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

* הורוסקופ – פיצ'ר המאפשר למשתמש לקבל מידע אודות תאריך הלידה שלו.

הפיצ'ר שולף את תאריך הלידה של המשתמש (במידה והמשתמש סיפק אותו לפייסבוק, במידה ולא - המשתמש יצטרך להכניס את התאריך ידנית) וע"י התאריך נשתמש במחלקת ZodiacSignFactory שמחזירה לו אובייקט מסוג ZodiacSign המחזיק את המזל של המשתמש.

בעזרת המזל המשתמש יוכל לקבל הורוסקופ יומי, לקבל את המזלות המתאימים לו וגם יוכל לבחור את המזל של בן הזוג ולקבל מידע אודות התאמה בין המזל שלו למזל שבחר.

המימוש נמצא בפרויקט נפרד (קובץ dll) הנקרא HoroscopeController, ויזואליות הפיצ'ר תימצא ב-MainForm תחת Horoscope Tab.

* פלייליסט – פיצ'ר המאפשר למשתמש ליצור פלייליסט משלו.

הפיצ'ר שולף את הקישורים אותם המשתמש פרסם בעברו ומכניס אותם לפלייליסט הנקרא "Facebook Posts".

המשתמש יכול ללחוץ פעמיים על וידאו בכדי לצפות בו מצד ימין.

הפיצ'ר מתייחס ל-2 אתרים המספקים שירותי צפייה בסרטונים – YouTube , Vimeo. (אם בעתיד נרצה להוסיף אתרים, הקוד לא יצטרך להשתנות מלבד הוספה של בן היורש ממחלקת Song ומממש את הפונקציונליות שלו – מונחה עצמים).

המשתמש יכול להוסיף שירים לפלייליסט שלו, יכול למחוק שירים מהפלייליסט ואף לפרסם שירים בפייסבוק.

המימוש נמצא בפרויקט נפרד (קובץ dll) הנקרא PlaylistController. ויזואליות הפיצ'ר תימצא ב-PlaylistForm – חלון הנפתח בלחיצת כפתור של "Get Playlist" ב- MainForm.

### תבנית מס' 1 – Factory Method – Static Factory Class

* סיבת הבחירה / שימוש בתבנית:

נרצה לייצר מחלקה מסוג ZodiacSign ללא תלות במזל של המשתמש הנוכחי.

התבנית Factory פותרת לנו את בעיית האי-ידיעה ע"י כך שמחזירה לנו מחלקה מסוג ZodiacSign המצביעה לבן הרלוונטי לפי מזל המשתמש.

כיוון שהשתמשנו בפולימורפיזם (כולם מתנהגים בצורה דומה) נוכל להתייחס למשתנה שחזר אלינו כ- ZodiacSign ללא תלות לאיזה מזל הוא מצביע.

בנוסף, אם בעתיד נרצה להוסיף מזל נוסף, נצטרך לשנות את הקוד במחלקת ה-Factory בלבד (מלבד להוספת class נוסף למזל הרלוונטי) אך איננו צריכים לגעת בקוד של ה-client.

* אופן המימוש:

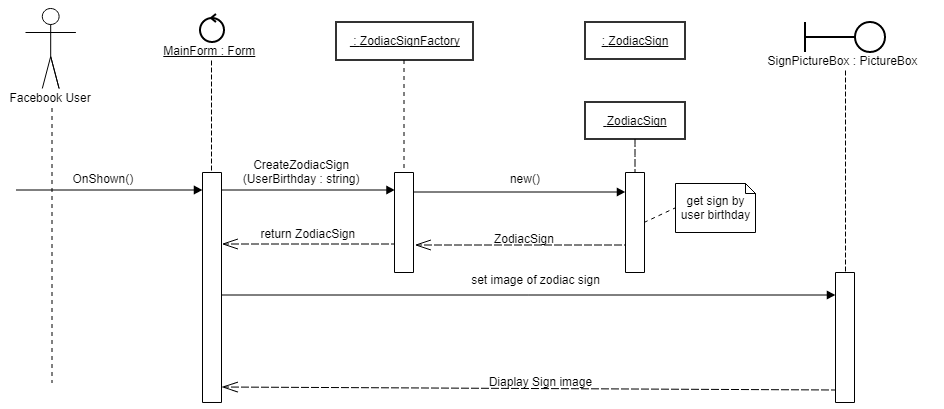
בנינו מחלקה בשם – ZodiacSignFactory.

