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

**Friends With Common Interest:**

פיצ'ר זה מציג רשימה של החברים איתם יש למשתמש באפליקציה עמודי פייסבוק משותפים (עמודים להם המשתמש והחבר עשו "לייק"). ברשימה זו יוצגו כל החברים עם עמוד או יותר משותפים.

הפונקציה שמבצעת זאת מופיעה בקוד ב-MainLogic, וה-form הראשי של האפליקציה עושה בה שימוש בעת לחיצה על כפתור "Click here to see them!".

**Artists Recommendations:**

פיצ'ר זה עושה שימוש ב-API חיצוני של last.fm.

האפליקציה בודקת מי האמנים האהובים על המשתמש, ובעבור כל אחד מהם ממליצה לו על 3 אמנים דומים.

עיקר ההופעה בקוד היא במחלקה LastFmApi, במחלקה קיימת מתודה שמבצעת קריאת API לשרת של lastfm באופן א-סינכרוני. הקריאה מחזירה תשובה בפורמט XML עם מידע על אומנים דומים לאמן המבוקש. מתוך המידע אנו מסננים רק את שמות האמנים ומחזירים רשימה שלהם.

### תבנית מס' 1 – [Singelton]

* סיבת הבחירה / שימוש בתבנית:

השתמשנו בסינגלטון עבור מחלקת הלוגיקה בתוכנית. מחלקה זו מחזיקה נתונים ומבצעת פעולות לוגיות עבור כלל הטפסים השונים במערכת. בחרנו להשתמש בסינגלטון עבור מחלקה זו כי רצינו לוודא שיש מקור יחיד של אמת עבור הנתונים השונים בלוגיקה של התוכנית, וכדי למנוע מצב בו יש כפילויות של נתונים אודות המשתמש או עבור תוצאות פעולות בלוגיקה. בנוסף רצינו שהנתונים והמופע של מחלקת הלוגיקה יהיה נגיש מכל טופס ללא צורך "להעביר" את המופע הזה בין הטפסים – גישה נוחה מנקודות שונות במערכת אשר תבנית סינגלטון מאפשרת.

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

המימוש במחלקת AppLogic (ובנוסף בטפסים המבצעים AppLogic.instance) :

* הבנאי נמצא בprivate.
* מופע סטטי בתוך המחלקה.
* פרופרטי סטטי שמחזיר את המופע הסטטי , ורק אם המופע עדיין לא נוצר , הוא יוצר כזה.
* Thread Safe : שימוש בlock.
* "שחקנים" :

Singelton:AppLogic

Clients:The forms.

* 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

### תבנית מס' 2 – [Adapter]

* 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

### תבנית מס' 3 – [Façade]

* סיבת הבחירה / שימוש בתבנית:

השתמשנו בתבנית זו כדי לאפשר לטופס הראשי עבודה מול Api חיצוני המספק המלצות מוזיקליות.

ביצענו זאת מכמה סיבות: לצורך בטיחות בשימוש בעבודה מול Api חיצוני שיכולות להיות בו תקלות . כמו כן לצורך פישוט העבודה של הטופס מול הApi (הFacade מחזיר בצורה נוחה את הנתונים הרלוונטים לטופס). לצורך תחזוקתיות ופולימורפיזם – אם בעתיד נרצה להשתמש בApi אחר של המלצות במקום הנוכחי(הנוכחי הוא (lastFm

אז השינויים שנצטרך לבצע יהיו רק בFacade ולא בשכבה המשתמשת בו.

* אופן המימוש:

[תיאור המימוש והיכן ניתן למצוא אותו בקוד]

במחלקת RecommendationsFacade (והשימוש בה בmainForm

המימוש הוא של Facade אטום:

מחלקת lastFmApi ) שאיתה הfacade עובד , מוגדרת internal , ומחלקת הfacade נמצאת

באותו dll .

הmainForm מחזיק מופע של RecommendationsFacade.

הfacade מחזיק מופע של המחלקה lastfm – ומשתמש במתודות שלו כדי למממש את המתודה GetArtistRecommendations שמונגשת באופן נוח לmainForm – המשתמש בfacade.

"שחקנים":

RecommendationsFacade- Façade

mainFrom – client

lastFmApi – package.

* Sequence Diagram

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

* Class Diagram 