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

* [תיאור קצר של הפיצ'ר הראשון]
* [תיאור קצר של הפיצ'ר הראשון]

### תבנית מס' 1 – [Singleton]

* סיבת הבחירה / שימוש בתבנית:

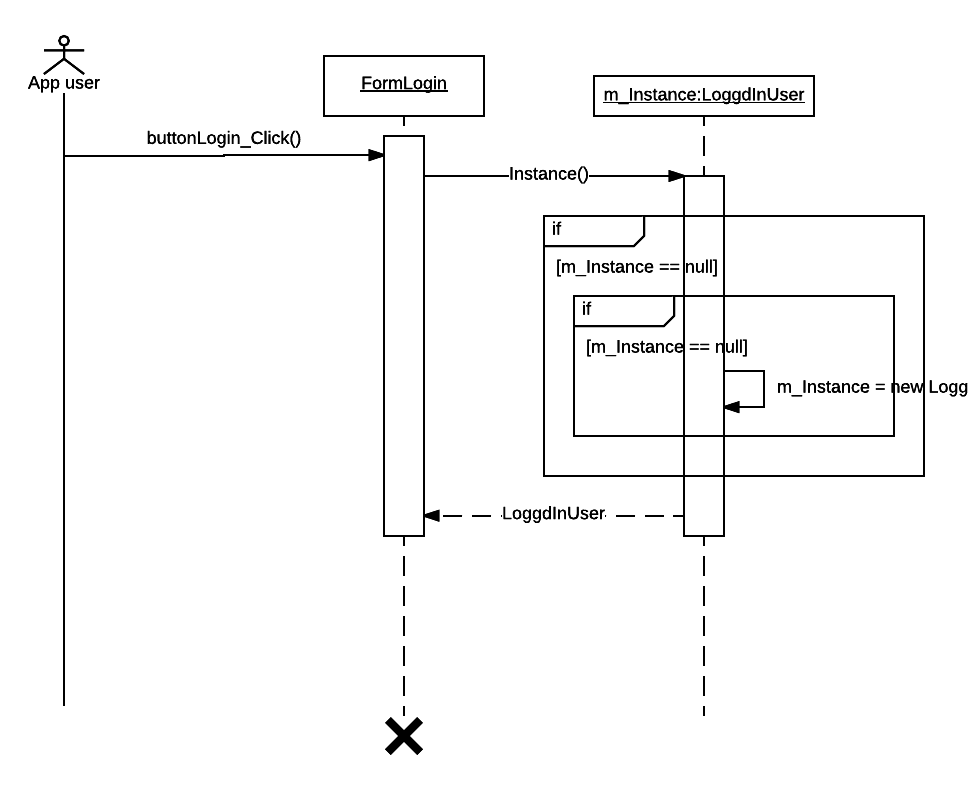
מכיוון שיש יותר מרכיב אחד שצריך להתמשמש במידע שאותה מחלקה מספקת ואין צורך ליצור יותר ממופע אחד של המחלקה וגם האפשרות לתת גישה גלובלית למחלקה רלוונטית במקרה זה. התבנית Singleton מתאימה ביותר.

* אופן המימוש:

