד אחד משותף.

הפונקציה שמבצעת את הפיצ'ר הנ"ל ממומשת (במטלה הראשונה) במחלקה AppLogic (במטלה הנוכחית ממומשת ב-FacebookUserProxy) והטופס הראשי (FormMain) משתמש בה בעת לחיצה על הכפתור "Click here to see your friends that like similar pages"

**MyEvents**

פיצ'ר זה מציג את רשימת האירועים להם המשתמש עשה Attend. למשתמש יש את האפשרות לסנן את האירועים שמוצגים לפי מיקום ההתרחשות שלהם – מתרחשים אונליין או מתרחשים במקום פיזי.

הפונקציה שמממשת את הפיצ'ר הנ"ל ממומשת (במטלה הראשונה) במחלקה AppLogic (במטלה הנוכחית ממומשת ב-FacebookUserProxy) והטופס הראשי (FormMain) משתמש בה בעת לחיצה על הכפתור "Fetch events". על הכפתור המשתמש ילחץ לאחר בחירה של אופציית ההצגה הרלוונטי.

**שימוש ב-Data Binding**

השתמשנו בפיצ'ר זה בפיצ'ר MyEvents. כמו כן, קישרנו את שמות ה-events לחלק מהשדות שלו והם מוצגים בהתאם לאירוע הנוכחי שהמשתמש בוחר לראות מידע עליו.

השימוש בקוד הוא במחלקת ה-MainForm, במתודה buttonFetchEvents\_Click() שמופעלת בעקבות הלחיצה על הכפתור "Fetch events".

**שימוש בתכנות אסינכרוני**

בקוד יש 2 שימושים עיקריים בתכנות אסינכרוני:

1. בפיצ'ר Get Nasa's Picture Of the Day שהוספנו (בעקבות הערה שקיבלנו על המטלה הקודמת), במתודות fetchNasaPictureByDate ו-fetchNasaPicToday יש שהן מתודות אסינכורניות שמשתמות במתודות GetNasaPicBYDate, GetNasaPicOfTheDay שממומשות ב- Façadeשיוזכר בהמשך. המתודות הנ"ל משתמשות במתודה GetNasaPictureByDate שנמצאת במחלקה NasaApi.

השימוש במתודות האלה נעשה כך על מנת לאפשר למשתמש פונקציונליות של שימוש שוטף ורציף בזמן טעינת תמונות, שלוקח זמן היות והוא תלוי בתשובה של שרת חיצוני (של NASA API). השימוש נעשה באמצעות thread.

1. ב-FormMain, כל הפונקציות שתפקידן להביא מידע ולהציג אותו למשתמש עושות שימוש בתכנות אסינכרוני, מכיוון שהמידע מיובא משרת חיצוני של פייסבוק. זאת באמצעות שימוש ב-threads – כמובן עם שימוש בפעולות מהצורה Invoke new Action כדי שלא תהיה חפיפה בין ה-threads (Cross thread operation). השימוש בתכנות אסינכרוני, באופן דומה לנקודה הראשונה, נעשה כדי לאפשר למשתמש להינות משימוש שוטף באפליקציה בעת שהנתונים נטענים.

### תבנית מס' 1 – [Singelton]

* **סיבת הבחירה / שימוש בתבנית**:

השתמשנו בסינגלטון עבור מחלקת הלוגיקה בתוכנית. מחלקה זו מחזיקה נתונים ומבצעת פעולות לוגיות עבור כל הטפסים השונים במערכת (זה שקיים וכאלה שיכולים להיתווסף בהמשך). בחרנו להשתמש בסינגלטון עבור מחלקה זו כי רצינו לוודא שיש מקור יחיד של אמת עבור הנתונים השונים בלוגיקה של התוכנית, וכדי למנוע מצב בו יש כפילויות של נתונים אודות המשתמש או עבור תוצאות פעולות בלוגיקה. בנוסף רצינו שהנתונים והמופע של מחלקת הלוגיקה יהיה נגיש מכל טופס ללא צורך "להעביר" את המופע הזה בין הטפסים – גישה נוחה מנקודות שונות במערכת אשר תבנית סינגלטון מאפשרת.

* **אופן המימוש**:

המימוש במחלקת AppLogic (ובנוסף בטפסים המבצעים AppLogic.instance) :

* הבנאי בעל access modifier - private.
* מופע סטטי בתוך המחלקה.
* Static property שמחזיר את המופע הסטטי, ורק במקרה שעדיין לא נוצר מופע שלו, הוא יוצר כזה.
* Thread Safe: שימוש בlock.

"שחקנים" :

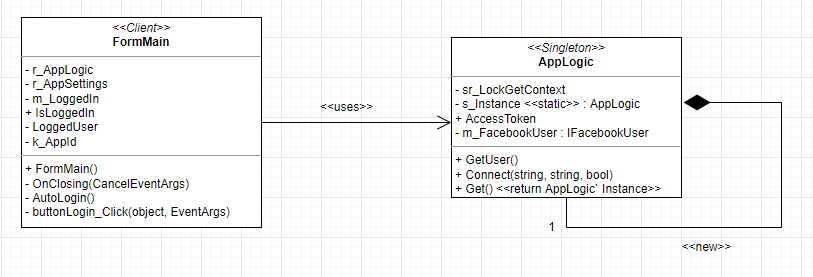
Singelton: AppLogic

Clients: FormMain

* **Sequence Diagram**

A picture containing text, indoor, white

Description automatically generated

* **Class Diagram**

### תבנית מס' 2 – [Proxy]

* **סיבת הבחירה / שימוש בתבנית**:

עבודה בטוחה ומוסדרת של FormMain מול ממשק חיצוני -Facebook User .

