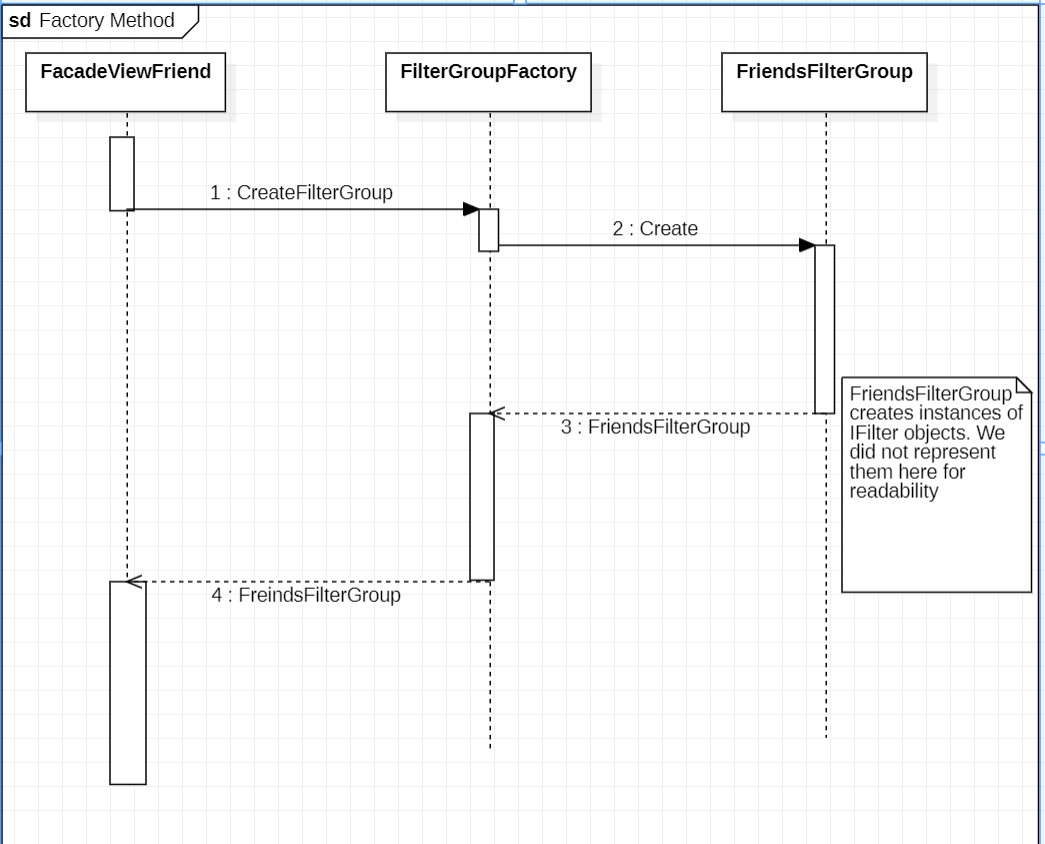
### 

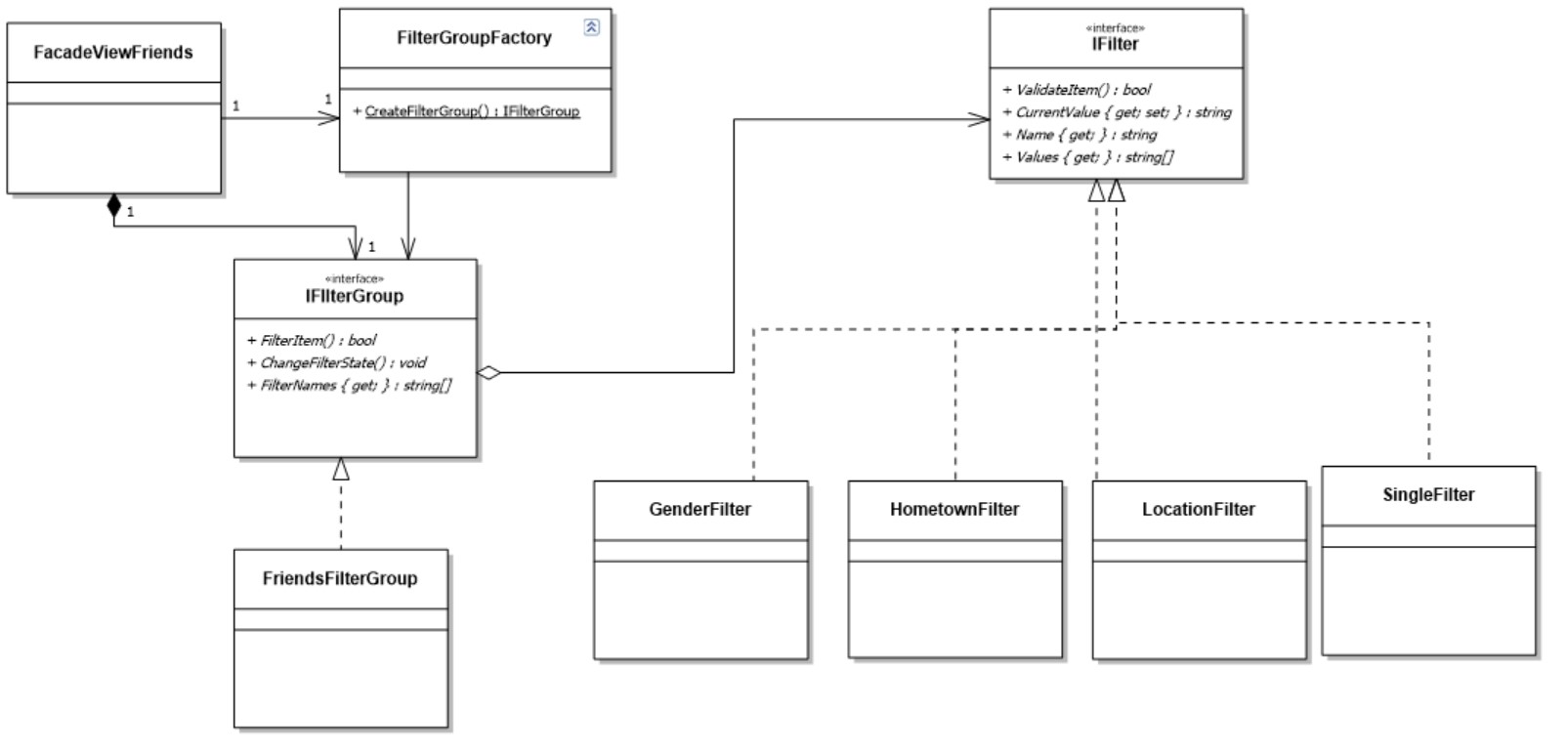
### תיאור קצר של הפיצ'רים שבחרנו לממש בתרגיל הקודם:

* תצוגת חברי המשתמש - הפיצ'ר מציג את חברי המשתמש המשתמשים באפליקציה, כאשר ניתן להפעיל על רשימת החברים פילטרים כדי להציג חברים רלוונטים העומדים בתנאים. כאשר בוחרים חבר מהרשימה מוצגים פרטיו.
* צפייה באלבומי תמונות - אפשרות לבחירת אלבום מאלבומי התמונות של המשתמש וצפייה בתמונות האלבום. ניתן לסדר את סדר הצפייה מהתמונה החדשה ביותר באלבום לישנה ביותר, או להיפך.
* צפייה וניהול פוסטים - מאפשר לצפות בפוסטים שהעלה המשתמש, בתגובות שהוגבו לפוסט וכתיבת פוסט חדש.

### תבנית מס' 1 – Factory Method

* ראינו שעבור חלק מהפעילויות שאנו מציעים באפליקציה אנחנו מפעילים תהליכי **סינון** על המידע, לפי דרישת המשתמש. בנוסף כל פעילות תצטרך אפשרויות סינון שונות, מהיות שסוג האובייקט עשוי להיות שונה, או שהדרישות לסינון עשויות להיות שונות בין פעילות לפעילות. זיהינו שיש פה צורך שחוזר במהותו, עם הבדלים קטנים במימושים השונים, ושיתכן ובעתיד נרצה להוסיף עוד סוגים שונים שלו.
* אופן המימוש: כדי לממש את הדפוס הגדרנו מספר סוגי מחלקות וממשקים שונים  
  IFilter - ממשק המגדיר איך מחלקה שמגדירה קרטריון סינון נבנה, והגדרנו בשלב זה מספר Filters שממשים אותו GenderFilter, SingleFilter, HometownFilter ו - LocationFilter.  
    
