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

חישוב מסלול בעזרת Google Maps-

באפליקציה למשתמש ניתנת אופציה לחשב מסלול ב Google Maps המתחיל במיקום לבחירתו ומסתיים במיקום לבחירתו, כאשר ניתנת לו האפשרות לבחור מבין הצ'ק אינים הקודמים שלו ומהעיר שאותה סימן כHOME TOWN ,פיצ'ר ישמש את המשתמש לתכנון טיולים במיוחד למקומות שבהם היה בעבר וביצע צ'ק אין.

נעשה בצורה הבאה:

כנקודת התחלה ניתן להזין טקסט חופשי או לייבא דרך הסרבר של פייסבוק את ה”hometown” של אותו משתמש. השארת שדה ריק תחשב את המסלול מהמיקום הנוכחי. לאחר מכן לבחירת היעד יבחר המשתמש בין הזנת טקסט חופשי או בחירת ביקור חוזר באחד מה"check-ins" שביקר בהם בעבר בפייסבוק. לאחר בחירת מוצא ויעד יוכל לבחור את סוג המסלול: נהיגה, תחבורה, רכיבה או הליכה. לבסוף ילחץ על “Plan my trip” ויוכל לראות בחלון שיפתח את הצעות המסלול שבחר בגוגל מפות.

ניתן למצוא את הפיצ'ר בקוד במחלקת הממשק (MainForm) במתודה startTripButton\_Click ובהמשך במחלקת AppLogic במחלקה GetTripUrl().

עריכת תמונת רקע:

באפליקציה למשתמש ניתנת אופציה לערוך ולשמור את תמונת הרקע שלו בצורה הבאה:

* בחירת תמונה הרקע שלו כרקע לאפליקציה
* "הרקדת" התמונה ע"י הזזה.
* סיבוב תמונה (בכל פעם התמונה תסתובב 90 מעלות ימינה).
* הפיכת התמונה מצבעונית לשחור לבן (תרחיש הsequence diagram)

כאשר יוכל המשתמש לשמור את התמונה על מכשירו האישי. אם למשתמש אין תמונת רקע בפייסבוק תוצג תמונת הפרופיל שלו והפיצ'ר יתנהל כרגיל.

ניתן למצוא את הפיצ'ר בקוד במחלקת הממשק (MainForm) במתודה activatePhotoButton\_Click ובהמשך במחלקת AppLogic (ע"מ לערוך את התמונה) במחלקות RotateImage() ו - TurnImageBlackAndWhite().

### תבנית מס' 1 – Proxy

* סיבת הבחירה / שימוש בתבנית:

התבנית Proxyעוזרת לנו "לעטוף" את הפונקציונליות שמפעילה המערכת, ובכך עוזרת לנו להוסיף לוגיקה משלנו, בעצם שכלול הפונקציונליות של הFacebookWrapper.

במקרה שלנו נרצה להציג את גיל הUser אך לנו נתון תאריך הלידה שלו, ולכן נצטרך לספק הזרקה לוגית ע"מ להציג את גיל המשתמש במספר ע"י ביצוע מספר פעולות בהתאמה. הפתרון לבעיה זו הוא מימוש תבנית עיצובית Proxy שמאפשרת שינוי/ הוספה של מידע, כך שמתאריך הלידה נוכל להציג למשתמש את גילו.

* אופן המימוש:

מימשנוinterface IAgeCalculator (מייצג את ISubject) שתפקידו לקשר תחת קורת גג אחת את UserAge (מייצגת את RealSubject ומחזיקה את תאריך לידתו של המשתמש דרך קומפוזיציה לUser) ואת UserAgeProxy (מייצג את Proxy), כאשר גם מחלקת UserAge וגם מחלקת UserAgeProxy יממשו אותו, ובעצם יאפשרו לקבל את גילו המפורש של המשתמש מהנתון של תאריך הלידה שלו.

