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

* **גרף להצגת מספר הפוסטים בשנה האחרונה:**

בפרטי "אודות המשתמש", מופיע גרף המייצג למשתמש מידע על מספר הפוסטים שהוא העלה בכל חודש במשך השנה הנוכחית.

הקוד של הפיצ'ר נמצא ב-LogicEngineManager וב- FormDesktopFacebook.

* **יצירת אירוע לחברים לפי גילאים:**

המשתמש יכול ליצור אירוע (לפי תאריך ושעה) ולבחור חברים שישתתפו באירוע זה, אשר יהיו מסוננים לפי גילאים שהמשתמש מזין.

הקוד של הפיצ'ר נמצא ב-LogicEngineManager.

### תבנית מס' 1 – Strategy

* **סיבת הבחירה / שימוש בתבנית:**

רצינו לאפשר למשתמשים בתוכנית שלנו למיין את רשימת החברים, האלבומים, הדפים והקבוצות לפי סדר עולה **וגם** לפי סדר יורד של השמות מבלי לשכפל את הקוד שאחראי על המיון.

לכן בחרנו להשתמש בתבנית Strategy שמאפשרת לנו לממש את מנגנון המיון ומאפשרת לנו בעתיד לשנות את ההתנהגות שלו בלי לשכפל\לשנות את הקוד האחראי עליו.

* **אופן המימוש:**

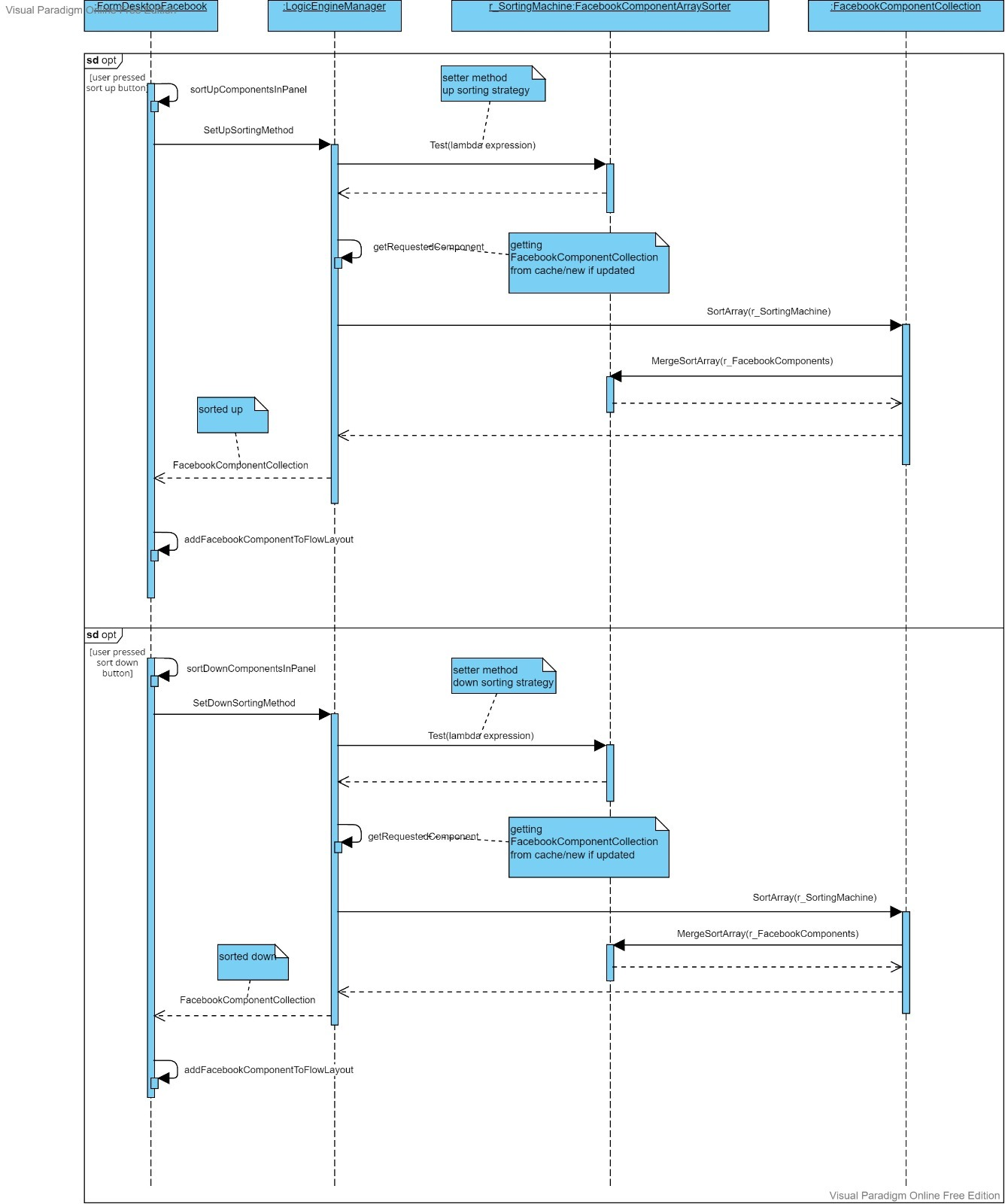
מימשנו מחלקה בשם **FacebookComponentsArraySorter** שאחראית על כל המיונים שלנו בתוכנית, ובתוכה מימשנו את השיטה **MergeSortArray** שמקבלת מערך של IFacebookComponents (שמהווה בסיס פולימורפי לחברים, אלבומים, דפים וקבוצות) וממיינת אותו בעזרת אלגוריתם המיון MergeSort**.**

