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

* פיצ'ר ראשון: מציג את האלבומים של היוזר שהתחבר, ומתוך אחד האלבומים ניתן לבחור תמונה, לערוך בצייר ולפרסם אותה באלבום.
* פיצ'ר שני: מאפשר להפעיל את מצלמת האינטרנט של היוזר שהתחבר ולהעלות תמונה ממצלמת האינטרנט ל- timeline.
* פיצ'ר שלישי: מאפשר ליוזר שהתחבר להודיע על "מצב רגשי" שמח, עצוב, כועס ולעדכן זאת ב- facebook.
* פיצ'ר רביעי: מאפשר למשתמש לסנן את ה- news feed לפי התיאור או לפי הלינק.

### תבנית מס' 1 – [Decorator]

* סיבת הבחירה / שימוש בתבנית:

השתמשנו בתבנית הנ"ל בפיצ'ר של הצייר, בפיצ'ר זה יש למשתמש אפשרות לבחור תמונה מתוך אחד מאלבומי התמונות הקיימים בפייסבוק שלו. באמצעות שימוש בתבנית הוספנו למשתמש את האופציה להוסיף מסגרת לתמונה וכמו כן גם לבבות וכל זה באמצעות לחיצת כפתור שנמצא ב- tool strip של הצייר.

* אופן המימוש:

בפרוייקט תחת תיקיית Decorator מופיעים: ממשק 1 (IPhotoDecorator), מחלקה אבסטרקטית 1 (PhotoDecorator) ו 3 מחלקות קונקרטיות (BasicPhoto שהיא הcoreDecorator, RedFramePhoto, HeartsDecoration).

הממשק בעצם מגדיר את הפעולה שמעצבת את התמונה ומחזירה אותה, המחלקה הקונקרטית BasicPhoto שהיא בעצם ה- coreDecorator מממשת את הממשק ומחזירה תמונה "בסיסית"- כלומר ללא העיצובים האופציונליים. המחלקה האבסטרקטית PhotoDecorator גם מממשת את הממשק ומבצעת את העיצוב בהתאם לבחירת המשתמש, כלומר בהתאם לכפתור עליו לחץ המשתמש. 2 המחלקות הקונקרטיות הנוספות HeartsDecoration ו- RedFramePhoto יורשות מהמחלקה האבסטרקטית PhotoDecorator ובהן מתבצעת דריסה (override) למתודה שמעצבת את התמונה.

**Participants:**

Class FormMain is the **Client**

Interface IPhotoDecorator is the **IComponent**

Class BasicPhoto is the **SomeCoreComponent**

Class PhotoDecorator is the **Decorator**

Classes HeartsDecoration, RedFramePhoto are the **ConcreteDecorator**

* Sequence Diagram



* Class Diagram

### תבנית מס' 2 – [Strategy]

* סיבת הבחירה / שימוש בתבנית:

ישנו פיצ'ר קיים במערכת שנקרא PostEmotionalStatus בו המשתמש בוחר את מצבו הרגשי ובהתאם לכך לוחץ על כפתור ובעת שהוא עושה זאת עולה לפייסבוק סטטוס בהתאם למצב הרגשי בו נמצא המשתמש, כמו כן גם תמונה ושיר תואם ב- Youtube.

* אופן המימוש:

בפרוייקט תחת תיקיית Strategy: הגדרנו ממשק IEmotionStrategy בממשק זה ישנם הצהרות על מתודות שמבצעות פתיחת שיר ב- Youtube , העלאת תמונה לפייסבוק ופרסום סטטוס בפייסבוק.

את הממשק מממשים שלושת המחלקות : AngryEmotionStrategy , HappyEmotionStrategy , SadEmotionStrategy וכל אחת מממשת את המתודות בממשק בהתאם ל"מצב הרגשי" של המשתמש.

כמו כן, ישנה מחלקה שנקראת EmotionManager והיא מחזיקה רפרנס לממשק IEmotionStrategy ובעצם מנהלת את המצב הרגשי בהתאם לאסטרטגיה (EmotionManager בעצם אחראית לפרסום הסטטוסים\ תמונות) .

**Participants:**

Class FormMain is the **Client**

Interface IEmotionStrategy is the **Strategy**

Classes AngryEmotionStrategy, HappyEmotionStrategy, SadEmotionStrategy are the **ConcreteStrategy**

Class EmotionManager is the **Context**

* Sequence Diagram



* Class Diagram



### תבנית מס' 3 – [Template Method]

* סיבת הבחירה / שימוש בתבנית:

הוספנו למערכת פיצ'ר חדש באמצעותו ניתן לסנן סטטוסים בפייסבוק לפי לינקים שמשתמשים \ חברים מפרסמים או לפי סטטוסים. המשתמש לוחץ על RadioButton ובוחר כיצד ברצונו לסנן ע"י בחירה של אחת מ-2 האופציות, ואלו יופיעו לו ב- ListBox.

* אופן המימוש:

בפרוייקט תחת תיקיית Template Method: הגדרנו מחלקה אבסטרקטית FeedSorter שמחזיקה מתודה אבסטרקטית SortFeedOnRequest שמקבלת Post ומחזירה מחרוזת, המחלקות שירשו מ FeedSorter יממשו אותה אצלן (מתודה זו מהווה Injection Point- סינון\ מיון לפי טקסט או לינקים או אם בעתיד ירצו מיון אחר זה יהיה אפשרי רק אם הערך שיוחזר הוא מחרוזת). כמו כן FeedSorter מחזיקה מתודה שמחזירה List של String-ים שהם אלו שיופיעו ב-ListBox לפי בחירת הסינון של המשתמש.

המחלקות הקונקרטיות SortByLink ו- SortByDescription דורסות את המתודה SortFeedOnRequest (override) וקובעות את אופי הסינון כלומר מה שייתווסף לרשימה.

**Participants:**

Class FormMain is the **Client**

Class FeedSorter is the **Mechanism**

Classes SortByDescription, SortByLin are the **ConcreteClass (which implement the Template** **Method)**

MethodSortFeedOnRequest is the **Template** **Method**

* Sequence Diagram



*  Class Diagram