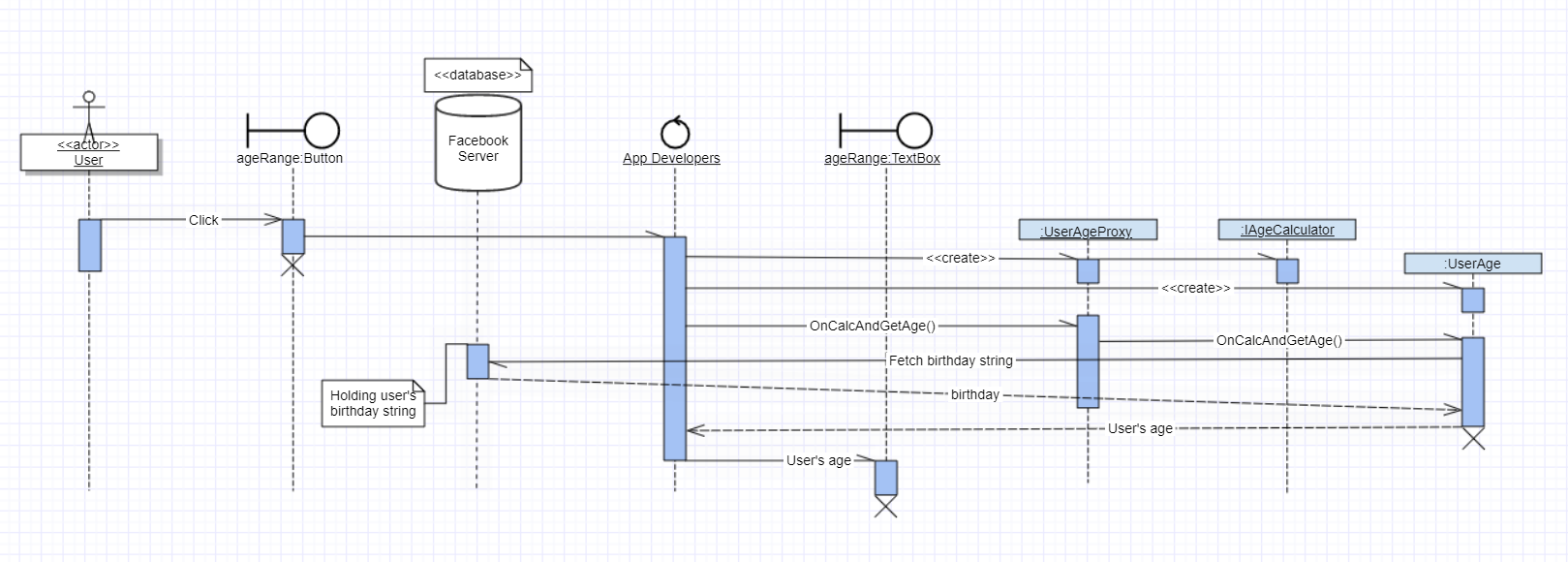
ניתן למצוא אותו בקוד:

בממשק המשתמש ישנה המתודה בשורות 438 – 441 במחלקת MainForm שתקבל את גילו של המשתמש בעזרת קריאה למתודה GetUserAge של מחלקת AppLogic בשכבה הלוגית בשורות 117 – 120.

