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

**פיצ'ר מס' 1 – "FaceRide!":**

פיצ'ר המאפשר למשתמש למצוא טרמפ מכל מקום לכל מקום בעזרת הרשת החברתית.

כדי להשתמש בפי'צר הנ"ל, על המתמש לבצע:

1. הזנה למערכת את נקודת האיסוף, או לחילופין שימוש במיקמו הנוכחי של המשתמש בעזרת שירותי מיקום.

2. הזנה למערכת את היעד, אותו יוכל להזין באופן ידני או "למשוך" מתוך רשימת האירועים אליו הוא רשום. לאחר מכן יתאפשר למשתמש לצפות ביעד בו בחר במפה בחלון נפרד.

3. בחירת רדיוס חיפוש רצוי ומגדר.

המערכת מאתרת מרשימת חבריו של המשתמש את כל אלו אשר נמצאים ברדיוס החיפוש בו בחר ומציגה אותם כרשימה המכילה מידע בסיסי אודותם. בעת בחירת המשתמש (לחיצה כפולה על החבר הרצוי), נפתח מסך ובו הודעה Auto Generated המוכנה לפרסום על הקיר או לשליחה דרך ה-Messenger, המציגה את בקשת המשתמש מאותו חבר לטרמפ אל היעד אליו הוא רוצה להגיע.

**מחלקות המעורבות בפיצ'ר**:

* DesktopFacebook
* FaceRideManager
* LocationServices
* AppController
* MapForm

**פיצ'ר מס' 2 – "Contacts":**

פיצ'ר זה נועד ליצור רשימת אנשי קשר מנתוני חברי הפייסבוק של המשמש ויצוא הרשימה לסמרטפון של המשתמש.

דרך הפעולה:

1. על המשתמש לעבר ללשונית הcontacts.

.2 לאחר מכן על המשתמש ללחוץ על כפתור הshow.

3 המערכת תיצור רשימת אנשי קשר ותזין אוטומטית את הנתונים לטבלה וויזואלית.

4. אופציונלי: המשתמש יכול לערוך בטבלה את נתוני החברים מהפייסבוק שהוזנו לפי טעמו.

5. אופציונלי: המשתמש יכול לערוך עמודות נוספות רלוונטיות לרשימת אנשי קשר שאינן חלק מנתוני הפייסבוק אך הן כן רלוונטית בשביל אנשי הקשר, חלק מעמודות אלו מותאמות בשמותיהן לתצוגה מיוחדת במכשרי הסמרטפון.

6. על המשתמש ללחוץ על כפתור הsave כדי לשמור את הנתונים כקובץ.csv

.7 בלשונית תפתח דפדפן פנימי אשר נמצא בדף הסבר מיוחד על איך לייצא את רשימת אנשי הקשר באופן מיידי לחשבון Gmail שיאפשר שימוש פרקטי ברשימת אנשי הקשר שנוצרה בסלולרי האישי של המשתמש.

**מחלקות המעורבות בפיצ'ר**:

* DesktopFacebook
* AppController
* GoogleContacts (המכיל תתי מחלקותEmail, Organization, Websites, Address, Phone )

### עבודה אסינכרונית

עבודה אסינכרונית בעזרת Threads ניתן למצוא לדוגמא ב:

**AppForm.cs**:

**שורה 86 - private void initializeAllTabs**

המטודה מתאחלת את כל הטאבים של הטופס בצורה אסינכרונית באמצעות גדיל לכל טאב.

האתחול מתבצע בצורה אסינכרונית על מנת לקבל טעינה מהירה יותר של הטופס.

**AppController.cs**:

**שורה 24 - private void startThreadsForAlbumsTabUpdate**

המטודה מופעלת כאשר המשתמש רוצה לעדכן את התמונות המוצגות בטאב "Albums".

תפקידה של המטודה הוא עדכון כל ה-PictureBoxes בטופס.

הפעולה מתבצעת בצורה אסינכרונית (גדיל עבור כל PictureBox) על מנת לספק למשתמש חוויה מהירה יותר של טעינת התמונות.

**FacebookAuthenticator.cs**:

**שורה 39 - public User LoginUser()**

המטודה מופעלת לאחר שהשמתמש ביצע אימות התחברות.

במידה והמשתמש סימן כי ברצונו להישאר מחובר, המטודה שומרת את הגדרות האפליקציה למכשיר.

שמירת ההגדרות מתבצעת בצורה אסינכרונית ע"י גדיל נפרד כדי למנוע את תקיעת האפליקציה במהלך השמירה.

### Binding Sources

שימושים בקוד:

1. **UserEventsComponent.cs:**

שורה 77 – EventsBindingSource

1. **:AppForm.cs**

שורה 479 – FriendsBindingSource

1. **AppForm.cs:**

שורה 125 - AlbumsBindingSource

### תבנית מס' 1 – Abstract Factory

* סיבת הבחירה / שימוש בתבנית:

במערכת קיימות הישויות הבאות:

1. UserAlbumPicturesComponent
2. UserEventsComponent
3. UserProfileComponent

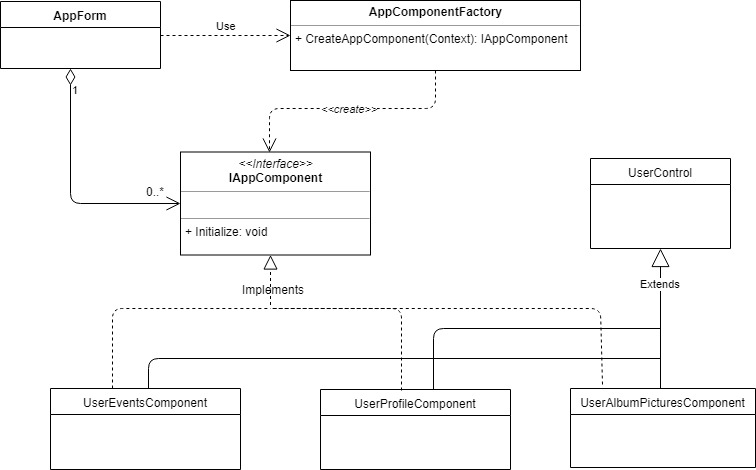
הישויות הנ"ל מתקיימות תחת משפחה פולימורפית אחת – IAppComponent אשר היוותה את המוטיבציה ליצירת AppComponentFactory המשמשת לייצור אובייקטים מן המשפחה (פירוט נוסף בדיאגרמות).

* אופן המימוש:

את המימוש ליצירת מופעי הישויות ניתן למצוא במחלקה AppComponentFactory תחת הפרוייקט ""View.

הממשק המשותף לישויות מוגדר בקובץ IAppComponent.cs.

* Sequence Diagram
* Class Diagram



### תבנית מס' 2 – [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

### 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