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

* **Album Scanner:**
  + מאפשר למשתמש לבחור אלבום שברשותו ולסרוק אותו על מנת להציג את תמונותיו. בנוסף, יוצר רשימה של כל האנשים אשר מתויגים בתוך האלבום, עם אפשרות סינון של תמונות לפי אנשים מתויגים.
  + אפשרות סימון ל - לייקים לתמונות שנבחרו באלבום (ע"י לחיצה עליהן), ואז בלחיצת כפתור מתבצע לייקים לכל התמונות שסומנו. ניתן לראות את מספר התמונות שנבחרו מתחת לכפתור ה – לייק.
* **Blast From The Past:** מאפשר למשתמש לבחור שנה כלשהי שבו הוא פרסם פוסטים על הקיר שלו ולקבל פוסט אקראי מאותה שנה, או את הפוסט עם הכי הרבה לייקים.

### תבנית מס' 1 – Command

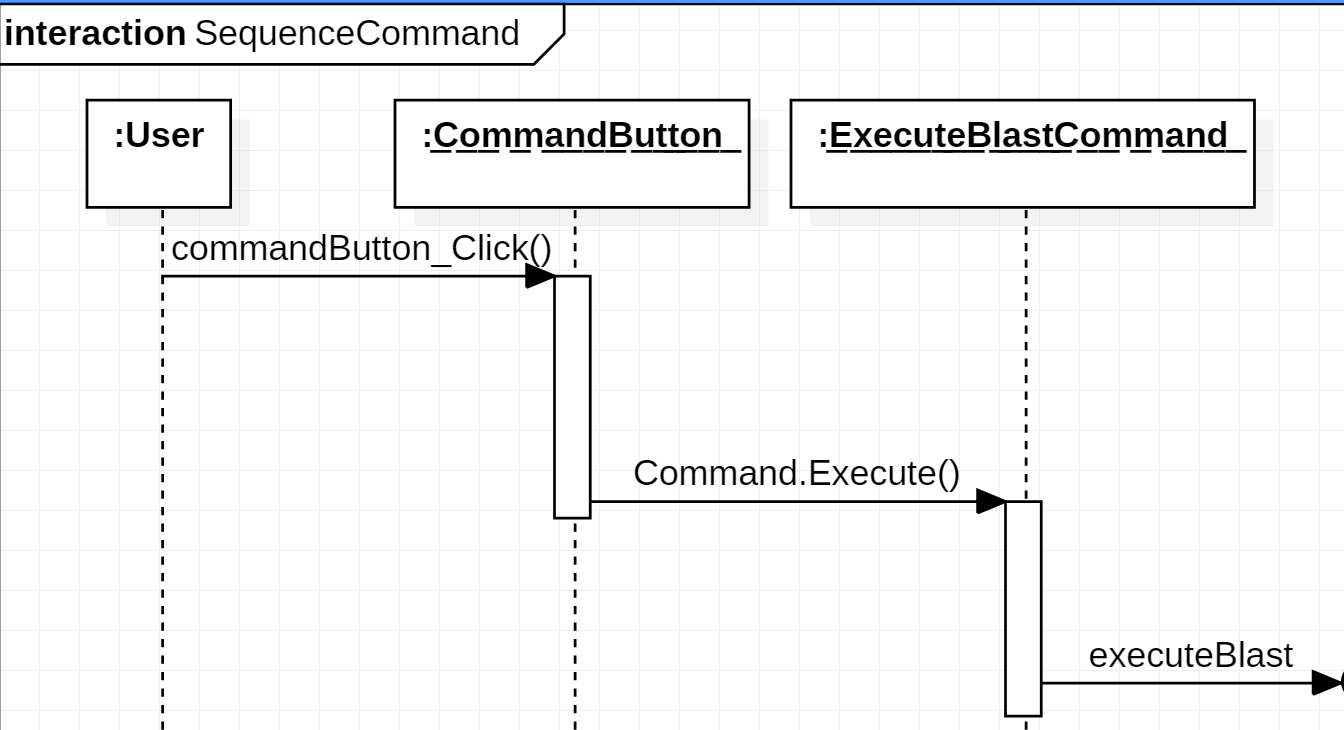
* סיבת הבחירה / שימוש בתבנית:

בשימוש הכפתור הישן היינו צריכים לממש את הקריאה לפונקציה הרלוונטית בשכבת המשתמש.  
באופן תמידי, אנו שואפים למינימום לוגיקה בשכבת המשתמש, לכן יצרנו כפתור שמחזיק ICommand, שזו פעולה השייכת לשכבה הלוגית.