  IFilterGroup - ממשק המגדיר איך רכיב המאגד מספר רכיבי סינון החולקים קונטקסט ממומשים.  
  הגדרנו את FriendsFilterGroup הממש את הממשק עבור מסננים לתצוגת חברי המשתמש.  
    
   FilterGroupFactory - מחלקה סטטית שאחראית על להחזיר מופעי FilterGroup לקליינטים בהתאם לצורך.
* Sequence Diagram



* Class Diagram

**המחלקות המקבילות:**

מחלקת FacadeViewFriends הינה מקבילה למחלקת ה - ***Client***.

מחלקת FilterGroupFactory מקבילה למחלקת StaticFactoryClass, ה - ***creator***

ממשק IFilterGroup - מקביל למחלקת ה- ***Product***

מחקלת FriendsFilterGroup מקבילה למחלקת ***ConcreteProduct***

### תבנית מס' 2 – Builder

* סיבת הבחירה / שימוש בתבנית:

בפ'יצר שמציג את חברי המשתמש ראינו צורך להגיד אובייקט שמכיל בתוכו רק את המידע שמעניין את המשתמש. קודם לכן כל "חבר" של המשתמש היה למעשה מסוג "משתמש" בפני עצמו. הוא הכיל המון מידע לא רלוונטי, והמידע שכן היה רלוונטי לא בהכרח היה מומש בצורה שהתאימה לצרכינו. רצינו שנעבוד עם אובייקטים המתאימים למטרתנו, ובהתאם לדרישות המשתמש, כאשר בעתיד נוכל להתאימו לפ'יצרים חדשים.

* אופן המימוש:

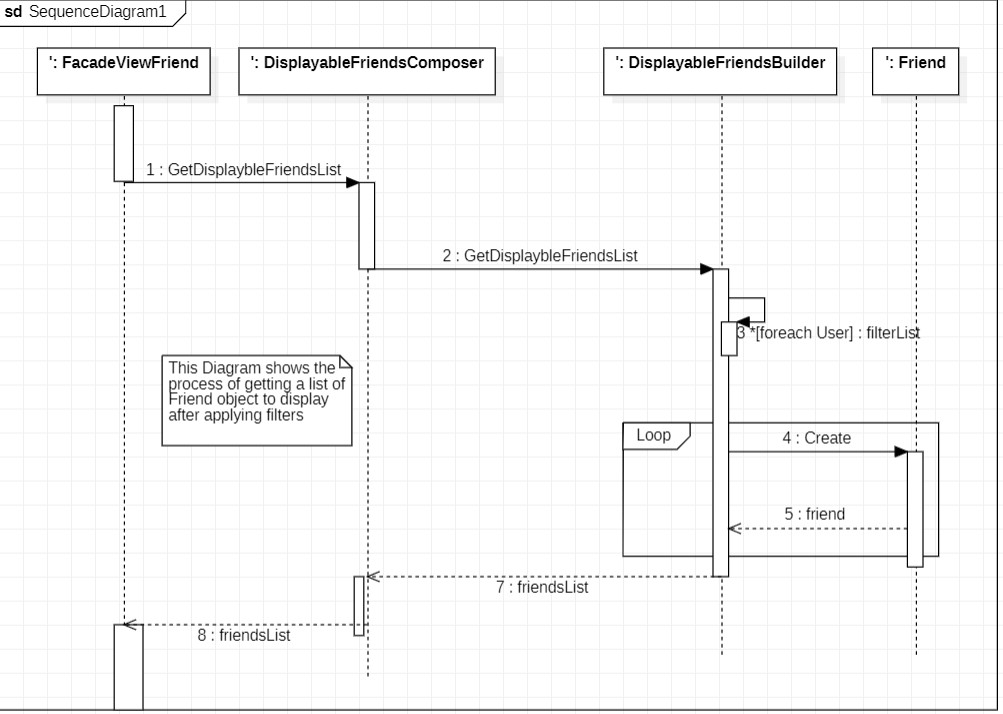
הגדרנו שלוש מחלקות חדשות על מנת לממש את הדפוס, סוג האובייקט החדש שנרצה להחזיר, ה - composer וה- builder.

