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

* הצגת פרטי חמשת החברים שעשו הכי הרבה לייקים.
* הצגת כל המקומות בהם המשתמש ביקר והעלה מהם תמונות, פוסטים וכו' **בשנה האחרונה**.

### תיאור קצר של פיצ'רים חדשים:

* שינינו את הפיצ'ר של הצגת כל המקומות בהם המשתמש ביקר והעלה מהם תמונות, פוסטים וכו' תוך פילטור וסינון המקומות בהם הוא ביקר בשנה האחרונה ע"י שימוש בתבנית Strategy.
* במחלקה Places אנחנו מחזיקים כעת Dictionary שהמפתח בו הוא Place והערך הוא מס' המייצג את מס' הפעמים בהם המשתמש ביקר באותו מקום.

### תבנית מס' 1 – Iterator

* **סיבת הבחירה / שימוש בתבנית:**

בפיצ'רים שלנו יש עבודה עם רכיבי aggregate, כלומר רכיבים שמחזיקים אוסף של משהו, כגון: טעינת כלל התמונות באלבום, טעינת כלל החברים שעשו לי לייק לפוסט,תמונה,סרטונים ועוד.

רצינו לתת גישה לסריקת כל הקולקציות האלה מבלי לחשוף את מבנה הנתונים שבו הם שמורים ובנוסף, כדי למנוע מצב עתידי שבו נשנה את מבנה הנתונים של אחד מהאובייקטים ואז הclient יצטרך להתאים את הקוד שלו לשינויים.

בנוסף, אם וכאשר המשתמש בעתיד ירצה לעבור רק על חלק מכלל הנתונים, הוא יוכל לעשות את זה ביעילות טובה יותר מאשר לסרוק את כלל מבנה הנתונים.

לסיכום, רצינו לממש ממשק אחיד לעבודה עם מבנים המכילים קולקציה.