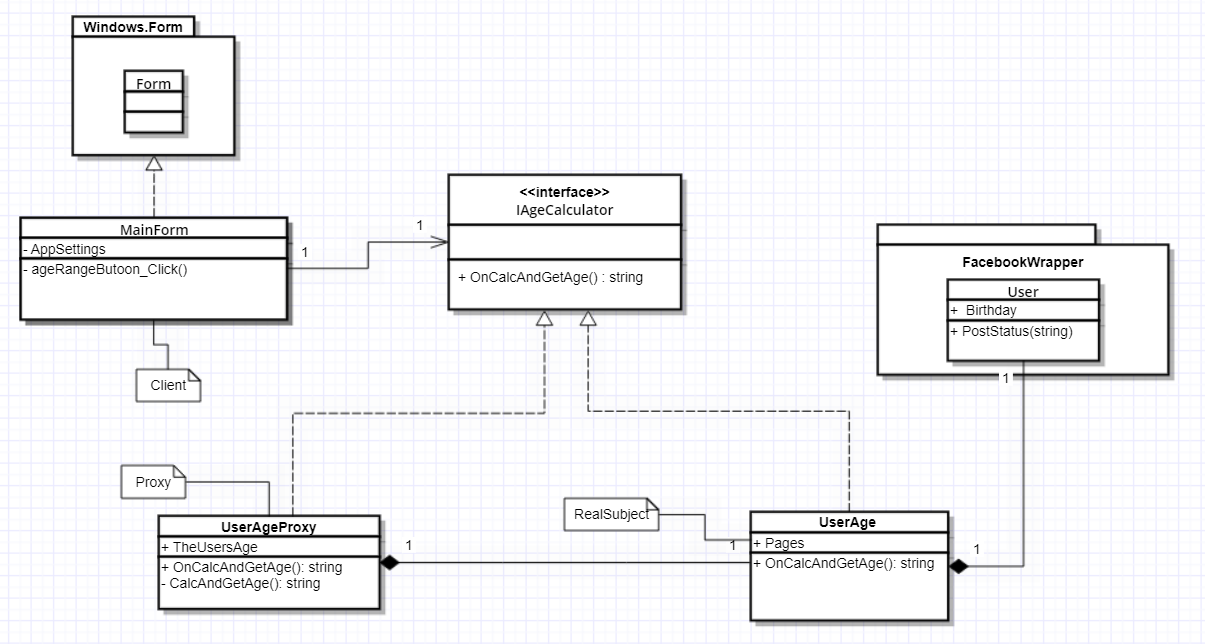
GetUserAge תיצור ממשק IAgeCalculator מסוג UserAgeProxy שמפעילה את מתודת OnCalcAndGetAge של הממשק IAgeCalculator.

מימוש OnCalcAndGetAge במחלקת UserAgeProxy נמצא בשורות 25-28 כאשר הוא משתמש במימוש של UserAge לאותה מתודה ממשקית, שמחזירה לו את תאריך לידתו של המשתמש (ממומשת ב UserAge בשורות 14-17). כאן תתרחש הזרקת הלוגיקה, כאשר UserAgeProxy אחראי לחישוב גיל המשתמש והחזרתו לGetUserAge() ומשם לממשק המשתמש.

* Sequence Diagram



* Class Diagram



### תבנית מס' 2 – Composite

* סיבת הבחירה / שימוש בתבנית:

בתכנית שלנו יש אופציה לפרסם פוסט בעמוד פייסבוק שסימנת לו לייק.

רצינו להוסיף את האפשרות למשתמש לפרסום בכמה עמודים במקביל או בעמוד יחיד בפעולה יחידה שתפעל באותו האופן. ככה המשתמש יוכל למשל לפרסם חיפוש דירה בכמה דפים במקביל ללא התאמצות נוספת, ואף לחבר שתי קבוצות דפים לקבוצה אחת גדולה בעתיד.

הפתרון לבעיה זו היא תבנית עיצובית Composite שמאפשרת להתייחס למספר פריטים באותו האופן כמו לפריט יחיד.