ה - composer - ממומש בקוד כמחלקת DisplayableFriendsComposer מקבל מהמשתמש את רשימת החברים של המשתמש (אשר הינה אוסף של User), את קבוצת המסננים של הטופס ואת המשתמש המחובר. הקליינט מבקש ממנו את האובייקט המורכב, וה - composer מעביר את דרישות המשתמש (שמבוטאים ב -FilterGroup שקיבל לבנאי)

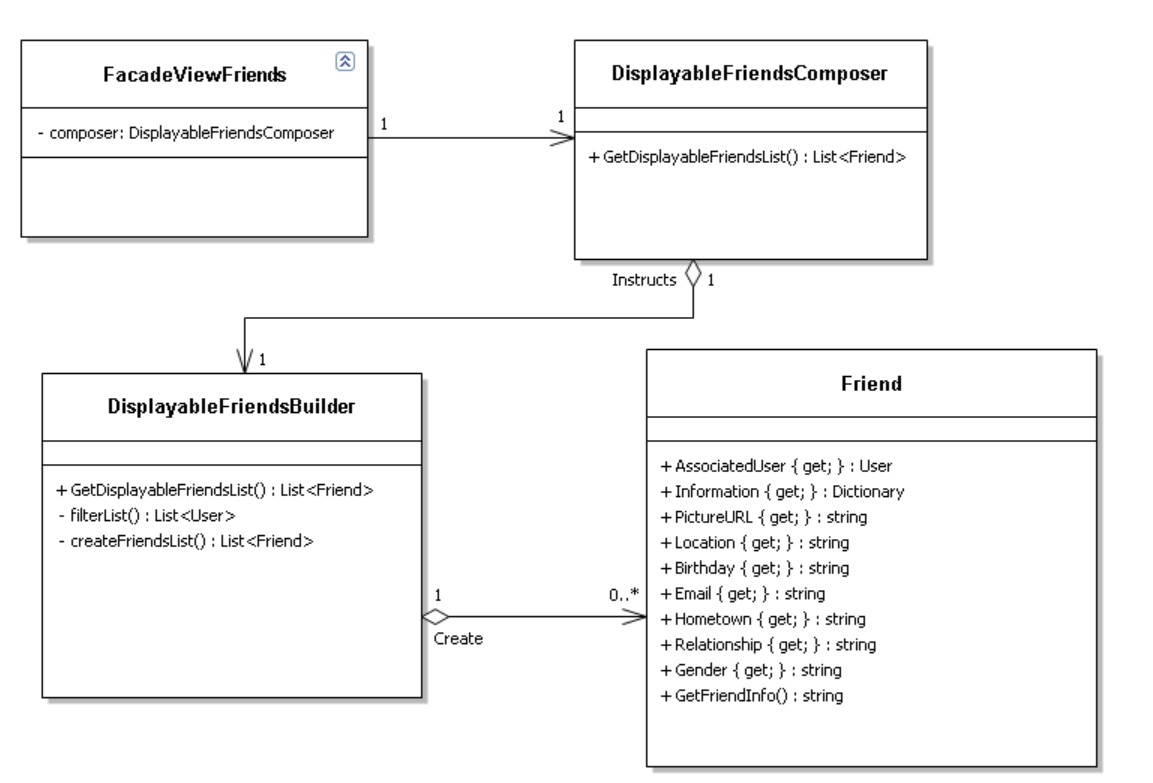
ה - Builder - לאחר שקיבל את הדרישות מה - composer, הוא מסנן את המשתמשים הרלוונטים, ועבור כל אחד מרכיב אובייקט Friend חדש המייצג אותו עם המידע הרלוונטי. לבסוף מחזיר רשימה של מופעי Friend.

מחלקת Friend - מחלקה המאגדת בתוכה את המידע הרלוונטי להצגה באופן הנוח לקליינט לשימוש.

