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

* פיצ'ר מזג אוויר: מביא אינפורמציה על מזג האוויר, דרך API בשימוש בJson.
* פיצ'ר מניות מביא אינפורמציה על מניות , דרך API בשימוש בJson.
* יצירה\עריכת פוסטים מוכנים מראש:

המשתמש יכול ליצור פוסטים מוכנים מראש ב editor ,לערוך אותם,למחוק, להשתמש בהם כtemplate ולפרסם בפוסט. ההודעות המתאימות ישמרו על קובץ מקומי במחשב.

* פיצ'ר לסידור פוסטים שפורסמו לפי הא"ב.
* **יצירת יום הולדת לחבר (חדש**): אפשרות ליצור יום הולדת לחבר קרוב ויום הולדת לחבר רחוק על פי בחירת המשתמש. (פרסום הודעת מזל טוב ואלבום תמונות משותף – עבור חבר קרוב ועבור חבר רחוק יהיו 2 פוסטים שונים . עבור החבר הקרוב יותר תכתב ברכה אישית יותר, ועבור החבר הרחוק תכתב הודעה שאינה אישית). בנוסף הופסנו באיזור החברים באפליקציה לוח שנה משופר המשמש להצגת יום ההולדת הלועזי והעברי של החבר שנבחר מרשימת החברים בlistbox . לוח שנה זה מאפשר קבלת אינפורמציה אודות תאריכים עבריים על ידי בחירת תאריך מסוים. לועזי בלוח).
* הוספנו לפרוייקט אופציית Dark Mode כאשר המשתמש מסמן ב"צ'ק בוקס" אופציית זאת המסך הופך לכהה יותר

### תבנית מס' 1 – Stradegy

* סיבת הבחירה / שימוש בתבנית:  
  השתמשנו בתבנית זאת על מנת לאפשר "הזרקת קוד" בזמן ריצה בשמירת קובץ הגדרות המשתמש . המשתמש בתבנית העיצוב ,לשנות את האלגוריתם של שמירת הקובץ של הגדרות המשתמש באפליקצייה,מבלי לשנות דבר במחלקת המשתמש .

נוכל לשנות את האלגוריתם ואת השימוש מבלי לשנות דבר בקוד המשתמש , דבר המשפר את התחזוקתיות והreusable של המחלקה. לדוגמא: כעת ה"אסטרטגייה" היא בעזרת שמירה באמצעות xml , אם בעתיד נרצה שימוש בדרך אחרת לשירת קבצים כל מה שנצטרך לעשות זה לממש בנפרד את האופציה .

בנוסף יצרנו הפרדה מוחלטת כאשר המשתמש לא מכיר את ה"פנימיות" של האסטרטגייה ובוודאי שהאסטרטגייה לא מכירה את המשתמש ,כלומר נוכל להשתמש באסטרטגייה שמירת קובץ גם אם נצטרך שימוש נוסף בשמירת קבצים .

Reuse לאסטרטגייה הקונקרטית של שמירה ב xml

* אופן המימוש:   
  השחקנים: IFileStradegy (אינטרפייס כללי של שמירת קבצים) ,XmlFileUtilsStradegy (מממש את האינטרפייס של ה ifilestradegy, ומבצע קונקרטית את "אסטרטגיית" המימוש של שמירה וload באמצעות (xml

Settings – המשתמש ,הוא זה שמשתמש באסטרטגיית שמירת והעלאת הקבצים. הוא קיבל reference כללי לממשק של IFileStradegy ורק בזמן ריצה המשתמש בloginFacade יחליט באיזו אסטרטגייה ישתמש, כרגע משתמש ב XMLFileStradegy

### תבנית מס' 2 - Proxy

* סיבת הבחירה / שימוש בתבנית:

בחרנו להשתמש בתבנית זאת כפרוקסי יורש (class proxy) עבור הפיצ'ר של הלוח שנה העברי.

הלוח שנה עברי מתנהג כמו לוח שנה רגיל (גרגוריאני) אך עם התוספת של הלוח השנה העברי. הלוח השנה העברי "מתחקה" לreal subject שהוא לוח שנה חודשי רגיל.