זיהינו את הקטע האחראי על על ההשוואה בין האיברים בתוך האלגורתים, שאותו החלפנו בשיטה שמקבלת שני איברים מסוג IFacebookComponents ומחזירה true/false האם להחליף בין האיברים או לא. שאת השיטה הזאת החזקנו כשדה של המחלקה **FacebookComponentArraySorter** והגדרנו setter עליה (בעזרת Automatic property) כדי שנוכל בעתיד לשנות את ההתנהגות של מנגנון המיון.

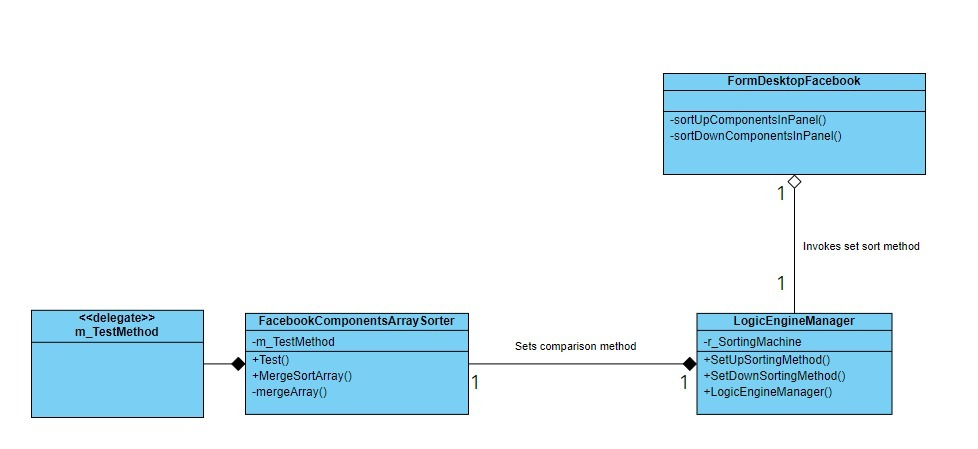
החלטנו להשתמש ב method strategy משום שהייתה לנו שיטה אחת בלבד שהלוגיקה שלה פשוטה.

(המקבילה של כל מחלקה משלנו בתבנית מצויין מתחת ל class diagram שמצורף בהמשך).

* **Sequence Diagram**



* **Class Diagram**



המקבילה למחלקה **Context** בתבנית היא המחלקה **FacebookComponentArraySorter** שהכילה בתוכה את מימוש המנגנון.

את הממשק IStrategy והמחלקות שממשות אותו בתבנית **הוחלפו** ב delegate שהוגדר כשדה במחלקה **FacebookComponentArraySorter** (עשינו זאת כיוון שישנה שיטה אחת בממשק שהלוגיקה שלה פשוטה) כלומר נעשה שימוש ב method strategy.

### תבנית מס' 2 – Observer

* **סיבת הבחירה / שימוש בתבנית:**

בתוכנית שלנו מימשנו מחלקה בשם **ComplexFacebookComponent** אשר הכילה בתוכה כשדות label וגם picturebox (שממוקמים בתוך panel שגם הוא בעצמו שדה של המחלקה) שתפקידם היה להציג למשתמש את השם וגם את התמונה של IFacebookComponent(שמהווה בסיס פולימורפי לחברים, אלבומים, דפים וקבוצות).