המחלקה מספקת 2 שירותים – קבלת מזל ע"י תאריך לידה, קבלת מזל ע"י מזל.

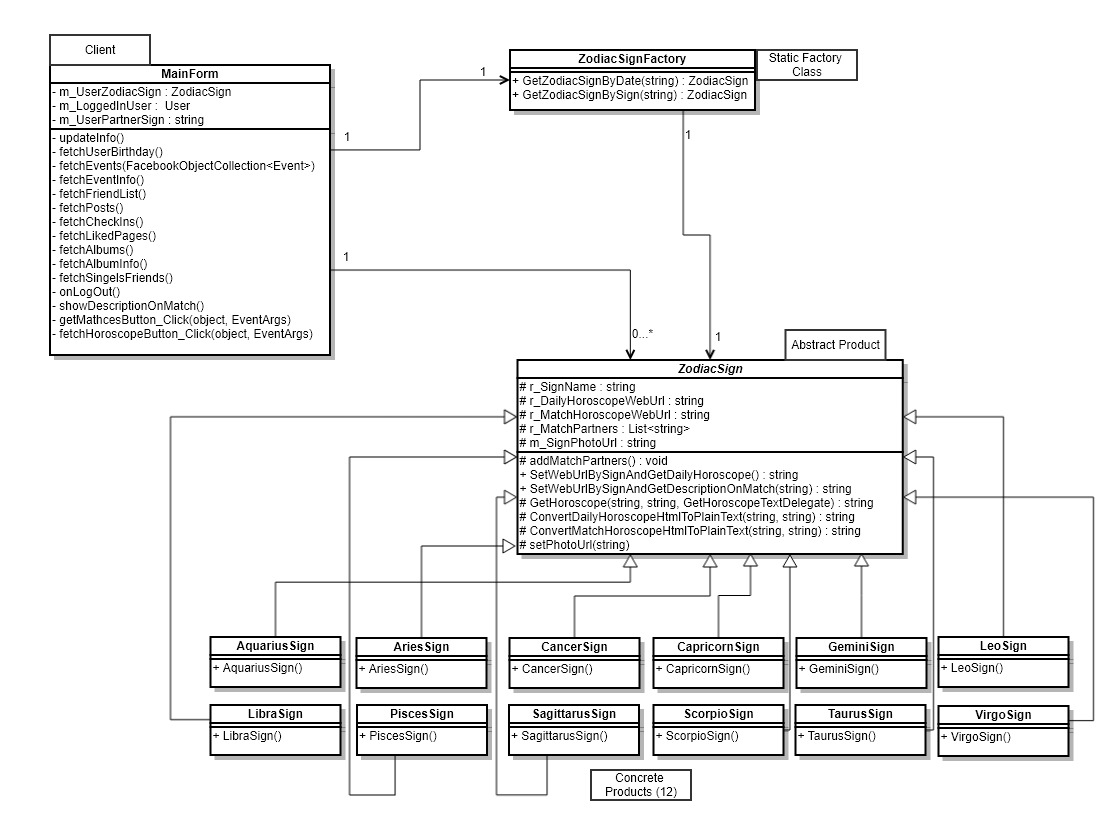
המחלקה מחשבת את המזל הנוכחי ע"פ תאריך מתאים/מזל מבוקש ומחזירה ZodiacSign המאותחל לבן המתאים (המזל הרלוונטי) לפי המזל הנוכחי.

ניתן למצוא את המימוש בקוד ב-HoroscopeController.

* Sequence Diagram



* Class Diagram



### תבנית מס' 2 – Facade

* סיבת הבחירה / שימוש בתבנית:

ישנן 3 אופציות לראות את רשימת החברים:

1. רגיל
2. ממוין לפי שם פרטי
3. ממוין לפי שם משפחה

רצינו להוציא את המימוש של הבאת הנתונים של המשתמש (רשימת חברים לפי סדר מבוקש) למקום נפרד. ע"י כך אנו מבצעים את כל תהליכי החישוב "מאחורי הקלעים" ול-MainForm (ה-client) אין ידיעה אילו חברים מוצגים בפועל. אם בעתיד נרצה להוסיף שיטת מיון חדשה, כל שנצטרך לעשות הוא להוסיף את המיון המבוקש לתבנית שמימשנו וה-MainForm (ה-client) לא ירגיש בהבדל.