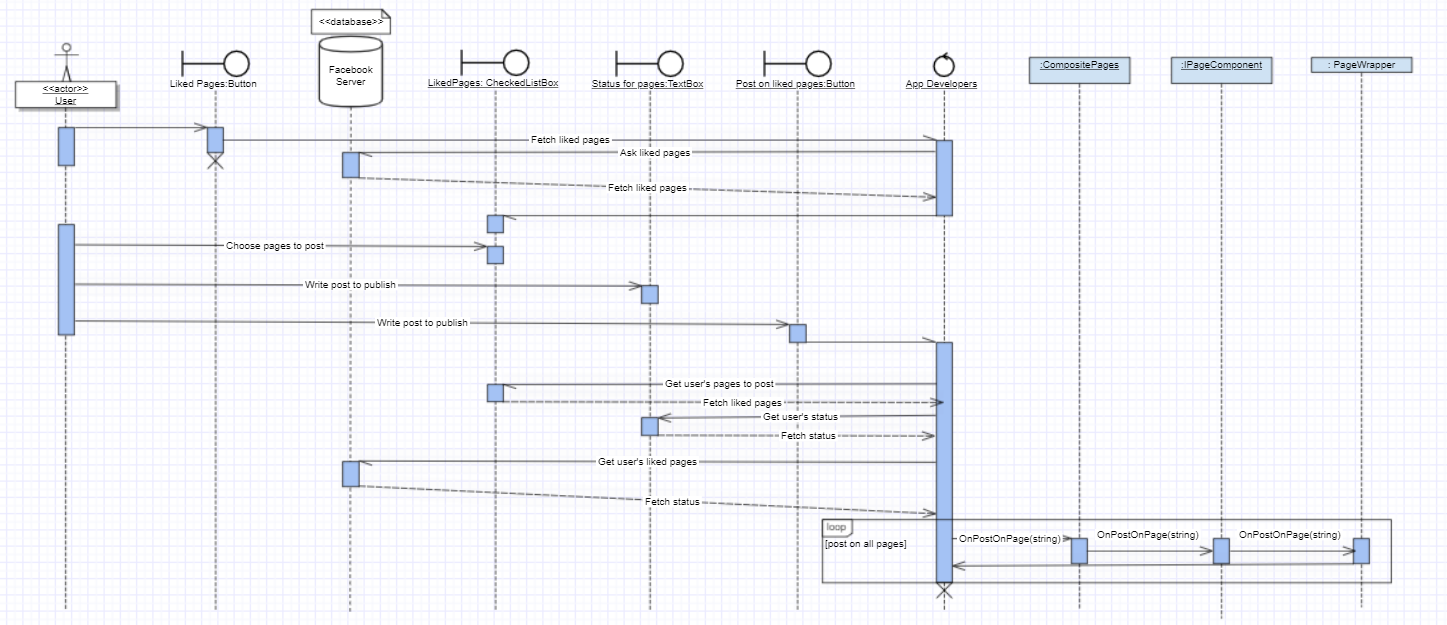
* אופן המימוש:

מימשנו interface IPageComponent (מייצגת את Component) שמממש פעולות המשותפות לדף יחיד ולאוסף דפים, כאשר גם מחלקת PageWrapper (מייצגת את Leaf) וגם מחלקת CompositePages (מייצגת את Composite) יממשו אותו, ובעצם יאפשרו לבצע את אותן הפעולות בצורה שתראה למשתמש דומה.

