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

* בחירת אלבום להורדה מאלבומי המשתמש.
* תיאור מקיף על האלבום הנבחר + חיווי להתקדמות ההורדה.

### תבנית מס' 1 – Singleton

* סיבת הבחירה / שימוש בתבנית:

היה צורך לגשת לנתונים של משתמש ה-facebook שהתחבר ממספר מחלקות שונות –

ביניהן FacebookAlbumApp, AlbumSaver.

כיוון שמדובר במשתמש יחיד לא רצינו לאפשר יצירה של משתמשים נוספים,

בנוסף רצינו לאפשר גישה בלתי תלויה לנתוני המשתמש.

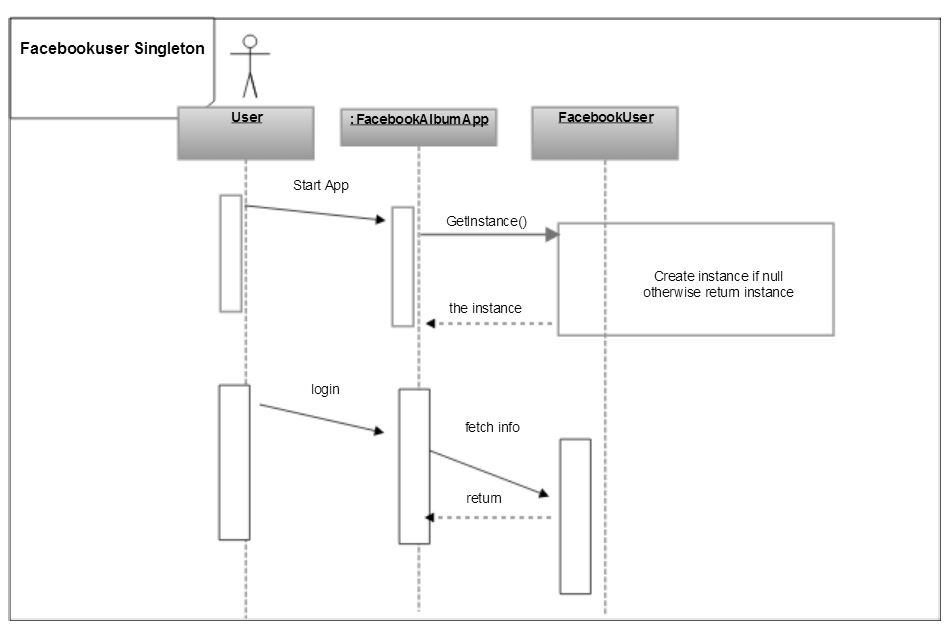
* אופן המימוש:

חילצנו את הלוגיקה שקשורה במשתמש, כגון התחברות והבאת אלבומים למחלקה FacebookUser, במחלקה הסתרנו את הבנאי בעזרת private.

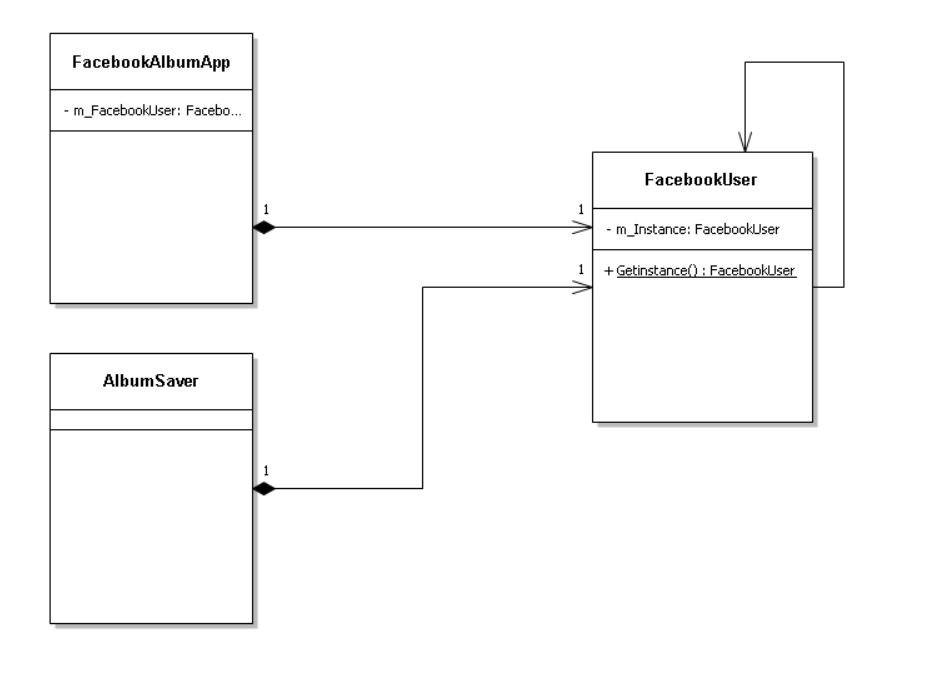
המחלקה FacebookUser מחזיקה משתנה Instance סטטי, והוא מאותחל בפעם הראשונה שוקראים ל-GetInstance.