* אופן המימוש:

יצרנו dll נפרד בשם FriendsColntroller המכיל 3 מחלקות:

1. FriendsFacade (public class) - מחלקה המכילה מתודה ראשית אשר מבצעת את תהליך הבאת הנתונים ללקוח, במקרה הזה ה-MainForm, בעזרת תת המערכת שנמצאת אצלה בקומפוזיציה (מחלקת FriendsManager).
2. FriendsManager (internal class) – מחלקה המכילה את כל פעולות החישוב.

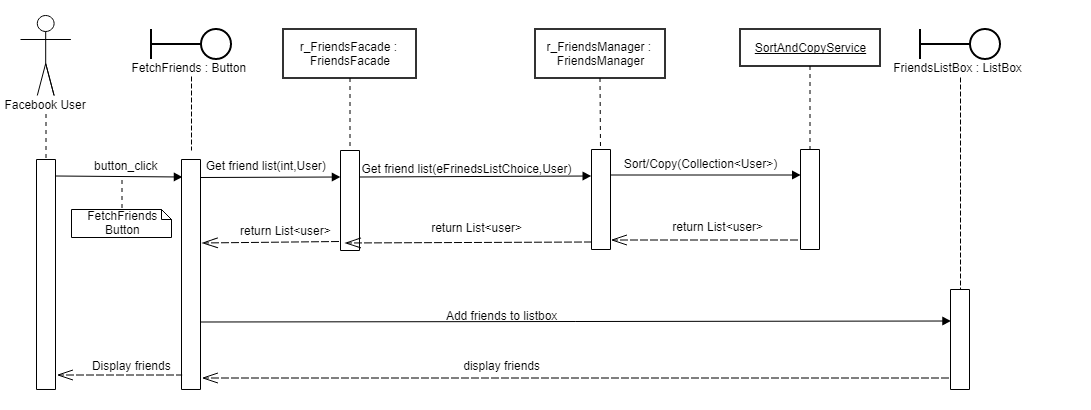
בודקת באיזה סדר המשתמש בחר להציג את רשימת החברים ומשתמשת ב-static class הנקרא "SortAndCopyService" אשר נועד לבצע את המיון על רשימת החברים לפי הסדר המותאם.

1. SortAndCopyService (internal class) – מחלקה האחראית על פעולות המיון / העתקה ממבנה נתונים אחד לאחר.

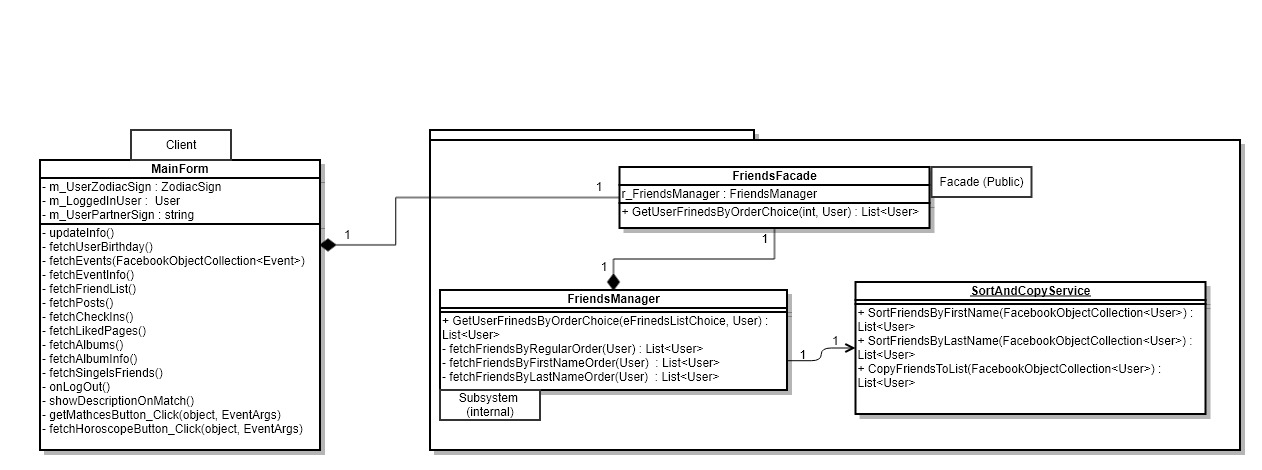
התבנית שאנחנו מימשנו הינה "שקופה". המחלקה היחידה שחשופה ללקוח הינה מחלקת ה-facade ואין ללקוח גישה למחלקות שמבצעות את החישובים בפועל, הוא אינן מכיר אותן כלל.