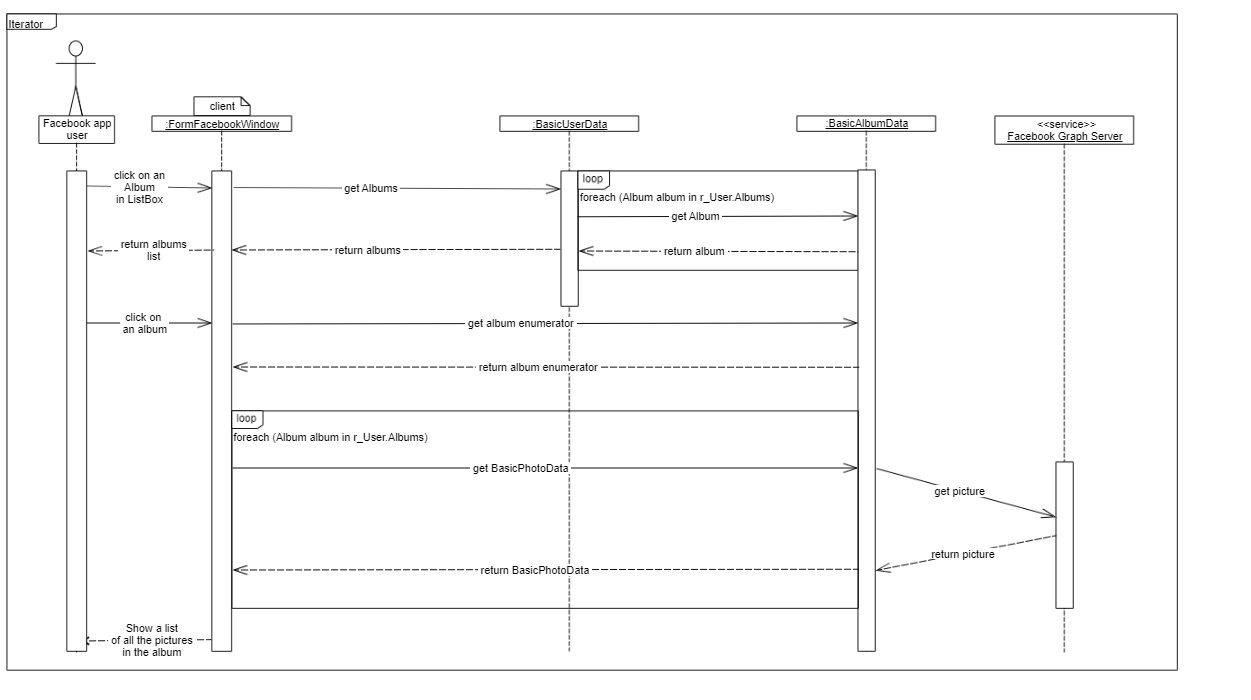
* **אופן המימוש:**

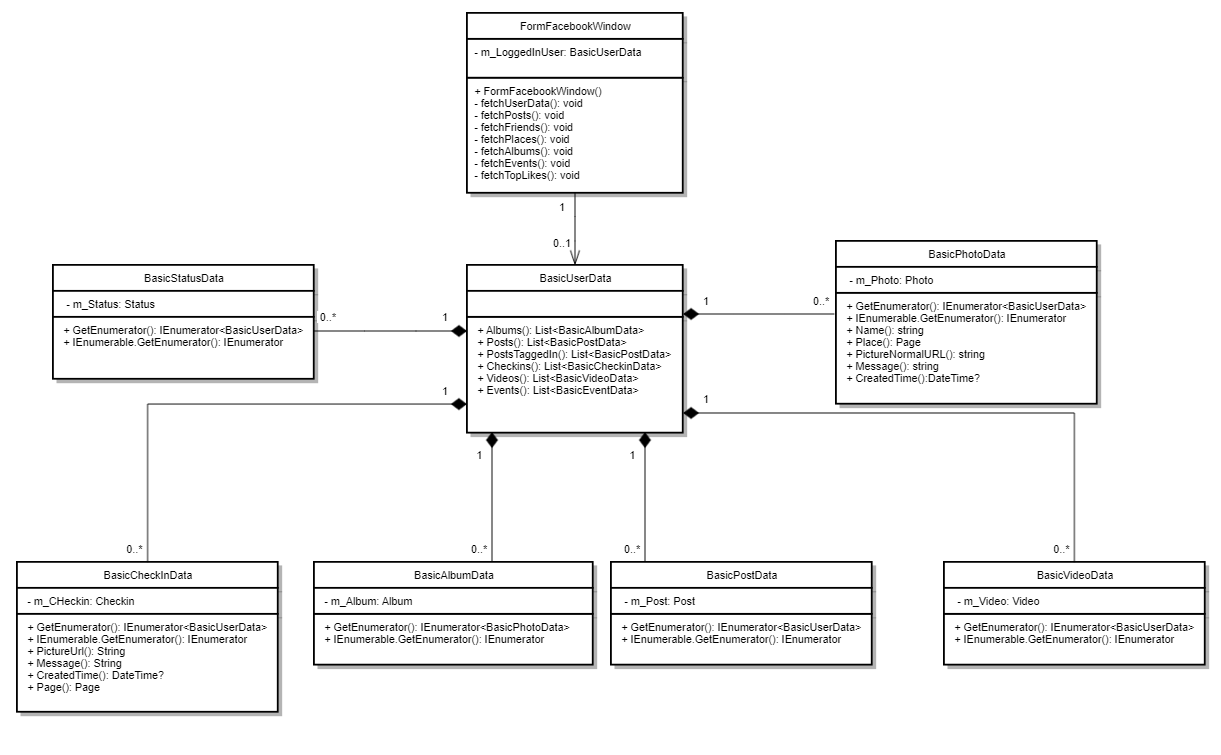
בפרויקט הלוגי - יצרנו מחלקות BasicPostsData, BasicAlbumData, BasicCheckInData, BasicPhotoData, BasicStatusData, BasicVideoData שבהם יש מימוש של Iterator, כל מחלקה היא IEnumerable ולכן מומש עבורה IEnumerator.

עברנו על המבנה/מבני הנתונים הרלוונטי/יים ב Facebook Wrapper והחזרנו איטרטור עבור כל אחת מהמחלקות בהתאם לצרכיה.

בפרוייקט הUI במחלקה FormFacebookWindow – בין היתר השימוש נעשה בפונקציות FetchAlbums כאשר אנחנו מקבלים את האיטרטור מהמחלקה ועבורים איתו על מבנה הנתונים (שלא ידוע לנו) על מנת להשיג את כלל התמונות.

