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

* **Friend Quiz:**   
  **אפשרות לקיום שאלון קצר על פעילות של חבר פייסבוק.**   
  **כולל: פנייה לקובץ טקסט חיצוני בו מופיעות השאלות,**   
  **הצגת שאלה למשתמש ומציאת התשובה בזמן אמת מתוך ה**database של פייסבוק + יצירת אופציות לתשובות השגויות.    
  לכל שאלה יש זמן מוגבל. בסיום הזמן או בבחירת אופציה - השאלה תסתיים ויופיע אינדיקטור לתשובה הנבחרת או הודעה מתאימה במקרה שלא הספקת בזמן.    
  בגמר השאלות, תופיע הודעה עד כמה אתה מכיר את החבר והשאלון ייסגר.    
  את הפיצ'ר ניתן למצוא בחלון הראשי(MainForm) כאשר המשתמש מחובר ליד רשימת החברים.
* **הצגת תמונות מהאלבומים של משתמש הפייסבוק על פי סינונים:**

בפייסבוק ניתן לראות תמונות מאלבומים באופן רגיל, ללא סינונים מיוחדים.  
אצלנו באפליקציה ניתן לראות תמונות של המשתמש לפי סינון של תאריך ובנוסף לסנן לפי מיקום. כמובן שגם ניתן לראות את התמונות מאלבום התמונות ללא סינון כלל.

את הפיצ'ר ניתן למצוא בחלון ה-FormAlbum כאשר המשתמש מחובר על ידי לחיצה על כפתור ה- show album's photo.

### תבנית מס' 1 – [Strategy]

* סיבת הבחירה / שימוש בתבנית:

בחרנו בתבנית העיצוב Strategy מכיוון ששמנו לב שבפיצ'ר של הפילטור תמונות, אנחנו מבצעים את אותה הלוגיקה, רק עם שינוי קל - מסננים לפי תאריך ולפי מיקום. בעקבות זאת, במקום לשכפל את הקוד ב-2 פונקציות נפרדות, החלטנו להשתמש בתבנית העיצוב שמטרתה לעשות Injection point ובכך להפוך את הקוד ליותר אקסטנסיבילי(במידה ובעתיד נרצה עוד שיטות פילטור).

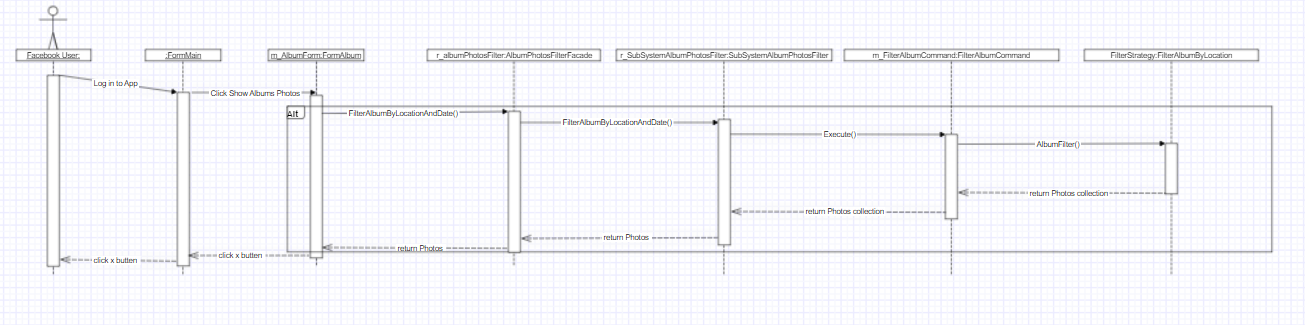
* אופן המימוש:

יצרנו ממשק בשם **IAlbumPhotosFilterStrategy**, אשר מכיל חתימה של המתודה **AlbumFilter**. את הממשק מממשות שני מחלקות, אשר כל אחת מהן מממשת את המתודה בממשק בצורה הרלוונטית עבורה: **FilterAlbumByLocation, FilterAlbumByDate**.