רצינו שה **ComplexFacebookComponent** יוכל לעדכן את ה **FormDesktopFacebook** שנלחץ עליו בלי שיהיה תלוי\קשור ב\אל הפורם (וכך נהפוך את המחלקה ל reusable) בכדי שנוכל להציג יותר מידע למשתמש על ה IFacebookComponent.

ולכן החלטנו להשתמש ב **Observer** שנותן מענה לדרישות שלנו.

* **אופן המימוש:**

בתוך המחלקה **ComplexFacebookComponent** הוספנו את השדה Action<IFacebookComponent> **WasClicked** שהגדרנו אותו להיות public event בכדי שיהיה חשוף לפעולת "=+" שמאפשרת הרשמה לאירוע על ידי שליחת שיטה שהנרשם ירצה שתופעל בהתרחשות האירוע.

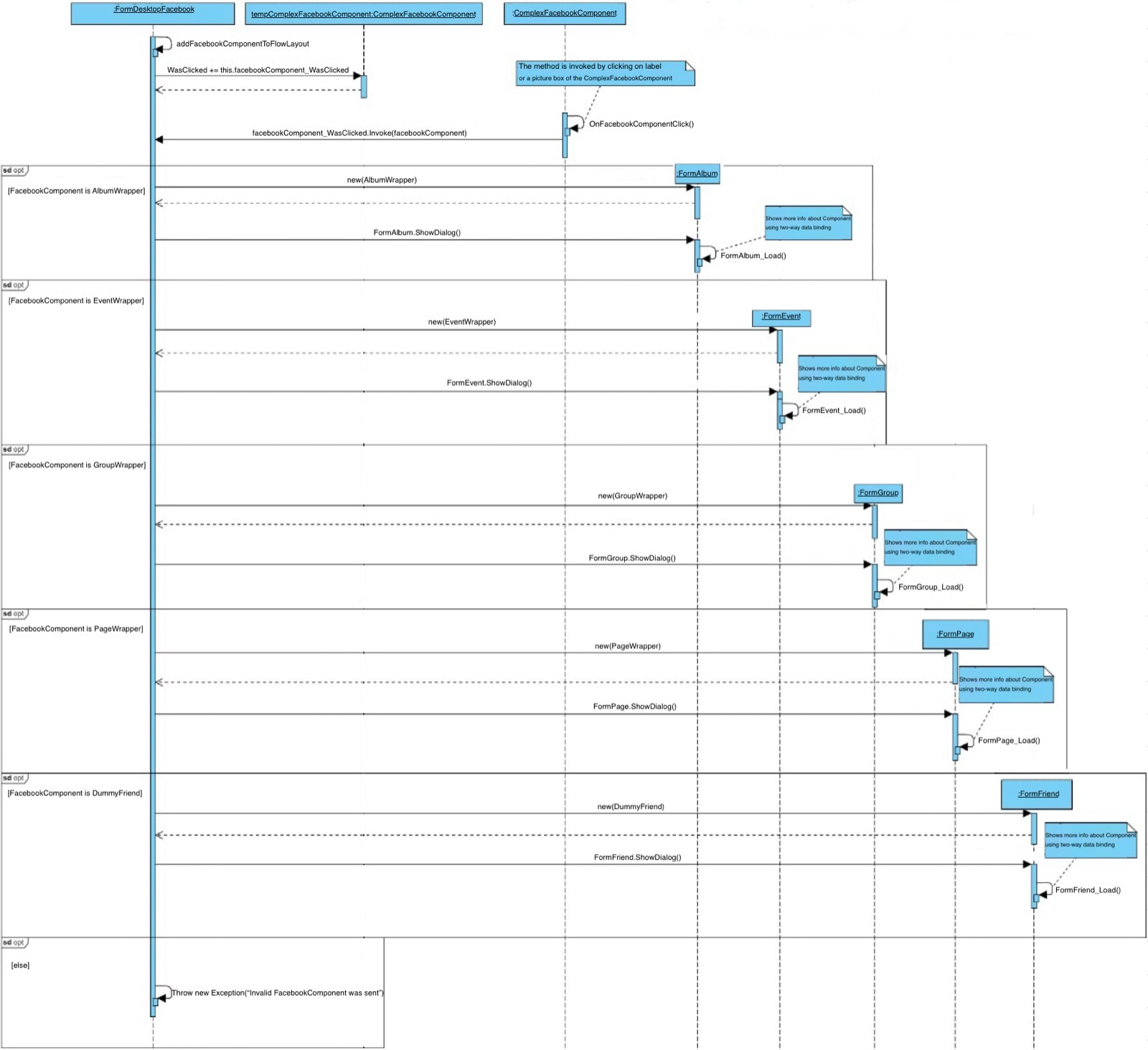
מימשנו גם שיטה בשם **OnFacebookComponentClick** (השיטה מופעלת כתגובה לאירוע הלחיצה על ה label או על ה picturebox שמוחזקים כשדות של המחלקה ומוצגים למשתמש בתוך הפורם)שבתוכה עוברים על כל השיטות של הנרשמים, מפעילים אותם ושולחים להם משתנה מטיפוס IFacebookComponent שהוא בעצמו גם שדה של המחלקה.

