### 

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

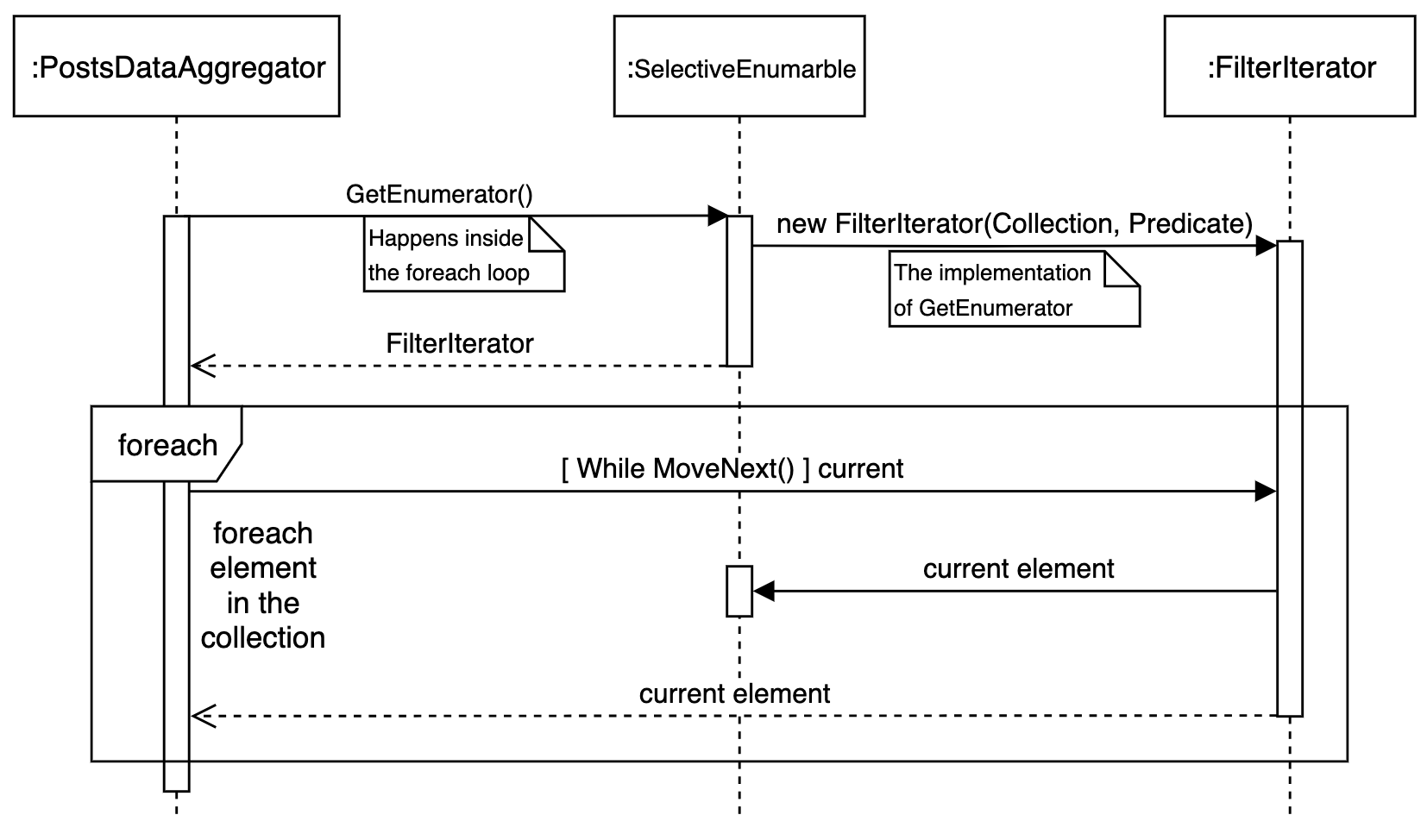
* ״Who likes me most?״ הינו פיצ׳ר המאפשר למשתמש לקבל רשימה המכילה את כל המשתמשים אשר אי פעם סימנו ״אהבתי״ (Like) לאחד או יותר מהפוסטים שפורסמו על ידו. בנוסף, הרשימה מסודרת לפי מספר הסימונים אותם עשה כל משתמש, ובכך מאפשרת למשתמש המבקש לראות מי הם המשתמשים אשר עוקבים אחר התוכן אותו הוא מפרסם ואוהבים אותו יותר מבין עוקביו.
* "Influence preservation grade per post" הינו פיצ׳ר המאפשר למשתמש לבחור פוסט ספציפי מבין כל הפוסטים אותם פרסם,ולקבל עליו ציון בין 0 ל-100 לאחר ניתוח השפעת אותו פוסט נבחר על קהל עוקביו של המשתמש. ככל שיותר משתמשים אשר עוקבים אחריו באופן קבוע סימנו Like לאותו פוסט, כך ציונו יהיה גבוה יותר בכך שעזר לשמר את השפעתו האינטרנטית של המשתמש על קהל עוקביו בכדי שימשיכו לעקוב אחר התוכן אותו הוא מפרסם.
* ״Influence expansion grade per post״ הינו פיצ׳ר משלים לנ״ל, המאפשר למשתמש לבחור פוסט ספציפי מבין כל הפוסטים אותם פרסם,ולקבל עליו ציון בין 0 ל-100 לאחר ניתוח השפעת אותו פוסט נבחר על קהל עוקביו של המשתמש. ככל שיותר משתמשים אשר אינם מסמנים Like באופן קבוע לפרסומים של המשתמש אכן סימנו Like לפוסט, ובפרט משתמשים אשר מעולם לא סימנו Like לאף פוסט של המשתמש, כך ציונו של הפוסט גבוה יותר, בכך שעזר להרחיב את השפעתו האינטרנטית של אותו משתמש ועזר לו להגיע לקהלים חדשים נוספים ובכך לאסוף משתמשים חדשים אשר יתווספו לקהל עוקביו.
* הוספת יכולת לעבוד ללא חיבור לאינטרנט - Offline Mode  
  על מנת לאפשר למשתמש יכולת עבודה גם כאשר איננו מחובר לאינטרנט, מימשנו בעזרת התבניות Adapter, Proxy, Facade אופציה של Remember Me אשר בעזרתה האפליקציה עושה שימוש ב-Cache ובך שומרת את נתוני המשתמש ונותן אפשרות להשתמש באפליקציה גם ב-Offline Mode.