האתחול אובטח לסביבות multi-threading בעזרת מנעול ובדיקה כפולה.

* Sequence Diagram



* Class Diagram



### תבנית מס' 2 – Facade

* סיבת הבחירה / שימוש בתבנית:

כשהמשתמש בוחר להוריד אלבום, ישנן מספר פעולות שמתבצעות ע"פ סדר מסוים ותלויות בשירותים שונים שקיימים במערכת.

החלטנו להסתיר את מימוש הפעולה המורכבת מהמשתמש (ה-UI במקרה שלנו).

מחלקת ה-Facade נקראת AlbumSaver, ומשתמשת בשירותים של FacebookUser, FolderBrowser ו-AlbumSaverProgress על מנת לבצע את הפעולה הנדרשת.

* אופן המימוש:

מחלקת ה-FacebookAlbumApp משתמשת בשירות AlbumSaver ע"י מתודה יחידה המקבלת פרמטר יחיד – את המזהה (ID) של אותו אלבום שברצוננו להוריד.

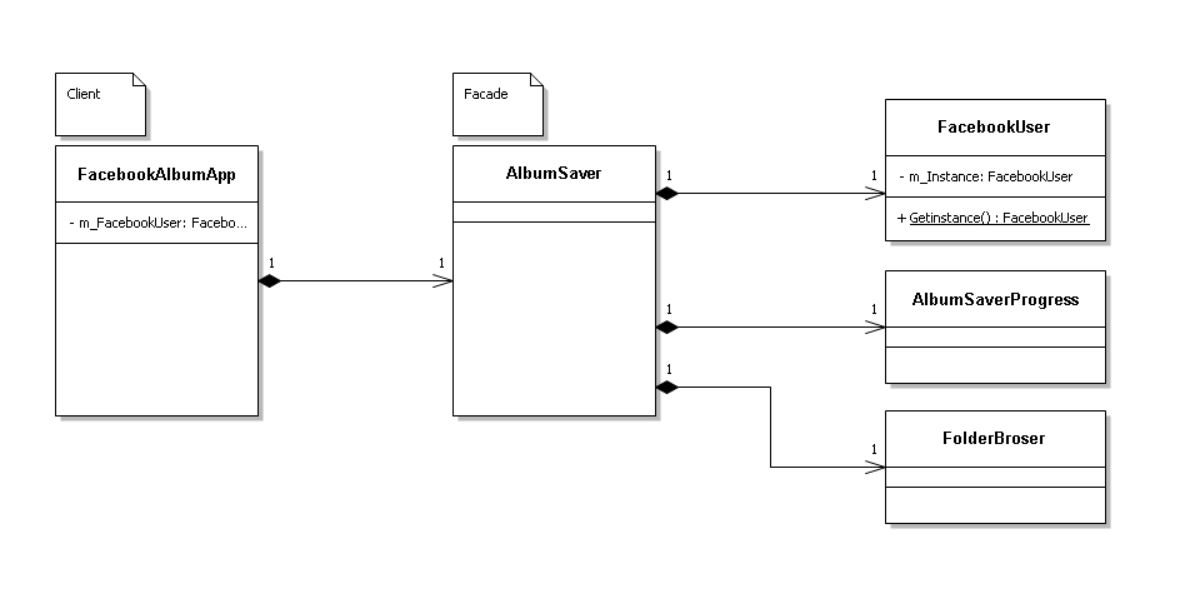
AlbumSaver ניגש ל-FolderBrowser כדי לקבל את המיקום אליו התמונות יישמרו.

בעזרת FacebookUser נביא את אוסף התמונות באלבום, ועבור כל תמונה שיורדת נעדכן זאת ב-AlbumSaverProgress.