* **Sequence Diagram**



* **Class Diagram**

### תבנית מס' 2 – Strategy

* **סיבת הבחירה / שימוש בתבנית:**

רצינו להשיג אי תלות בין רצונו של הקליינט לבין הממשק הלוגי. ע"י השימוש בתבנית זו הקליינט יכול לסנן כרצונו את כל המקומות בהם המשתמש ביקר. למשל, הקליינט יכול כעת ללא שינוי שורת קוד אחת בלוגיקה לבחור להציג למשתמש את כל המקומות שבהם המשתמש ביקר מתחת ל10 פעמים, בדיוק פעמיים, בשנתיים האחרונות וכו'.  
אי התלות מאפשרת לקליינט להוסיף אפשרויות פילטור רבות בלי צורך בשינוי הפרוייקט הלוגי (maintainability) וכן זה מאפשר reuse של המחלקה FilteredPlaces.

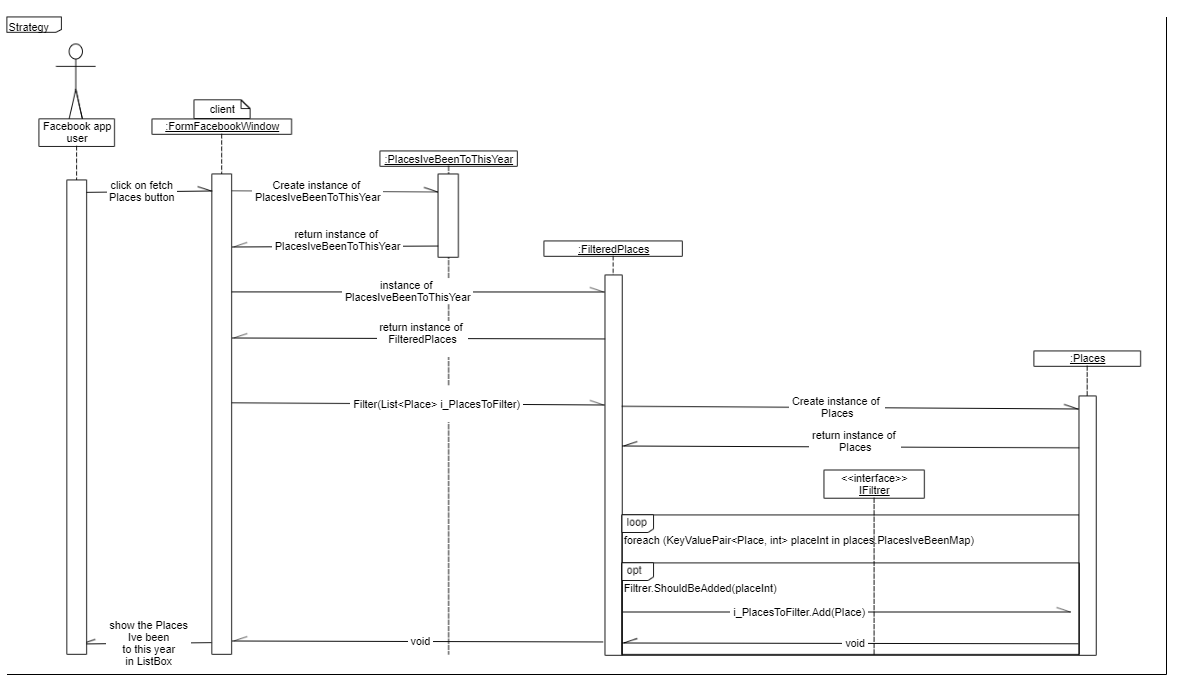
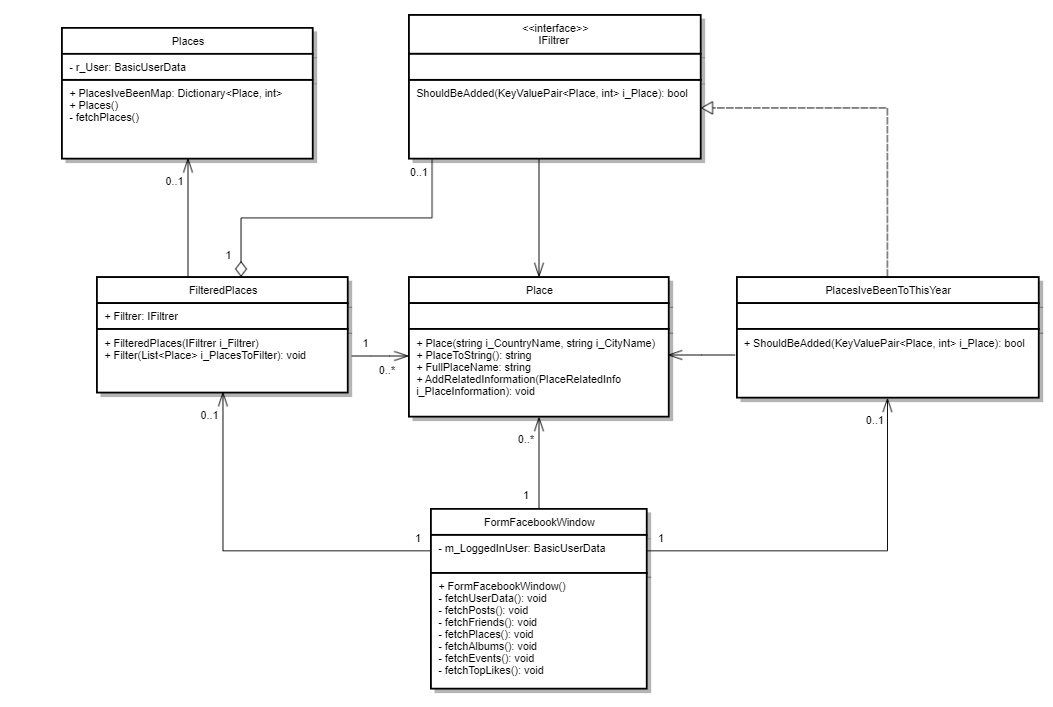
* **אופן המימוש:**