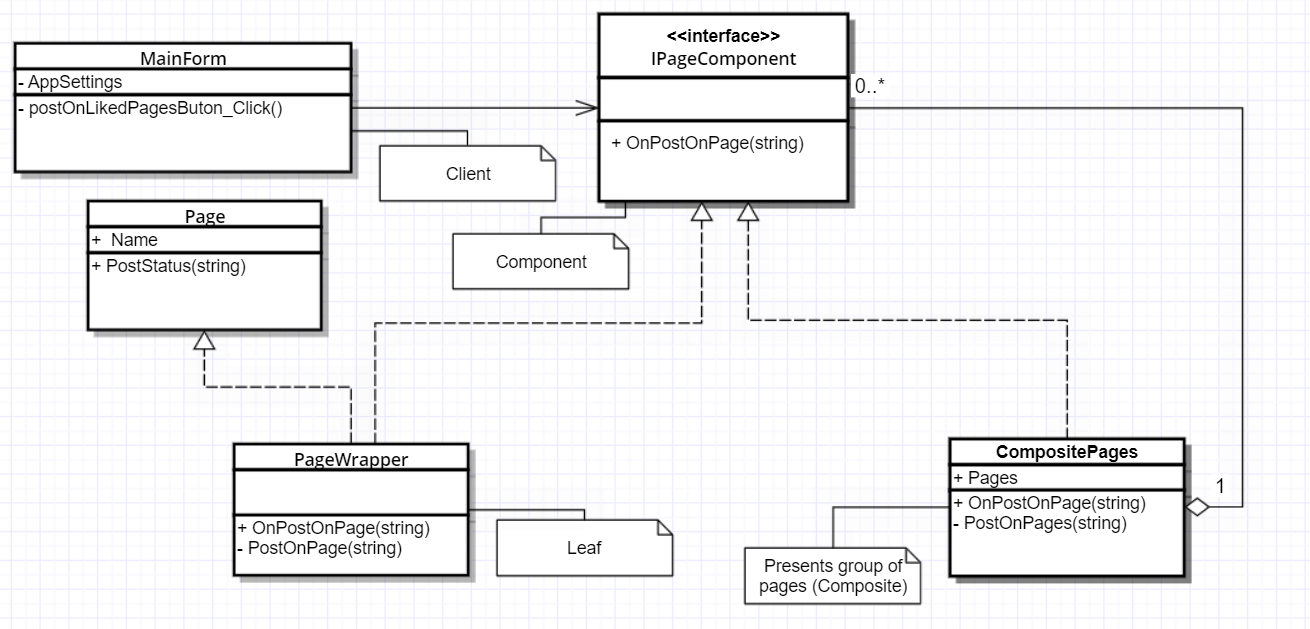
ניתן למצוא אותו בקוד:

בממשק המשתמש, תופעל המתודה בשורות 527 – 500 במחלקת MainForm (מייצגת את Client בתבנית) שתארגן את הדפים שנבחרו ותשלח אותם לשכבה הלוגית למתודה PostToPages בשורות 122 – 125 במחלקה AppLogic שתיצור אוביקט מסוג CompositePages שמייצג דף או אוסף דפים, ותפעיל את מתודת OnPostOnPages שתדע בזכות השימוש בתבנית Composite שמימשנו כיצד להשתמש בדף או בדפים בצורת רבים ובצורת יחיד כאחד ואף לצרף קבוצת דפים לקבוצת דפים אחרת.

* Sequence Diagram



* Class Diagram



### תבנית מס' 3 – Facade

* סיבת הבחירה / שימוש בתבנית:

המערכת שלנו מכילה סוגים שונים של פונקציונליות ולוגיקה "מאחורי הקלעים". על מנת לעשות סדר ולהנגיש את המערכת באופן פשוט לתפעול בחרנו להשתמש בתבנית Facade, שתוכל לעשות סדר וחלוקה נכונה בלוגיקה של המערכת. בצורת בניה זו נרוויח גם בטיחות שימוש שכן הורדה חשיפתן של הרבה תתי- מחלקות, וכמובן עקרון הEncapsulation (במיוחד בגרסה האטומה שלו) שממומש ע"י הסתרה והוספת שלב ביניים מקשר.