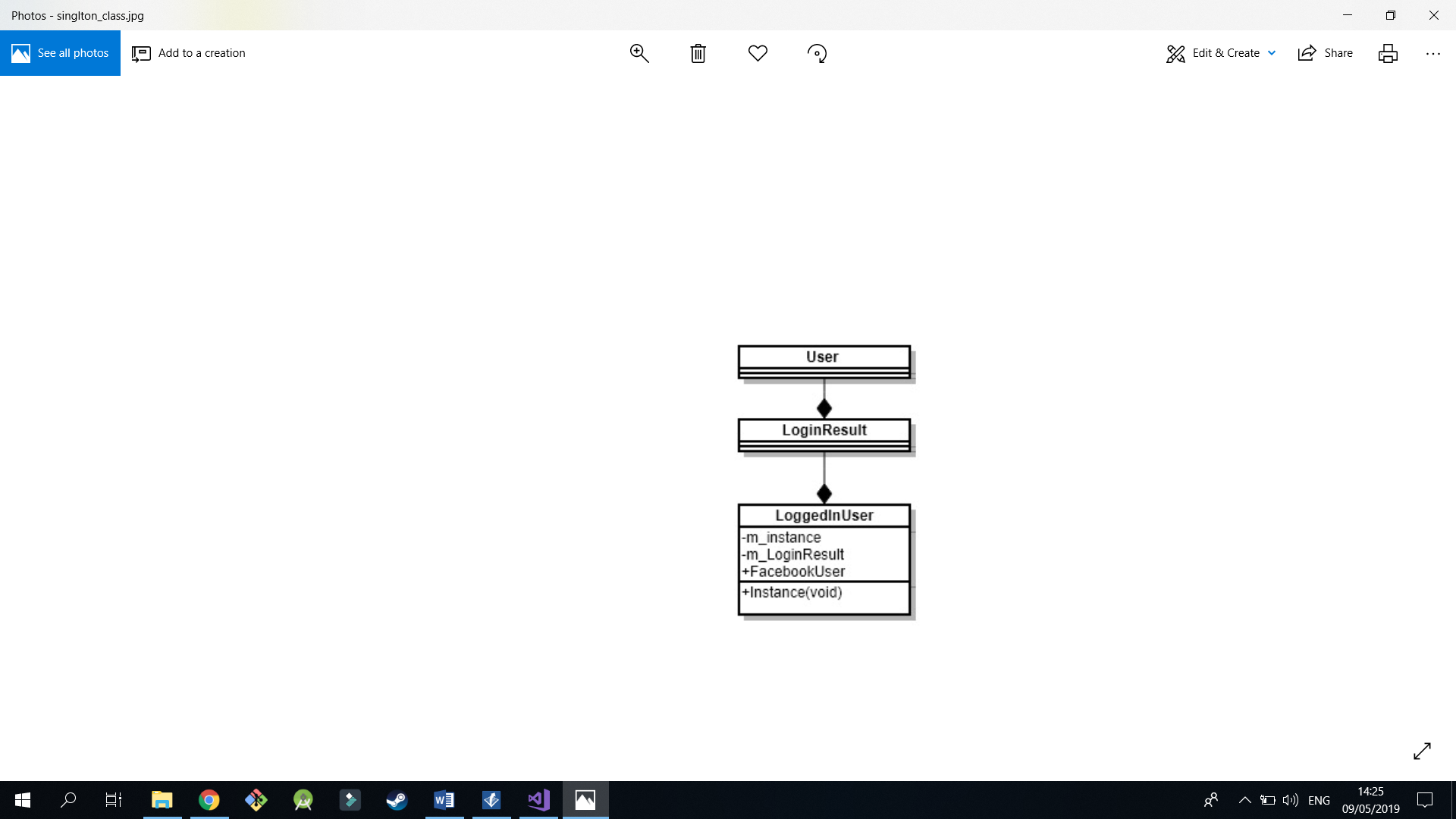
המחלקה LoggedInUser מוגדרת כ-Singlton והיא שומרת את המידע של היוזר המחובר למערכת. מימוש התבנית עומד בשני עקרונות:

1. מופע יחיד של המחלקה 2
2. גישה גלובלית. נתנו את האפשרות לגשת לאיבר במחלקה

* Sequence Diagram



* Class Diagram



### תבנית מס' 2 – [Factory Method]

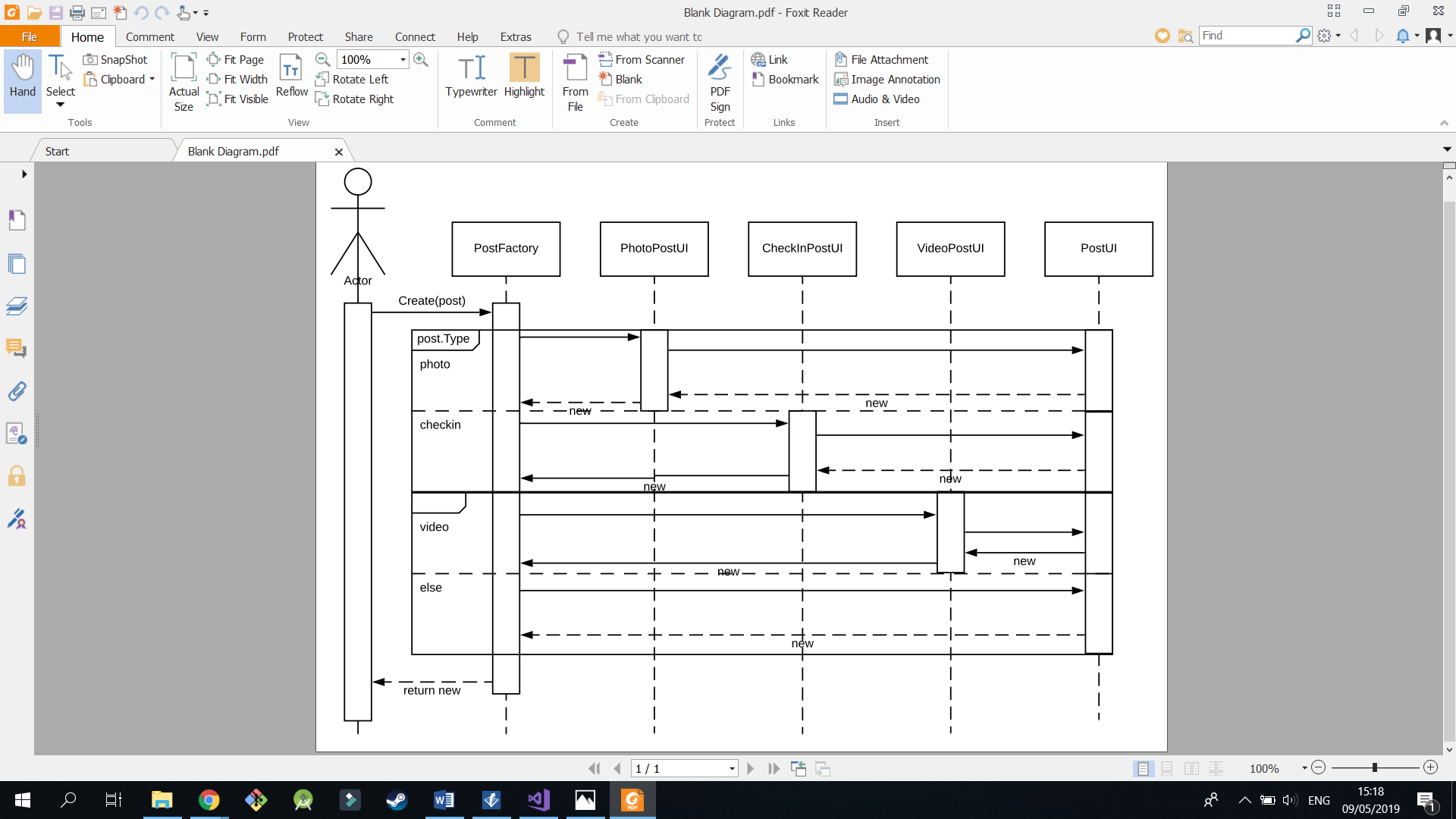
* סיבת הבחירה / שימוש בתבנית:

מכיוון שיש סוגים שונים של פוסטים שמצריכות פונקציות שונות בUI, והשוני נודע אך ורק בזמן ריצה,  
יצרנו מחלקות שונות לממשקים של הפוסטים. בכדי ליצור ממשק מתאים יצרנו Factory שמייצר פוסטים לפי הפוסט הנתון