ניתן למצוא את המימוש תחת הפרויקט FriendsColntroller.

* Sequence Diagram



* Class Diagram



### תבנית מס' 3 – Proxy

* סיבת הבחירה / שימוש בתבנית:

באפליקציה אנו מציעים ממשק של רישומים על המפה. כלומר, כל הרישומים שהמשתמש ציין שהיה בהם יופיעו על מפה לפי ה-Location שהוא סיפק.

בנוסף המשתמש יכול לראות מידע על כל CheckIn שהוא ציין שהיה בו כמו: תמונה מהמקום, תיאור הרישום וכו'.

ניתן לנו מה-dll של facebook את המחלקה CheckIn – Legacy component אשר היינו צריכים לעטוף במחלקה שלנו בכדי לאפשר תיאום בין המפה בטופס (גם Legacy component) לבין ה-checkin.

בכדי להציג את הרישום על המפה, היה צריך להוסיף נתונים ל-checkin אשר לא היו לפני ואנחנו לא יכולים לשנות. לכן, היינו צריכים לייצר "מתחזה" למחלקה checkin תוך שמירה על האובייקט שנתון לנו בפועל. את הבעיה הנ"ל פותרת תבנית Proxy.

* אופן המימוש:

בנינו מחלקה בשם "CheckInWrapper" המשמשת בתור CheckIn.

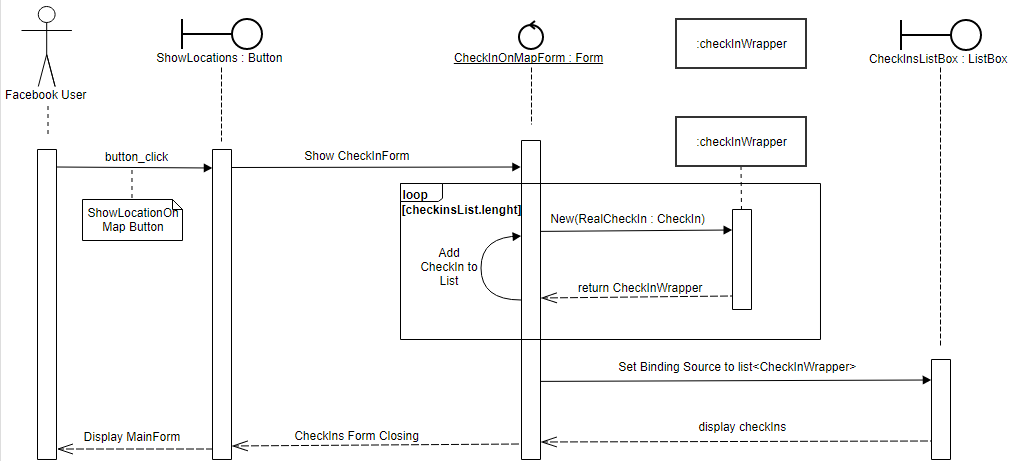
למחלקה הוספנו משתנים אשר היינו צריכים בכדי לייצר את ה-checkin על המפה, כמו: מיקום במפה ע"פ נקודות ציון, מרקר מבוקש על המפה.

המחלקה מחזיקה בקומפוזיציה את ה-CheckIn הנוכחי שצריך להיות מקושר אליה. את כל הנתונים שנצטרך שה-CheckIn מספק גם ה"מתחזה" מספק, רק שהוא דואג לשנות ערכי null לערכים ממשיים שבחרנו למימוש ולכן זה חוסך בדיקות אצל הלקוח.