בפרוייקט הלוגי –

1. הוספנו ממשק הנקרא " IFiltrer" ומכיל את המטודה הבאה:  
   bool ShouldBeAdded(KeyValuePair<Place, int> i\_Place)
2. הוספנו מחלקה חדשה הנקראת FilteredPlaces"" ומכילה את מטודת הפילטור:  
   public void Filter(List<Place> i\_PlacesToFilter)  
   ע"י שימוש באגריגציה של הממשק IFiltrer
3. הגדרנו כי המחלקה PlacesIveBeenToThisYear תממש את הממשק " "IFiltrer

בפרוייקט הUI במחלקה FormFacebookWindow –

במטודה FetchPlaces יצרנו אובייקט מטיפוס PlacesIveBeenToThisYear אשר מממש את הממשק IFiltrer ושלחנו אותו לctor של המחלקה FilteredPlaces.

* **Sequence Diagram**
* **Class Diagram**

### תבנית מס' 3 – Template Method

* **סיבת הבחירה / שימוש בתבנית:**

רצינו לאפשר גמישות באופן הפילטור של קולקציית החברים שמכילה המחלקה Friends.

בפרט רצינו לאפשר לקליינט לפלטר את קולקציית החברים שלו לפי חמשת החברים שעשו לו הכי הרבה לייקים בכל פוסט, תמונה, פוסט וכו' שהעלה.

התבנית מאפשרות לקליינט להוסיף בקלות דרך פילטור חדשה במידה והוא ירצה לממש אחת כזו בעתיד. כלומר, ניתן לשנות את המימוש או להוסיף פילטור חדש באופן עצמאי ללא שינוי שורת קוד אחת בפרוייקט הלוגי ובלי תלות במחלקה Friends עצמה.