### תבנית מס' 1 – Iterator

* סיבת הבחירה / שימוש בתבנית:  
  בחרנו להשתמש בתבנית Iterator מכיוון וישנו Legacy Component אשר איננו יודעים באיזה מבנה נתונים הוא עושה שימוש, אשר מספק נתונים על המשתמש, בהם האפליקציה עושה שימוש כחלק מהפיצ׳רים שהוספנו. את ה-Iterator מימשנו   
  כ-Filter Iterator אשר בעזרת שימוש ב-Predicate, מבצע סינון של האובייקטים הרלוונטיים לאפליקציה ומחזיר רק את אותם אובייקטים למחלקה העושה בהם שימוש.
* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram



* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם

### תבנית מס' 2 – Strategy

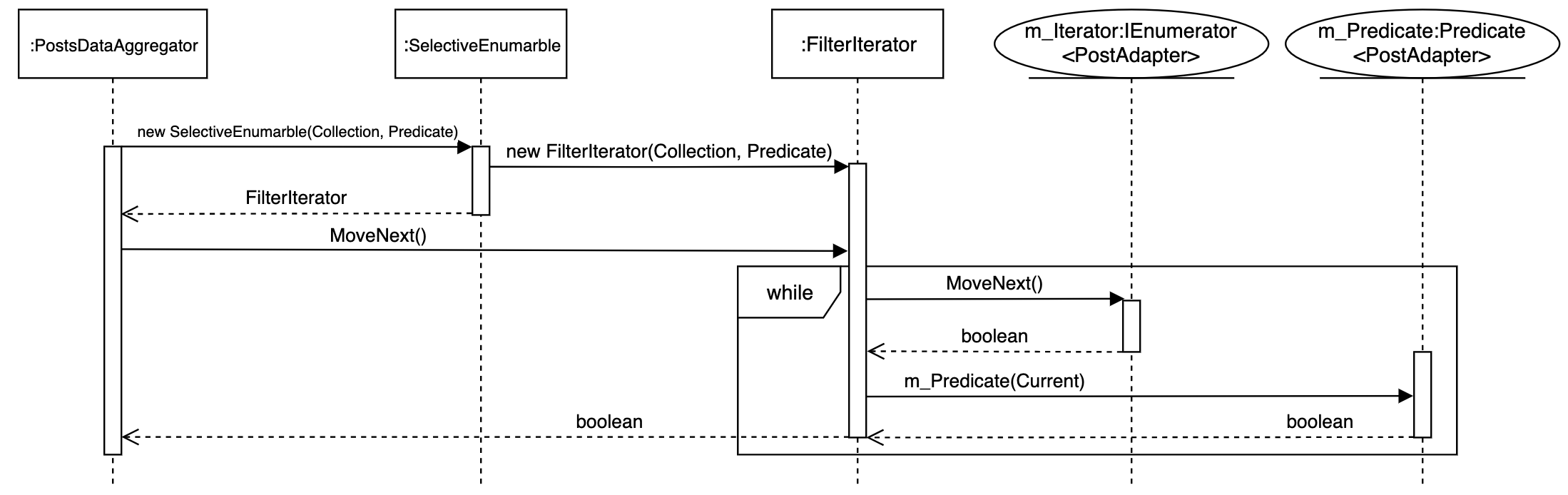
* סיבת הבחירה / שימוש בתבנית:

בחרנו להשתמש בתבנית Strategy על מנת לאפשר שימוש גנרי בתבנית ה-Iterator הנ״ל. בעזרת שימוש בתבנית Strategy, העברנו ל-Iterator את ה-Predicate הרלוונטי לאותו Iterator, ובכך בעצם יצרנו Filter Iterator אשר סינן את האובייקטים הרלוונטיים בלבד.

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram



* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם

### תבנית מס' 3 – [שם התבנית]

* סיבת הבחירה / שימוש בתבנית:

[תיאור הסיבה / הצורך בשימוש בתבנית במערכת שלכם]

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

* Sequence Diagram

[sequence diagram שמציגה את התהליך שקשור לתבנית והאינטראקציה בין המחלקות]

* Class Diagram

[class diagram שמתארת את המחלקות שלכם שמעורבות בתבנית (תיאור מלא שכולל Properties ו- Methods) והיחסים ביניהם (תאור מלא עבור כל יחס כפי שלמדנו בכיתה). עבור כל מחלקה שלכם, כיתבו מי המקבילה שלה (אם יש כזו) בתבנית (Pattern) שבחרתם