* אופן המימוש:  
  ה"שחקנים" בתבנית הזו : real subject ) MonthCalendar שיורש מcontrol). והפרוקסי הוא MonthCalendarProxy שיורש מה.real subject המשתמש רוצה להציג תאריך יום הולדת בעת בחירת יום בלוח שנה ( המצב שקורה בפועל הוא מתנהג כמו MonthCalender אך מציג לו את התאריך העברי של יום ההולדת. במידה ובחר יום בלוח השנה תתבצע הצגה של התאריך העברי אשר רלוונטי לאותו היום).





### תבנית מס' 3 – Builder

* הפיצ'ר של "פרסום ברכת יום הולדת ויצירת אלבום עושה שימוש בתבנית זו. הסיבה שהשתמשנו בפיצ'ר זה משום שהוא מורכב מכמה חלקים ויש חשיבות לסדר הפעולות. אם נרצה בעתיד להוסיף עוד משימות לבנייה,נוכל לשמור על רצף לוגי נכון ולוגיקת הבנייה מתבצעת ברכיב ייעודי שמיועד לכך. לכל סוג יום הולדת נוכל ליצור בילדר ייחודי ופולימורפי ולהרחיב את הפעילות בעתיד.

* אופן המימוש:

השחקנים: BirthDayManager אשר בתפקיד ה Director ומחזיק בקומפוזיציה Interface של BirthdayActivityBuilder.

הדיירקטור אחראי על סיפוק פעילות יום ההולדת על ידי שימוש בבילדר המתאים לפי בחירת המשתמש. הבנייה תתבצע בסדר של בניית סטטוס מזל טוב ולאחר מכן בניית אלבום משותף. הדיירקטור עושה שימוש בבילדר ואינו אחראי על הפעולות השונות אלא רק על הסדר אשר יתבצעו. כך שאם נרצה בעתיד לשנות לוגיקה של אחת מפעולותיו של הבילדר נעשה זאת דרך הבילדר המתאים.

BirthdayActivity - זהו האובייקט שעלינו ליצור ולקבל.

IBirthDayActivityBuilder - Interface אשר מצריך מימוש של שתי מתודות :   
 BuildStatus - פרסום סטטוס "מזל טוב" על ה"קיר" של החבר.  
BuildAlbum - בניית האלבום המשותף.

את IBirthDayActivityBuilder ממשים שני Concreate Builders ובונים את האלבום והסטטוס בצורה היחודית להם:  
CloseFriendBirthdayActivityBuilder - עבור החבר הקרוב יותר תכתב ברכה אישית יותר ואלבום אישי יותר.  
וFarFriendBirthdayBuilder - ועבור החבר הרחוק תכתב הודעה שאינה אישית ואלבום שאינו אישי.





### עבודה אסינכרונית

בתוכנית שלנו השתמשנו בעבודה אסינכרונית בכמה שימושים:

בשליפת מידע מהפייסבוק של המשתמש בתוכנית בכלל הפיצ'רים:

Fetch Albums

Fetch Groups

Fetch Friends

Fetch Liked Pages

Fetch Posts

ע"י יצירת thread חדש לכל אחת מהמתודות אשר שולפות את המידע

וInvoke עבור פעולות של הפקדים שלא מאפשרים גישה עבור Thread שהוא לא ה Main Thread שיצר אותם.

הסיבה שהשתמשנו בעבודה אסינכרונית עבור פעולות אלא,משום שלא רצינו "לתקוע" את הUI עבור המשתמש כאשר אנחנו מבקשים שליפה משרת הפייסבוק, ושהשתמש יוכל לעשות עוד פעולות עד ש הUI יוצג עבורו.

### עבודה עם Data Binding

בתוכנית שלנו השתמשנו בdata binding עבור הצגת הקבוצות של המשתמש והתמונת נושא של הקבוצה.

לצורך כך השתמשנו בהצגת הקבוצות של המשתמש.

השתמשנו ב DataBindingWizard בצורה אוטומטית כך שבחרנו להציג את התמונה של המשתמש כאשר הוא בוחר בlistbox של הקבוצות את הקבוצה.

השורת קוד היחידה שכתבנו הייתה במתודה fetchGroups והיא: groupBindingSource.DataSource = LoginFacade.LoginUser.Groups;

על מנת לקשר בין הdata source והנתונים של הקבוצה של המשתמש.

בנוסף גם בfetchFriends השמשנו בזה.