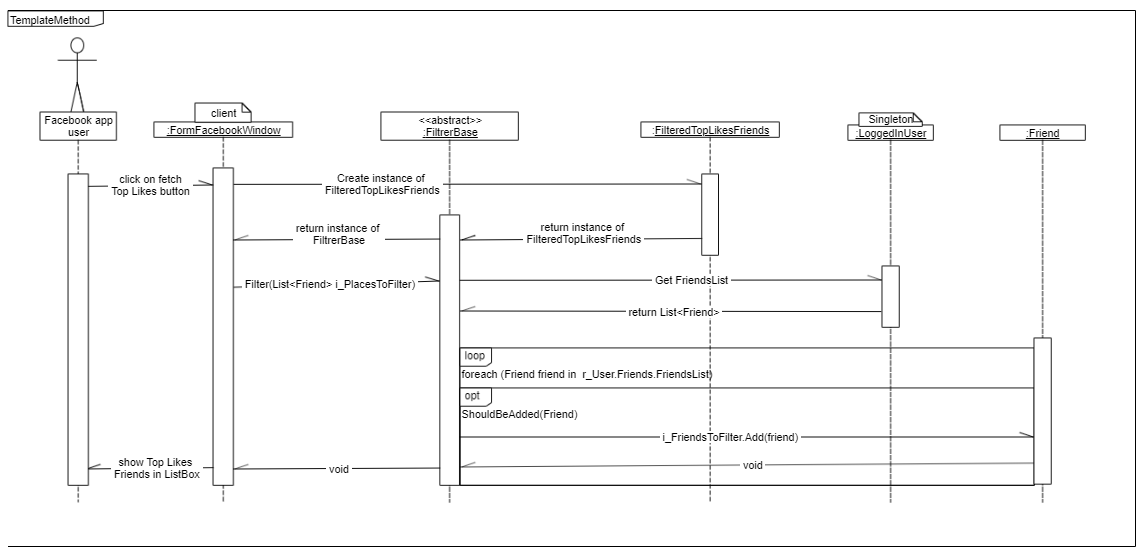
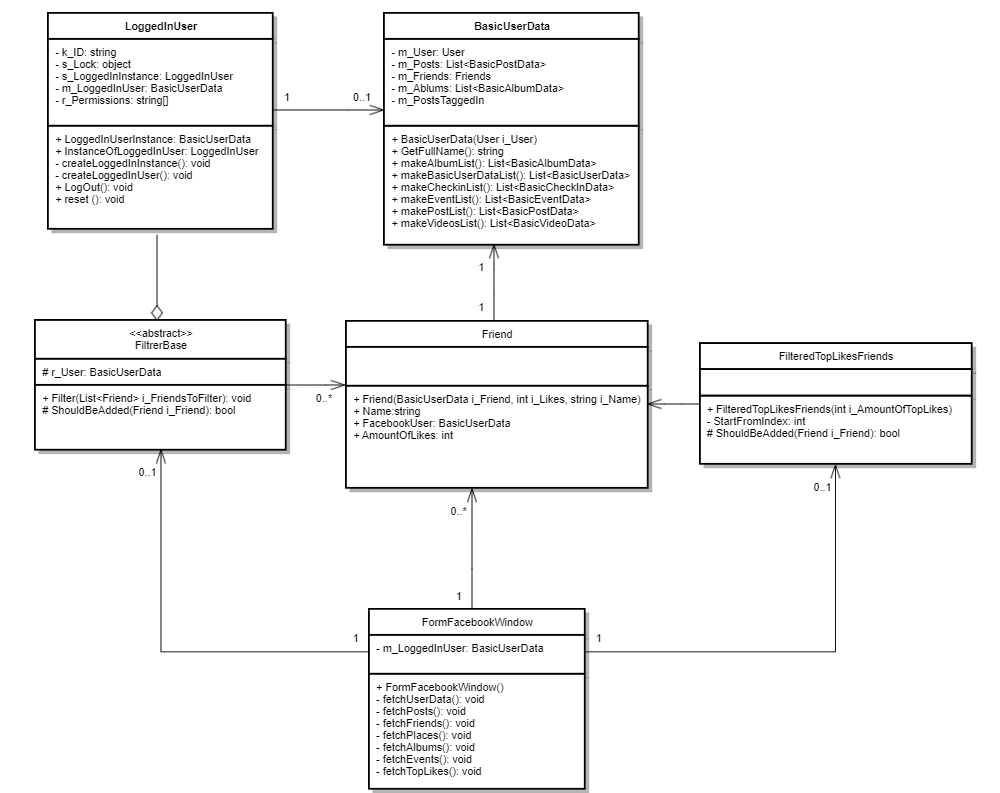
* **אופן המימוש:**

בפרוייקט הלוגי-

1. הוספנו מחלקה אבסטרקטית חדשה הנקראת FiltrerBase"" אשר מכילה את מטודת הפילטור:  
   public void Filter(List<Friend> i\_FriendsToFilter)  
   ואשר דורשת מהיורשים שלה לממש את המטודה האבסטרקטית:  
   protected abstract bool ShouldBeAdded(Friend i\_Friend)
2. הגדרנו כי המחלקה FilteredTopLikesFriends תירש מהמחלקה FiltrerBase ותממש את המטודה האבסטרקטית של האבא:   
   protected override bool ShouldBeAdded(Friend i\_Friend)

בפרוייקט הUI במחלקה FormFacebookWindow –

במטודה fetchTopLikes יצרנו אובייקט חדש של המחלקה FilteredTopLikesFriends המוחזק באופן פולימורפי ע"י אובייקט מטיפוס האבא   
 (FiltrerBase) אשר עליו הפעלנו את מטודת הפילטור.

* **Sequence Diagram**
* **Class Diagram**