* אופן המימוש:

מימוש ממשק ICommand ע"י ExecuteBlastCommand שבתוך מחלקת BlastFromThePast כך שבתוכו יהיה את פרמטרי השנה שנבחרה ורנדומי או הכי הרבה לייקים.  
יצירת CommandButtom אשר מכיל בתוכו אובייקט ICommand אשר BlastFromThePastForm יחזיק וכך שבכל פעם שנלחץ, יתרחשו תהליכים בלוגיקת הכפתור שתגרום לפקודה להתבצע (ע"י Execute()).

* Sequence Diagram



\* בשאר התבניות נציג את המשך ה – Sequence לאחר ביצוע executeBlast.

* Class Diagram

A close up of text on a black background

Description automatically generated

### תבנית מס' 2 – Strategy

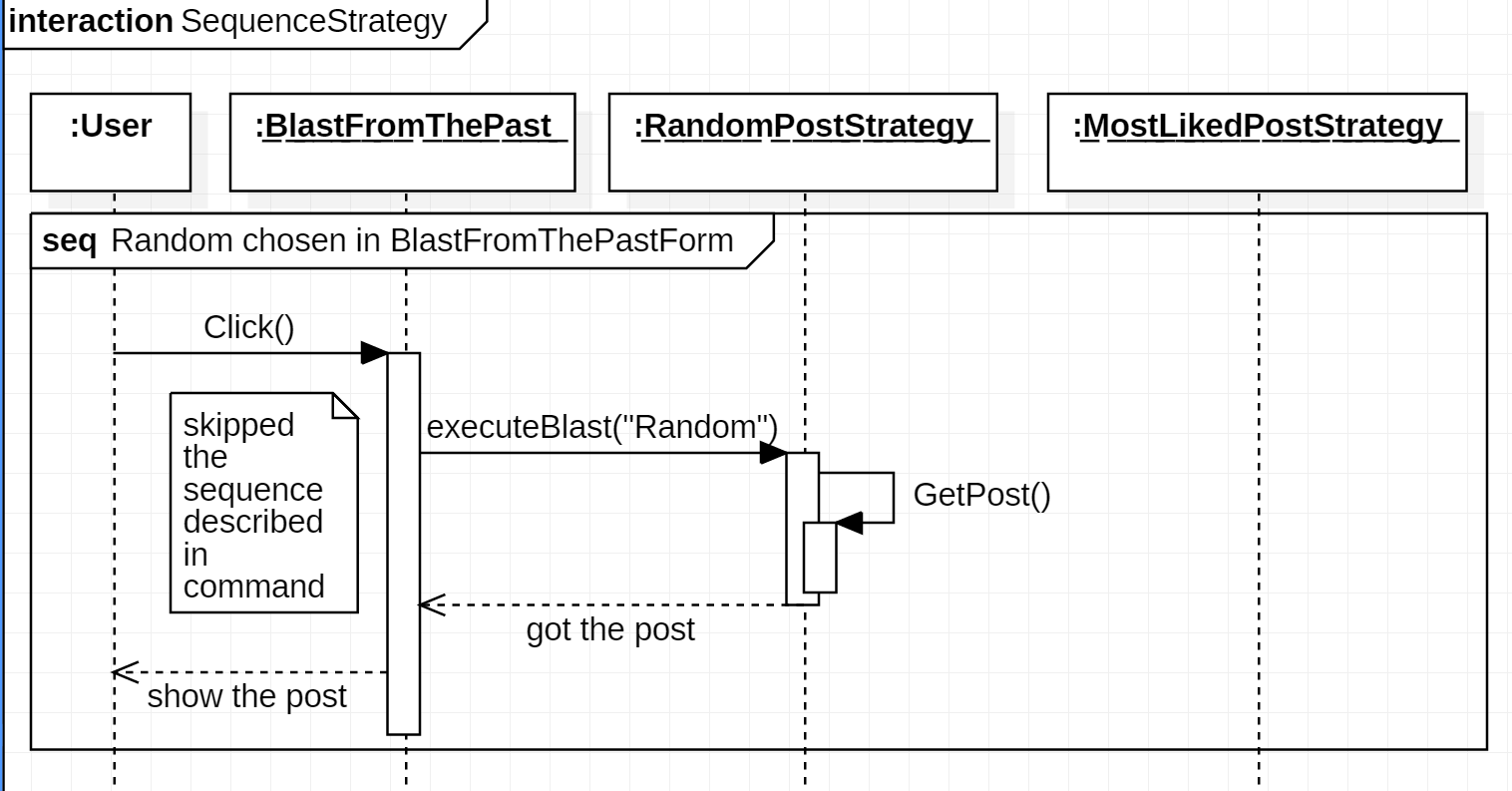
* סיבת הבחירה / שימוש בתבנית:

בפיצ'ר Blast From The Past ישנן שתי אפשרויות להציג פוסט: רנדומי או הכי הרבה לייקים. מכיוון שיש הרבה במשותף במימוש כל אחד מלבד הסוג שיטה, בחרנו להשתמש בתבנית זו, שתהיה סוג בחירת הפוסטים. איחוד המימושים הללו גורם למניעת שכפול קוד. בנוסף, אם בעתיד נרצה להוסיף עוד אלגוריתם מיצוע פוסט, ניתן לעשות זאת בקלות רבה בזכות מימוש תבנית זו.

* אופן המימוש:

יצרנו תחילה ממשק IPostStrategy אשר מכילה מתודה שמשתנה בין כל מימוש. ישנן שתי מחלקות (מקוננות בתוך BlastFromThePast) אשר מממשות זאת: RandomPostStrategy ו MostLikedPostStrategy כל אחת מממשת זאת לפי סוג הפוסט שאנו רוצים. השימוש נמצא במתודה executeBlast שבמחקלה BlastFromThePast.

* Sequence Diagram



* Class Diagram

A screenshot of a cell phone

Description automatically generated

### תבנית מס' 3 – Observer

* סיבת הבחירה / שימוש בתבנית:

שימוש בObserver מונע Tight Coupling בין שכבת המשתמש לשכבת הלוגיקה.   
אנו בחרנו שלכפתור, יהיה event ולו יהיו מאזינים. ברגע שהוא מסיים לבצע פעולה המשוייכת לכפתור (מסוג ICommandה – מתואר בPattern Command), הוא מודיע למאזינים על כך.   
המאזינים שלנו הם הForms של הפיצ'רים אשר נמצאים בשכבת המשתמש. הם נרשמים לפעולת הכפתור ע"י העברת Delegate (Action) ל – Event (כלומר, ה – Event מכיר רק את המתודה שמעוברת ולא את ה – Form), וברגע שביצוע הפעולה מסתיימת, מופצת הודעה לForms השונים המאזינים לפעולה וה – Forms מעדכנים את עצמם בהתאם (ממשק המשתמש מתעדכן).

* אופן המימוש:

בשביל לממש את תבנית ה Observer, אנו נגדיר:   
 Publisher (Subject)- CommandButton אשר נקרא commandButtonExecuteBlast בפיצ'ר BlastFromThePast.  
Subscriber (Observer) – BlastFromThePastForm עבור הפיצ'ר BlastFromThePast.  
במחלקת הכפתור CommandButton הוספנו event אשר נקרא CommandFinished אשר מופעל בסיום פעולת הExecute של הכפתור ומודיע לBlastFromThePastForm (ה – Subscriber) להציג את תוכן הפוסט ומידע נוסף על הפוסט למשתמש. ההודעה ל – לBlastFromThePastForm ע"י כך שה – CommandFinished מפעיל את המתודה שBlastFromThePastForm רשם ל – CommandFinished.

* Sequence Diagram

A screenshot of a cell phone

Description automatically generated

* Class Diagram

A screenshot of a social media post

Description automatically generated

\* בדיאגרמות ובאופן המימוש מתואר הObserver על הפיצ'ר BlastFromThePast אך ישנו מימוש דומה של הObserver על הפיצ'ר AlbumScanner.