* Sequence Diagram



* Class Diagram



**המחלקות המקבילות**

המחלקה FaceViewFriends מקבילה ל - ***Client***.

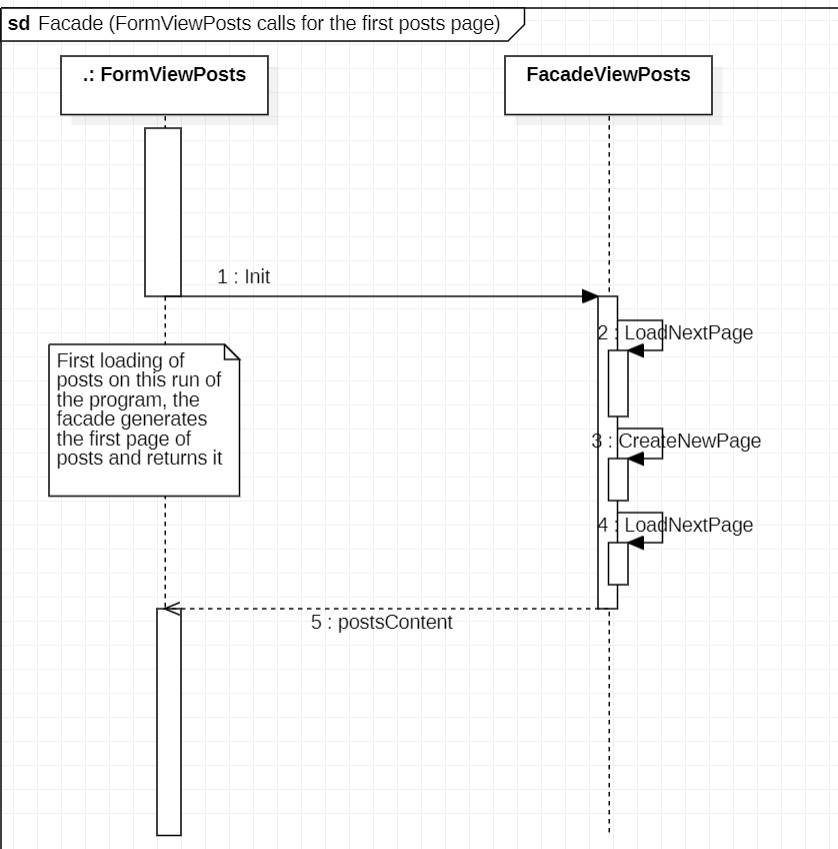
המחלקה DisplayableFriendsComposer - היא ה -***Composer***