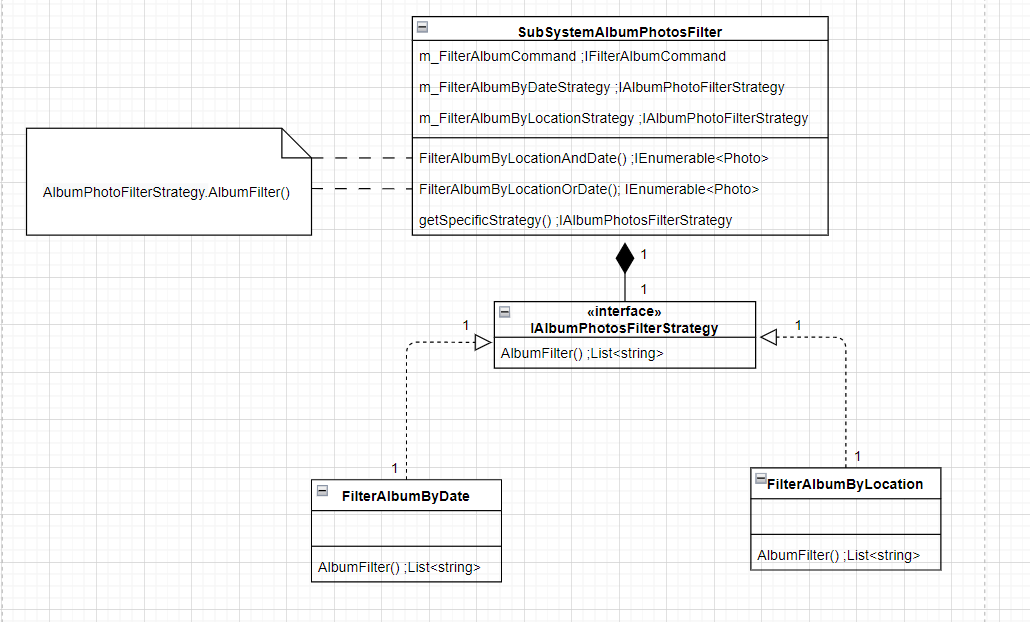
בתוך המחלקה שאחראית על הלוגיקה של הפיצ'ר של הפילטור - **SubSystemAlbumPhotosFilter**, יצרנו מופע של הממשק **IAlbumPhotosFilterStrategy**. זה נתן לנו בסיס פולימורפי, כך שנוכל לשנות אותה בעת הצורך.

* Sequence Diagram

**במידה והתמונה לא ברורה ,מצורף תמונה בתיקיית הפרוייקט בשם :Sequence Diagram Strategy**



* Class Diagram



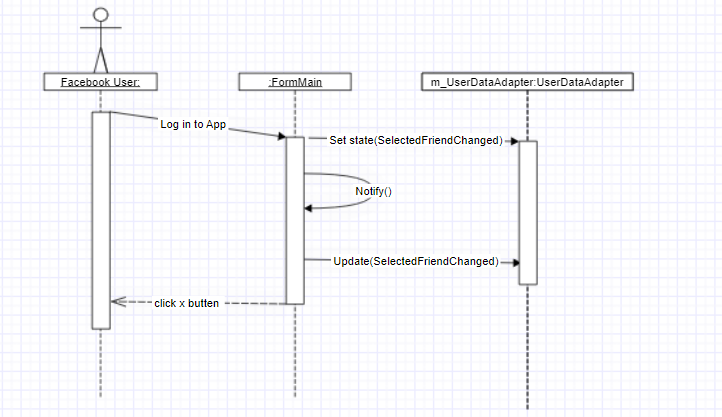
### תבנית מס' 2 – [Observer]

* סיבת הבחירה / שימוש בתבנית: הסיבה שהשתמשנו בתבנית הזו היא שבמקום שנייצר כל פעם את האובייקט (UserDataAdapter) עבור אותו אינדקס אנחנו מייצרים פעם אחת מופע של UserDataAdapter ואותו מופע מאזין לאירוע עם המתודה SelectedFriendChanged וכך הוא יודע להכניס את המידע עבור אותו User שנבחר בListbox
* אופן המימוש:

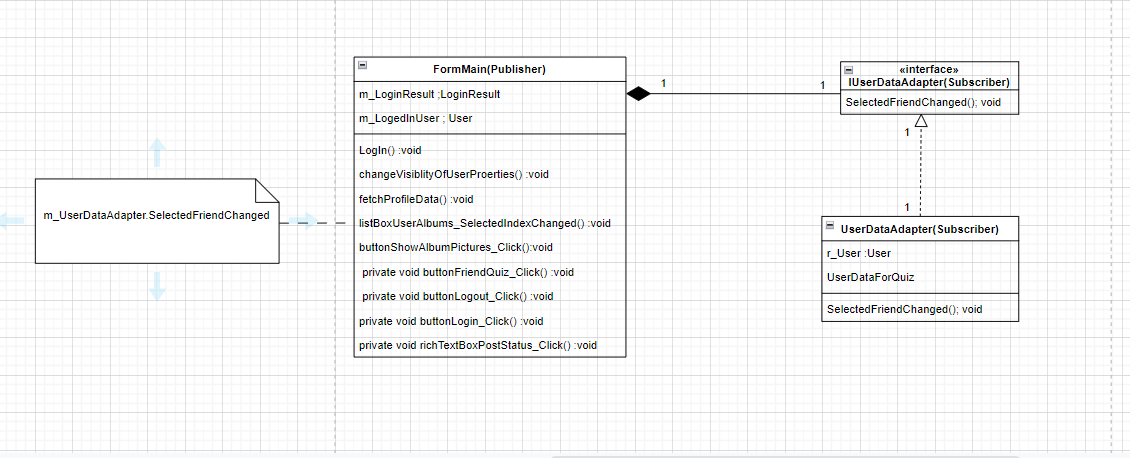
יצרנו את מופע של המחלקה **UserDataAdapter(Subscribers)** אשר מחזיקה במתודה SelectedFriendChanged אשר אחראית לשנות את בחירת החבר שעליו נרצה לעשות את הQuiz בהתאם לאירוע שיצרנו שאליו הוא נרשם(SelectedIndexChanged),

המחלקה FormMain מייצרת מופע של UserDataAdapter רק פעם אחת ונרשמת לאירוע SelectedIndexChanged עם הפעלת המתודה SelectedFriendChanged של המחלקה m\_UserDataAdapter .

* Sequence Diagram



* Class Diagram



### תבנית מס' 3 – [Command]

* סיבת הבחירה / שימוש בתבנית:

הסיבה שבחרנו להשתמש בתבנית זו היא כדי לעטוף ולהסתיר את המידע הרלוונטי להרצת

המתודה FilterAlbum ללא שום פרמטרים עם אפשרות הרחבה בעתיד לעוד קומנדים נוספים.

* אופן המימוש:

יצרנו Interface בשם IFilterAlbumCommand כדי לתת בסיס פולימורפי במידה ונרצה להוסיף עוד Command בהמשך ויצרנו מופע שלה בתוך

ה-SubSystemAlbumPhotosFilter, בנוסף יצרנו מחלקה

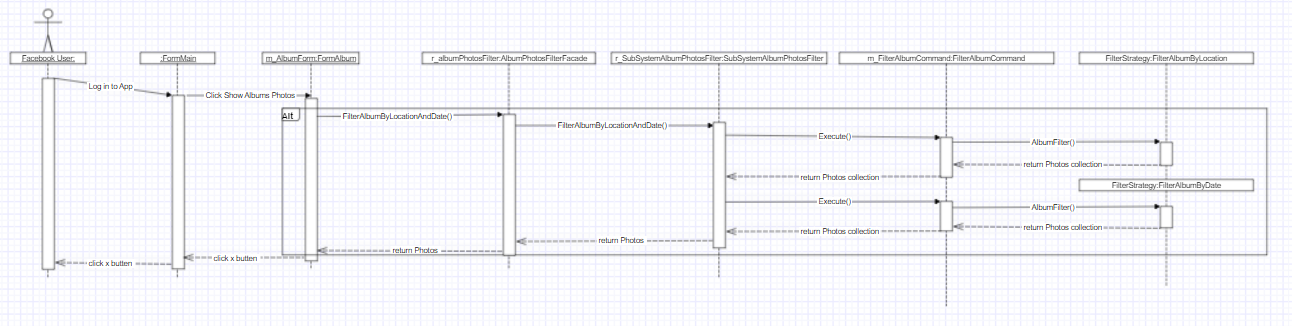
בשם FilterAlbumCommand כאשר הוא מממש את הInterface ואת

המתודה execute אשר מפעילה את המתודה FilterAlbum בהתאם לאסטרטגיית הפילטור המתאימה שהוא מחזיק בתוך המחלקה כ - Data Member

בשם FilterStrategy מסוג IAlbumPhotosFilterStrategy

* **Sequence Diagram**

**במידה והתמונה לא ברורה ,מצורף תמונה בתיקיית הפרוייקט בשם :Sequence Diagram Command**



* **Class Diagram**