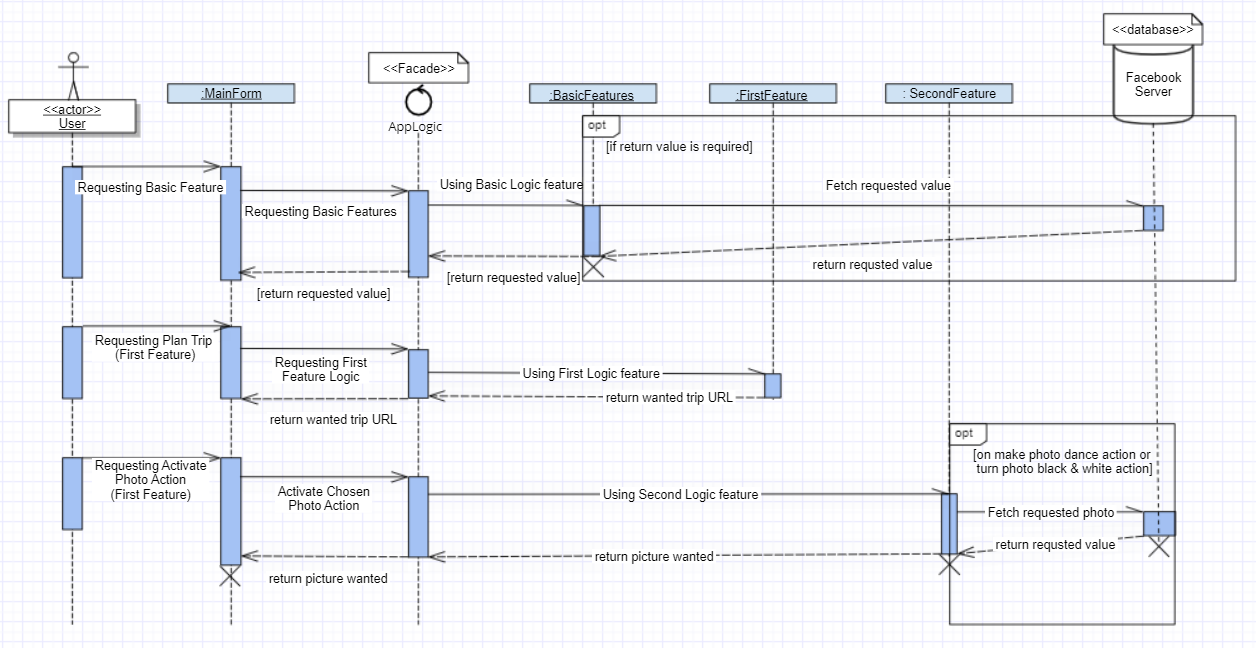
* אופן המימוש:

מחלקת MainForm (מייצגת את Client) שאחראית על ממשק המשתמש תחזיק בקומפוזיציה מופע של AppLogic (מייצגת את Facade), וכך יתנהל עבודת המשתמש מול הרכיב הלוגי, בצורה פשוטה ומסודרת.

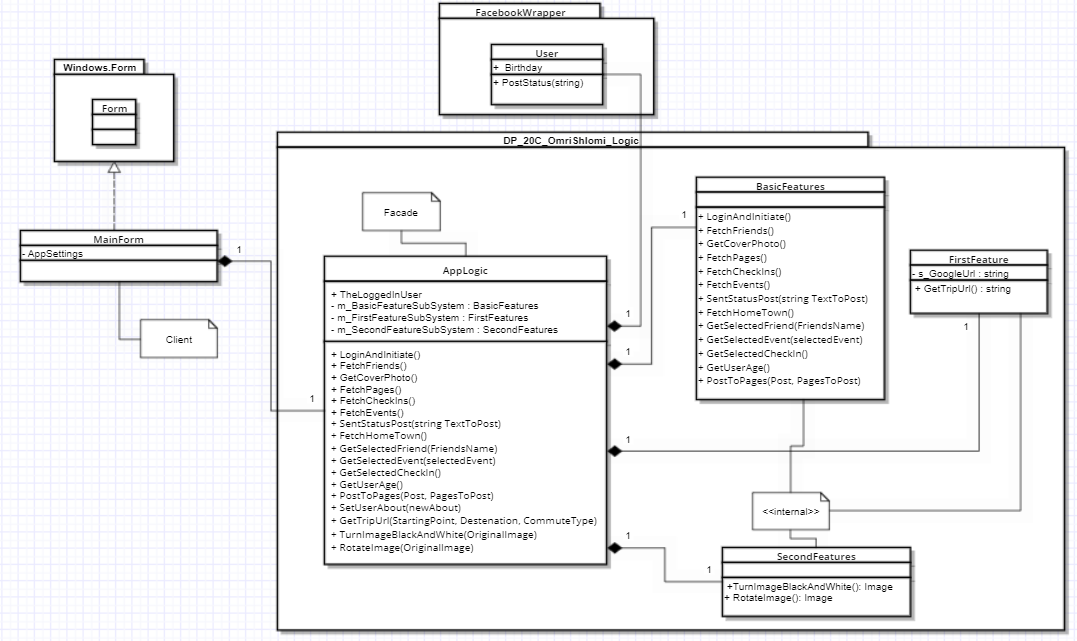
מחלקת הAppLogic תחזיק בתוכה 3 שדות פרטיים שיהיו אחראיים על מרכיבים לוגיים בתכנית (מייצגות את SubSystems): BasicFeatures (מחלקה שאחראית על כל הפיצ'רים הבסיסיים במערכת), FirstFeature (מחלקה שאחראית על הפונקציונליות של הפיצ'ר הראשון) ו SecondFeature (מחלקה שאחראית על הפונקציונליות של הפיצ'ר השני).