המחקלה DisplayableFriendsBuilde - היא ה - ***Builder*.**

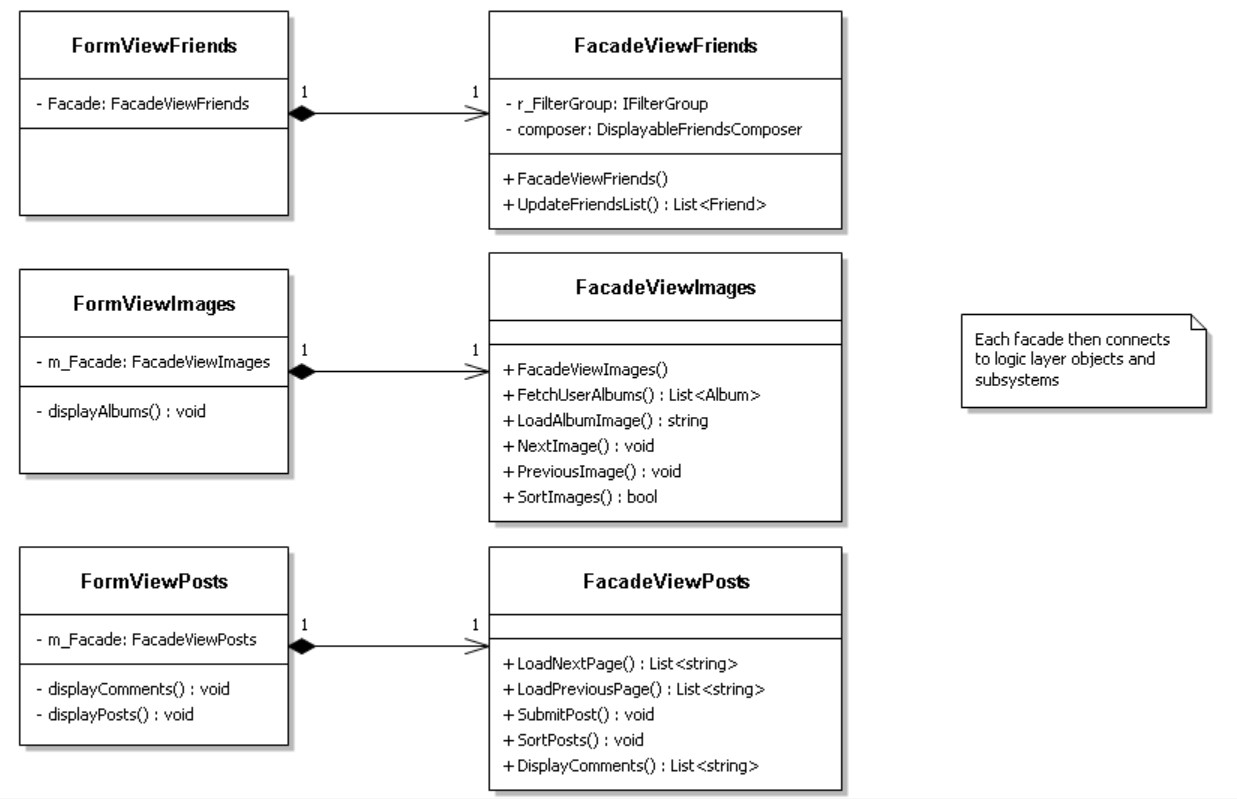
כאשר מחלקת Friend מהווה חלק מהאובייקט המורכב שנחזיר.

### תבנית מס' 3 – Facade

* בחרנו ב - Facade עקב צורך לפשט את שכבת ה -UI. הלוגיקה שכל רכיב UI כלל בתוכו סיבכה ללא צורך את שכבת רכיבי ה - UI.אין צורך שברכיבים שאחרים על תצוגת המשתמש גם נבצע פעולות לוגיות כמו מיון, בניית אובייקטים מורכבים וחישובים, רצינו לבצע הפרדה בין הלוגיקה לבין רכיבי חווית המשתמש, ע"י סיפוק ממשק פשוט בין רכיבי ה - UI ללוגיקה.
* אופן המימוש: עבור כל Form של פעילות (צפייה בחברים, פוסטים ואלבומי התמונות) הגדרנו מחלקת Facade תואמת (לדוגמה עבור FormViewFriends הגדרנו את FacadeViewFriends).  
  כל טופס מחזיק מופע של ה - facade המתאים לו.



* Class Diagram



כאשר הטפסים (FormViewFriends, FormViewPosts, FormViewImages) מקבילים למחלקות ***Client****.*

התייחסות לעבודה א-סיכנרונית

השתמשנו בתכנות א-סנכרוני כדי לטעון את תכני הטפסים בממשק המשתמש במקביל להופעתם הראשונה, זאת על מנת שהתוכנית לא תתקע בפעם הראשונה שבה המשתמש פותח פעילות כלשהי. ניתן למצוא את הקוד הא-סינכרוני בשלושת טפסי הפעילויות (להלן FormViewFriends, FormViewImages, FormViewPosts), כאשר עדכנון רכיבי UI מתבצע בעזרת מתודות ה - Invoke של הרכיבים.

התייחסות למימוש DataBinding

השתמשנו בדאטא ביינדינג על מנת להציג אינפורמציה של מופעי Friend, מחלקה שיצרנו לטובת הפעילות של FormViewFriends. כאשר מתקבל רשימה מסוננת של מופעי חברי המשתמש המחובר שברצוננו להציג, השתמשנו בדאטא ביינדיג כדי לבנות את תצוגת המופע בצורה נוחה ואינטואיטיבית. קישרנו את הdatasource ל - listbox שדרכנו נבחר איזה חבר להציג.