* Sequence Diagram

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

* Class Diagram



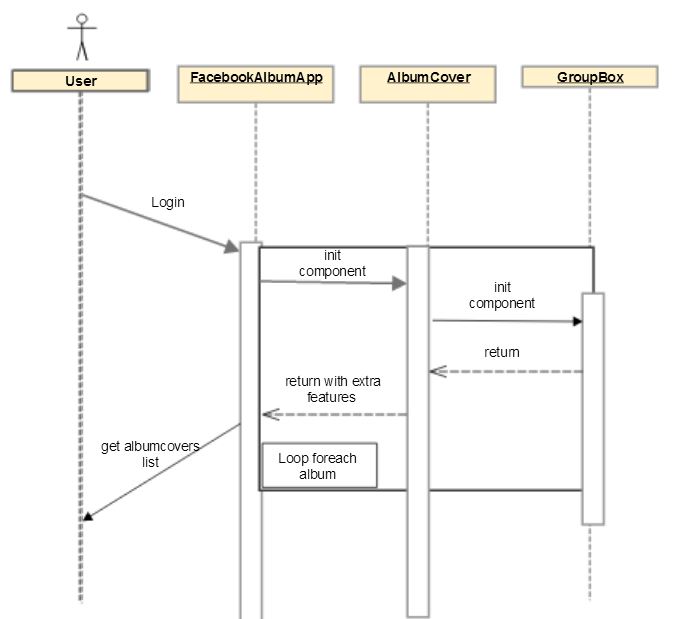
### תבנית מס' 3 – Proxy

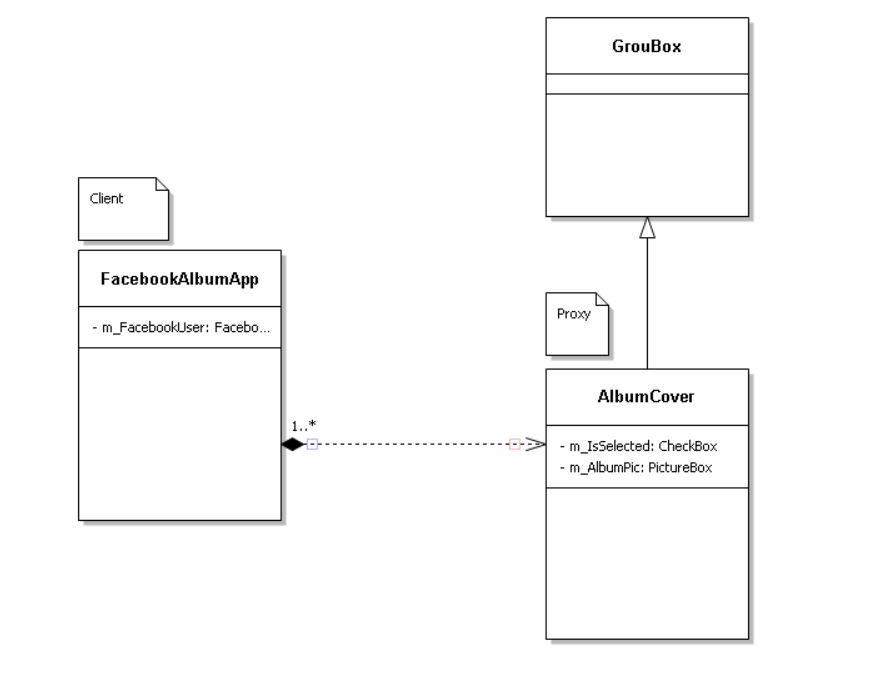
* סיבת הבחירה / שימוש בתבנית:

לצורך מימוש הפיצ'ר של הורדת אלבום תמונות רצינו פקד אשר יתפקד כמו PictureBox, אולם יתן גם יכולת סימון למשתמש באופן פשוט ומקובל, על כן היה צורך לצרף CheckBox ל-PictureBox, כלומר פרוקסי היורש ממחלקה קיימת ובכך מספק שירותים זהים בתוספת יכולת נוספת המולבשת על אותן תכונות של מחלקת האב.

* אופן המימוש:

המחלקה AlbumCover הינה מחלקה היורשת מ-GroupBox ומאגדת בתוכה PictureBox, CheckBox ופעולות נוספות. המחלקה בונה את רכיב הפרוקסי הנ"ל בקונסטרקטור שלה ומשלבת מעט תכונות נוספות כגון העלאה של תמונת האלבום, חיווי האם האלבום נבחר וכו'.

* Sequence Diagram
* Class Diagram



### עבודה אסינכרונית

כאשר המשתמש מבקש לעשות Login, מתבצעת פעולת תקשורת עם שרתי פייסבוק.

בזמן התקשורת ה-UI היה במצב blocking, כלומר לא ניתן להזיז את החלון או לסגור אותו, בעצם החלון קפאה.

בכדי לפתור את הבעיה,

פרט לכך שהעברנו את כל לוגיקת ההתחברות לפייסבוק למחלקה נפרדת בשם FacebookUser, הוספנו מתודת Login המקבלת כפרמטר מתודת - Action<bool> i\_ActionSuccess.

במימוש Login השתמשנו ב-Thread נוסף על מנת לבצע את התקשורת מול פייסבוק,

זה כבר פתר את בעית ה-Blocking.

רצינו בנוסף לבצע שינוים ב-UI כאשר הפעולה הסתיימה, בין אם זו הצליחה או לא, ולכן בסיום מתודת Login יש קריאה ל-i\_ActionSuccess עם פרמטר בוליאני המעיד על הצלחה.

לסיום - דאגנו לכך שהמתודות שמשפיעות על ה-UI בעקבות הפעולה יקראו בעזרת Invoke,

כיוון שהפעולות הללו עדיין מתבצעות על ה-Thread שייצרנו.

### עבודה עם Data-Binding

לצורך עבודה עם Data-Binding הרחבנו את ממשק האפליקציה בתחום של רשימת החברים. כעת בחירה של חבר מתוך הרשימה ב-listBox המתאים מאפשרת לראות נתונים על אותו חבר כגון, תמונתו, שם מלא, יום הולדת, אימייל, קישור ועוד.  
הכל ממומש בצורת Two-Way Automatic Data Binding כאשר האובייקט הינו User (שכן כל חבר הוא יוזר פייסבוק המתקבל מרשימת החברים) ומקור המידע הינו כאמור הליסט-בוקס של החברים.