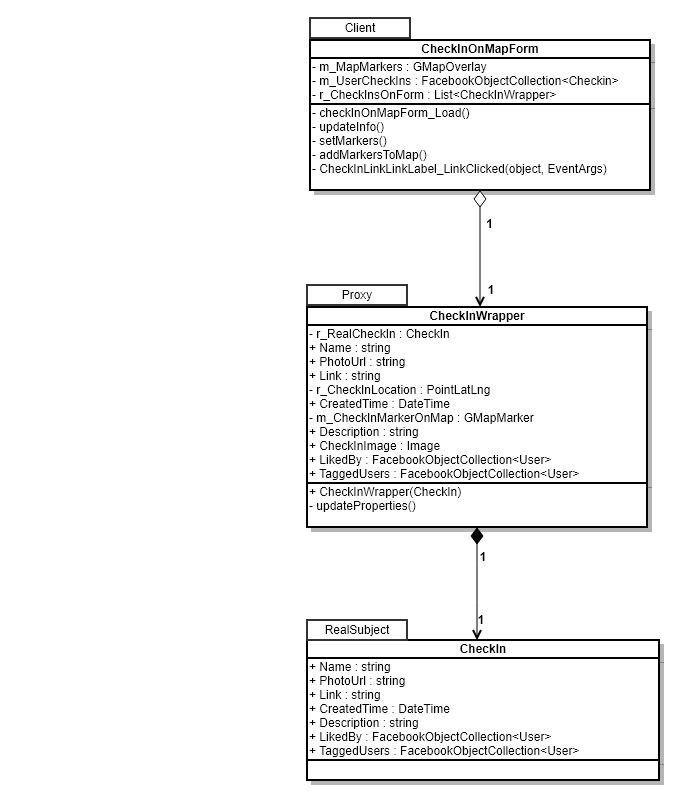
המחלקה מקבלת בבנאי את ה-CheckIn אליו היא צריכה להיות מחוברת, שומרת אותו ומאתחלת את ה-properties בהתאם.

ללקוח השימוש מול ה"מתחזה" שקוף לחלוטין.

* Sequence Diagram



* Class Diagram



במטלה זו ביצענו שימוש בפעולות אסינכרוניות ובdata binding :

**פעולות אסינכרוניות:**

במטלה הקודמת כאשר התבצעה טעינת הנתונים אל ה-Form, הוקפא החלון עד אשר הטעינה הסתיימה

והמשתמש לא יכול היה לעשות כלום בזמן זה.

על מנת לאפשר לו להמשיך לעבוד בעת טעינת הנתונים אל הUI השתמשנו ב-Thread נפרד בכל מצב בו ביצענו פעולות שינוי ב-Form.

למשל, כאשר ה-MainForm נטען, מתבצעות מספר פעולות – הבאת נתונים, הצגת אירועים וכו'

ה-MainThread ממשיך לרוץ וברקע אנו מריצים Thread-ים אחרים אשר מסייעים לפעולת הבאת הנתונים.

פעולה זו חסכה המתנה ארוכה של תצוגה שהייתה במטלה 1.

שימוש בעבודה אסינכרונית בקוד נעשה ב:

MainForm:

* UpdateInfo
* fetchPosts
* fetchFriendList
* fetchEvents
* fetchLikedPages
* fetchAlbums
* fetchDailyHoroscope
* showDescriptionOnMatch

PlaylistForm

* fetchUserLinkes

המימוש נעשה ע"י הפרדה ל-Thread חדש נפרד מה-MainThread. כך ה-Form ממשיך להתקיים מבלי לדאוג לפעולות החישוב "מאחורי הקלעים" וניתן לראות בבירור שנחסך זמן בעליית המסך.

השימוש בתהליכונים נפרדים נעשה בכדי לזרז את פעולות ממשק המשתמש ולייצר מצב בו המשתמש מחכה פחות זמן לבקשות שהוא מבקש מה-UI.

CheckInOnMapForm

* updateInfo

**שימוש בData Binding:**

* הפקד EventsListBox עושה שימוש ב- data binding.

הפקד משתמש ב-EventBindingSource כ-DataSource המקושר לאירועים של המשתמש לפי ה-radioButton.

בעת לחיצה על אירוע מהרשימה יוצגו פרטי האירוע מצד שמאל של הרשימה.

* הפקד AlbumsListBox משתמש ב-AlbumsBindingSource כ-DataSource.
* הפקד checkInsListBox משתמש ב-CheckInWrapperBindingSource כ-DataSource. (נמצא ב-CheckInOnMapForm).