שלוש המחלקות הן internal שדואגות כי הFacade יהיה אטום (נגישות רק לאותו הdll שבו נמצאות) ומחזקות את בטיחותו. ניתן למצוא את הFacade בקוד במחלקות המתוארות כאשר הוא ממומש דרך קומפוזיציה ומגיע לידי ביטוי בכל המתודות וחברי המחלקות.

* Sequence Diagram



* Class Diagram



**דרישות נוספות**

**עבודה אסינכרונית**

נעשה שימוש בתכנות אסינכרוני במחלקה –

MainForm

במטודה שאחראית להתחברות ואתחול הערכים של פקדים שונים באפליקציה , שורות 26-33

לאחר קישור בין הפרופרטי שמחזיק את המשתמש השייך לשכבה הלוגית, נוצר תהליך אסינכרוני האחראי לאתחול חלק מבין הפקדים במידע המתאים להם,הבאת תמונת הקאבר של המשתמש, תמונת הפרופיל,הצגת הודעה למשתמש הכוללת את שמו והעברת כל הפקדים באפליקציה למצב ENABLED

תהליך זה קורה "במקביל" לריצת התהליך הראשי,משפר מעט את ריצת התכנית ומונע ממנה "להתקע" בזמן שהיא מביאה נתונים אלו.

בשתי מטודות האחראיות לאתחול רשימת הצ'ק אינים בפקדים המתאימים נעשה שימוש בתכנות אסינכרוני גם כן.

שורות 192-195 ו 345-348

נוצר תהליך האחראי על התקשרות לשכבה הלוגית והבאת רשימת הצ'ק אינים. התהליך מקבל מטודה לבצע , מטודה זו אחראית על הבאת הצ'ק אינים של המשתמש מהחלק הלוגי,

תהליך זה קורה "במקביל" לריצת האפליקציה ולא בתהליך הראשי , בזמן שהוא רץ ניתן להזיז את הטופס ולבצע פעולות אחרות , תורם לשיפור חווית המשתמש.

**עבודה עם Data Binding**

קישרנו את הפאנל dataBindingPanel לאובייקט User , הפאנל מביא את פרטי ה-ABOUT של המתשמש באמצעות פיצ'ר הData Binding של ויז'ואל סטודיו, לאחר עזיבת תיבת הטקסט שבה ניתן לשנות את ה-ABOUT מתעדכן הABOUT של המשתמש.

בחרנו בשימוש זה כי רצינו שיעשה עדכון אוטומטי לשדה הabout.

שימוש נעשה במחלקהMainForm , מטודה ששמה bindDataSources.