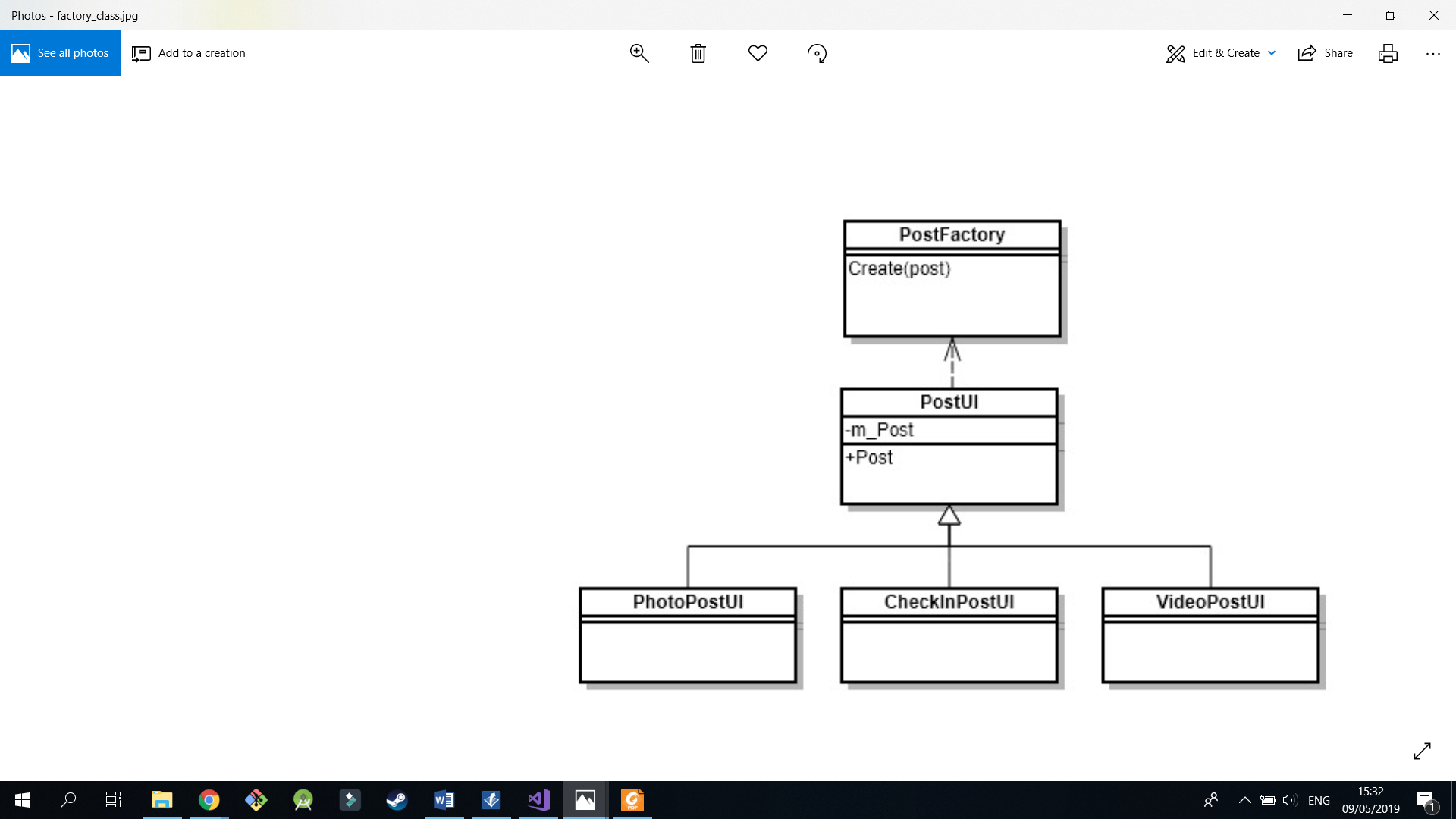
* אופן המימוש:

יצרנו מחלקה PostFactory שיוצרת ממשק בהתאם לסוג הפוסט שנשלח איליה

* Sequence Diagram



* Class Diagram



### 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