מצד הנרשם שבמקרה שלנו זה המחלקה **FormDesktopFacebook** מימשנו שיטה בשם **facebookComponent\_WasClicked** שמקבלת משתנה מטיפוס IFacebookComponentעושה לו down casting לאחת המחלקות GroupWrapper, PageWrapper, EventWrapper, AlbumWrapper או DummyFriend ואז מציגה את כל המידע בפורם חדש שהיא מייצרת תוך שימוש ב two-way data binding.

המחלקה **FormDesktopFacebook** נרשמת לאירוע הלחיצה בעזרת הפעולה "=+" שהוגדרה על השדה **WasClicked** בתוך המחלקה **ComplexFacebookComponent** תוך כדי שהיא מייצרת את המופעים של המחלקה ומציגה אותם למשתמש. וזה מתרחש בתוך השיטה **addFacebookComponentToFlowLayout.**

(המקבילה של כל מחלקה משלנו בתבנית מצויין מתחת ל class diagram שמצורף בהמשך).

* **Sequence Diagram**



* **Class Diagram**

Graphical user interface, diagram, text, application

Description automatically generated

ה **Observable** אצלנו בקוד היינו המחלקה **ComplexFacebookComponent**.

ה **Observer** אצלנו בקוד היינו המחלקה **FormDesktopFace**.

החלטנו לוותר על הממשק IObserver ומבמקומו להגדיר Action<IFacebookComponent> בתוך המחלקה complexFacebookComponent. (עברנו ל method oriented Programming)

### תבנית מס' 3 – Iterator

* **סיבת הבחירה / שימוש בתבנית:**

בתוכנית שלנו יש אוסף של IFacebookComponent (שמהווה בסיס פולימורפי לחברים, אלבומים, דפים וקבוצות).

רצינו לאפשר למשתמשים לעבור על אוסף זה מבלי להתייחס לסוג מבנה הנתונים בו הם מוחזקים, מה שיאפשר לנו בעתיד לשנות את מבנה הנתונים (לפי הצורך) מבלי לעדכן את המשתמש משום שזה לא ישפיע על האופן שבו הוא סורק.

החלטנו להשתמש בתבנית iterator מכיוון שתבנית זו עוזרת לנו להסתיר את מבנה הנתונים מהמשתמש ומספקת למשתמש שיטות לסריקת הקולקציה (שאינן תלויות במבנה הנתונים), דבר זה שומר על encapsulation ופילטור מידע.

* אופן המימוש:

מימשנו מחלקה בשם **FacebookComponentCollection**, שבתוכה החזקנו את מערך של IFacebookComponent שאותו המשתמש סורק (מבלי לדעת את סוג מבנה הנתונים) שאותו המנוע שלנו ממלא בעזרת שתי השיטות **add** וגם **addMultiple** שאותן מימשנו כשיטות של המחלקה

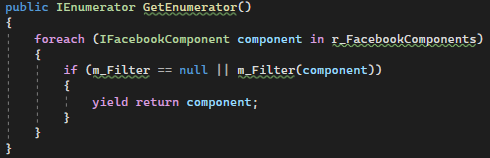
המחלקה **FacebookComponentCollection** מימשה את הממשק **IEnumerable** שאת השיטה **GetEnumerator** שמחזירה IEnumerator מימשנו בעזרת הפיצ"ר **yield return**.