זהו -Protective Proxy , כיוון שFacebook User הינו ממשק חיצוני המשתמש בשרתי פייסבוק. אנו רוצים להגן על המשתמש מפני תקלות שיכולות להיות בשרת, בזמן השימוש שלו ב-FormMain (תקלה אפשריות היא שישנן מתודות שקיימות, אך בפועל בעבודה מול פייסבוק עלולות להיות תקלות והמידע לא יתקבל, או לא יתקבל בצורה תקינה).

הגדרנו בממשק מסודר מהן המתודות שבטוח להשתמש בהן. בנוסף, ביצענו אינקפסולציה – Facebook User חשוף באמצעותProxy , ובאמצעות הממשק שבו יש את המתודות אליהן יש התאמה, ובכך גם משיג בטיחות שימוש – ה-FormMain ניגש למתודות דרך הממשק.

* **אופן המימוש**:

הממשק IFacebookUser מגדיר את המתודות שעל ה-Proxy לממש, ו-FormMain מכיל מופע של הממשק – IFacebookUser.

FacebookUserProxy מכיל מופע של User, ומממש את הממשק IFacebookUser באמצעות שימוש בProperties של User, וכך FormMain מממש את המתודות שמשמשות לעבודה מול Facebook User תוך שימוש בממשק, הממומש על ידי ה-Proxy, ומכיל מופע של User.

"שחקנים":

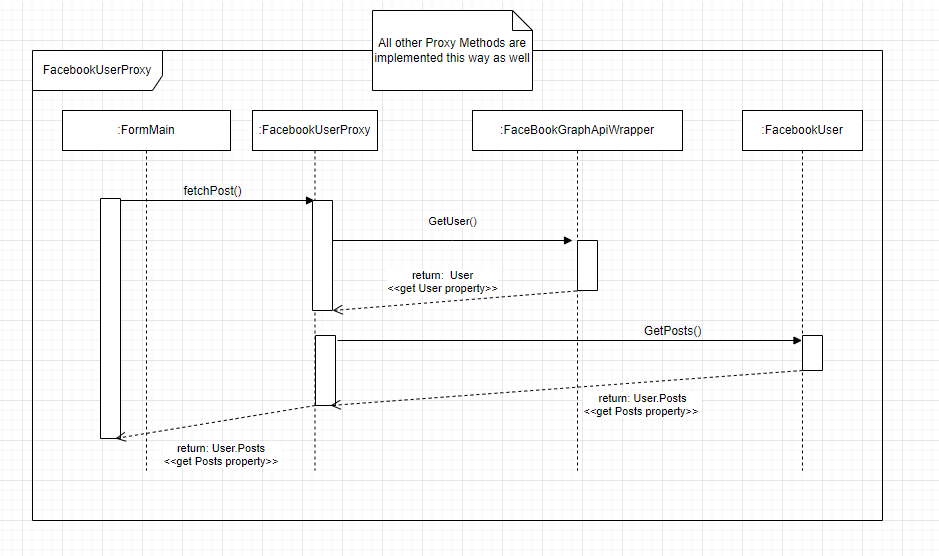
Interface: IFacebookUser

Client: FormMain

Proxy: FacebookUserProxy

RealSubject: User (from FacebookGraphApiWrapper)

* **Sequence Diagram**



* A picture containing schematic

  Description automatically generated**Class Diagram**

### תבנית מס' 3 – [Façade]

* **סיבת הבחירה / שימוש בתבנית**:

השתמשנו בתבנית זו כדי לאפשר לטופס הראשי עבודה מולApi חיצוני המספק תמונה יומית של החלל מהאתר של Nasa.

ביצענו זאת מכמה סיבות: ראשית, לצורך בטיחות שימוש בעבודה מול Api חיצוני שעלולות להיות בו תקלות. שנית, על מנת לפשט את העבודה של FormMain מול הApi (ה-Façade מחזיר בצורה נוחה את הנתונים הרלוונטים לטופס). כמו כן, ביצענו זאת לצורך תחזוקתיות ופולימורפיזם – אם בעתיד נרצה להשתמש ב-Api אחר של שליפת תמונה במקום הנוכחי (Nasa Api), נצטרך לבצע שינויים רק ב-Façade ולא בשכבה המשתמשת בו.

* **אופן המימוש**:

במחלקת NasaFacade (והשימוש בה בFormMain-)

המימוש הוא של Façade אטום:

מחלקת NasaApi שאיתה ה-Façade עובד , מוגדרת עם access modifier של internal ומחלקת ה-Façade נמצאת באותו dll.

ה- FormMainמחזיק מופע של NasaFacade.

ה-Façade מחזיק מופע של המחלקה NasaApi – ומשתמש במתודות שלו כדי למממש את המתודה GetNasaPictureByDate שמונגשת באופן נוח ל-FormMain המשתמש ב-Façade.

"שחקנים":

Façade – NasaFacade

FormMain – client

NasaApi – package

* **Sequence Diagram**

Chart

Description automatically generated

* **Class Diagram**

A picture containing chart

Description automatically generated