**תוספת**

1. **Filtered iterator:**

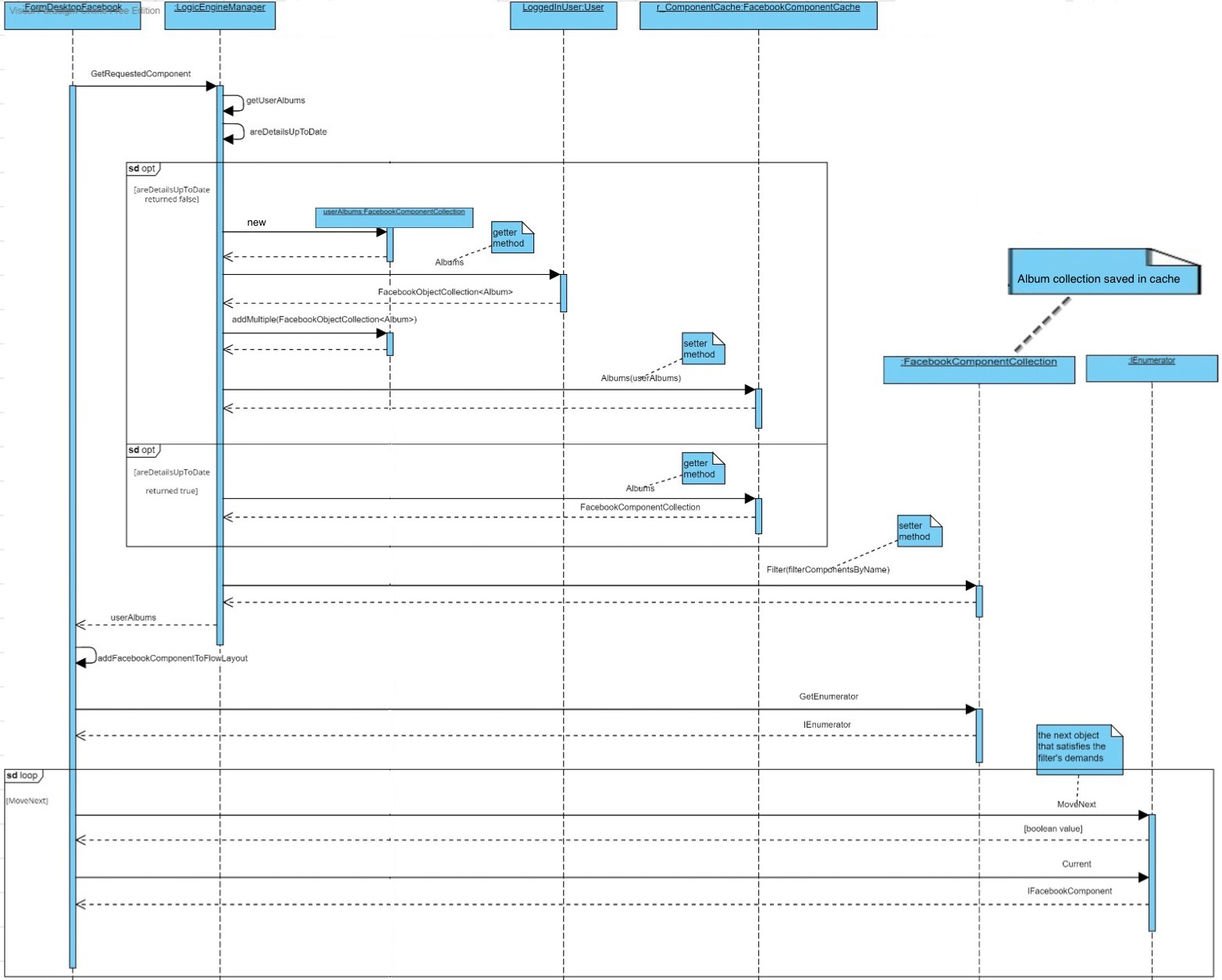
מימשנו את האיטרטור שלנו באופן שבו אפשרנו למשתמש לשלוח למחלקה פילטר שבעזרתו יוכל לפלטר את האיברים בתוך מבנה הנתונים.

את האפשרות הזו מימשנו על ידי עטיפת ה **yield return** בקריאה ל delegate שמקבל **IFacebookComponent** ומחזיר **true** אם המשתמש ירצה לחשוף את האיבר ו **false** אחרת, שאת ה delegate החזקנו כשדה של המחלקה ומימשנו עבורו **setter** בכדי לאפשר למשתמש לשנות את שיטת הפילטור (נעשה שימוש בתבנית Strategy).

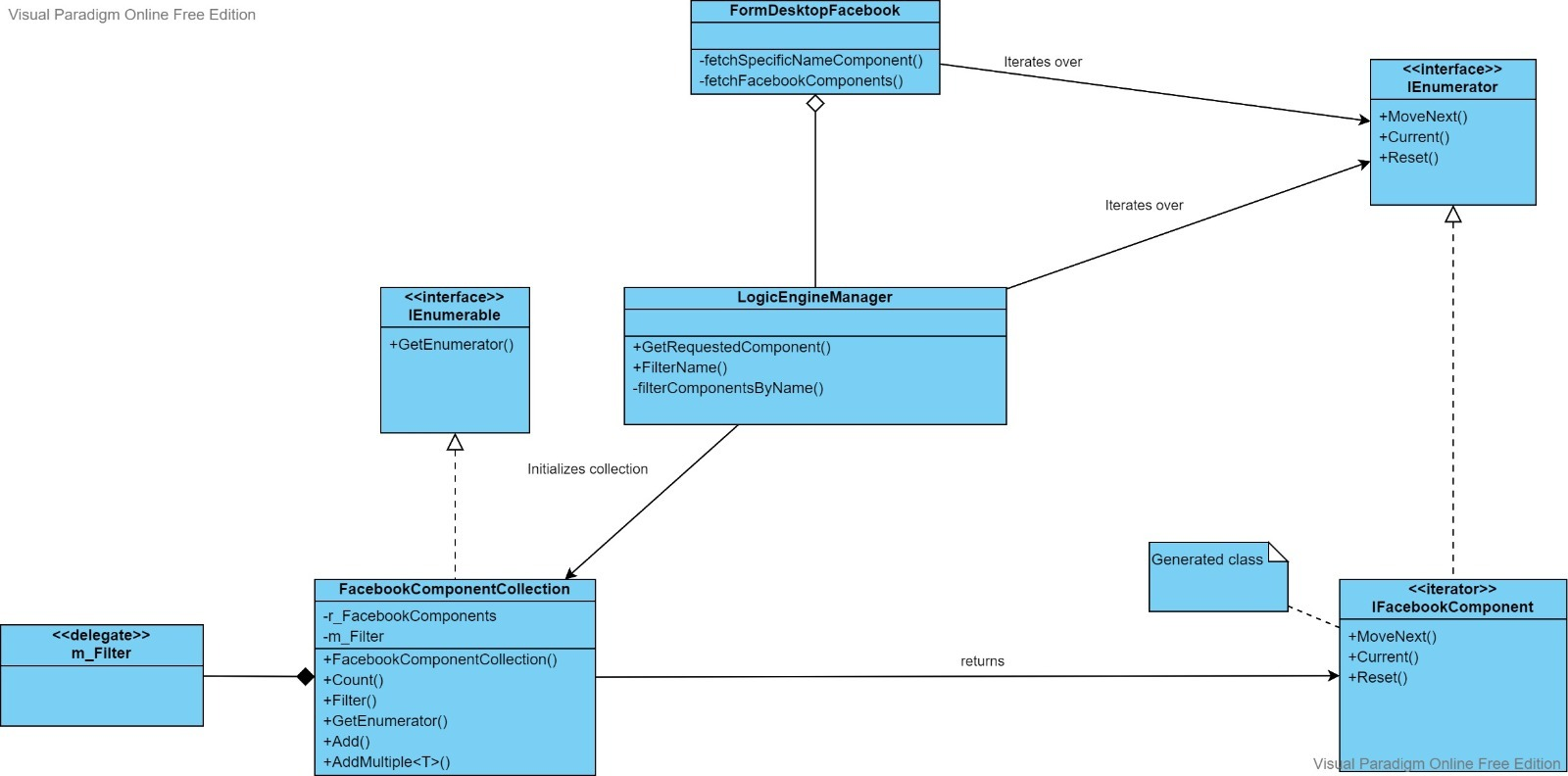


(המקבילה של כל מחלקה משלנו בתבנית מצויין מתחת ל class diagram שמצורף בהמשך).

* **Sequence Diagram**



* **Class Diagram**



ה **ConcreteAggregate** אצלנו בקוד היינו המחלקה **FacebookComponentCollection.**

ה **ConcreteIterator** אצלנו בקוד היינו **generator** **class** שקיבלנו בעזרת הפיצ"ר yield return.

ה **Clients** שמשתמשים באיטרטור וסורקים את האוסף שהאיטרטור מספק הינם המחלקות **LogicEngineManager** וגם **FormDesktopFacebook.**