***רשימת בונוסים בפרוייקט:***הוספת תמיכה (בכל השכבות) בתמונות המוצרים

שלב 1:

* **לבונוס** :לצורך המרות של מספרים ותאריכים מהקלט - להשתמש בפונקציה TryParse- עשינו בונוס זה.
* **לבונוס**: בתוכניות בדיקה של שכבת נתונים - שימוש במתודה\-ות גנרית\-יות  הרחבה עם reflection על מנת להדפיס את האובייקטים של ישויות הנתונים - עשינו זאת, פונקציית ההרחבה הזו נקרא אצלנו ToStringProperty ונמצאת במחקלת Tools תחת ה namespace DO .
* מחיקה לוגית (סימון עם שדה נוסף "פעיל" או דומה), ניהול אובייקטים שנמחקו ואפשרות לשחזר (סטודנטים)

הוספנו בישויות הלוגיות של שכבת הנתונים שדה נוסף- IsDeleted מסוג bool המשמש מעין דגל להגיד האם הישות "נמחקה" או שהיא קיימת.  
היתרון בכך הוא שהמידע לא באמת נמחק ואם יש צורך לשחזר נתונים זה אפשרי.

* עשינו את מחלקת dataSorce כסינגלטון שהוא Thread Safe ועם Lazy Initialization מירבי.

שלב 2:

* **לבונוס**: ב-BL - שימוש במתודה\-ות גנרית\-יות  הרחבה עם reflection על מנת להעתיק נתונים בין אובייקטים של ישויות נתונים  ושל ישויות לוגיות עבור תכונות עם אותו שם ואותו טיפוס-עשינו בונוס זה , פונקציה זו נקראית אצלנו:

public static Target CopyPropTo<Source, Target>(this Source source, Target target).

* **לבונוס**: בתוכניות בדיקה של השכבה לוגית - שימוש במתודה\-ות גנרית\-יות  הרחבה עם reflection על מנת להדפיס את האובייקטים של הישויות הלוגיות –

וגם

public static Target CopyPropToStruct<Source, Target>(this Source source, Target target) where Target : struct

וגם עבור רשימה

public static IEnumerable<Target> CopyListTo<Source, Target>(this IEnumerable< Source > sources) where Target : new()

* **לבונוס**: כל שימוש מיוחד ויצירתי בדלגטים, מתודות הרחבה, reflection, סוגים\טכנולוגיות שלא נלמדו (למשל( RegExp) בכל מקום מתאים המאפשר לשמור על רמה נמוכה של מורכבות מיותרת בקוד ורמה גבוהה של בהירות הקוד

-השתמשנו במספר פונקציות הרחבה (הנמצאות במחלקת Tools תחת namespace BlApi) כגון:

1. calculateStatus – פונקציית הרחבה עבור הזמנה - חישוב סטטוס להזמנה.
2. CalculateAmountItems – פונקציית הרחבה עבור הזמנה-חישוב מספר פריטים בהזמנה בעזרת מתודות הרחבה של המערכת כגון .sum()
3. CalculatePriceOfAllItems – פונקציית הרחבה עבור הזמנה-חישוב עלות ההזמנה

ועוד רבים וטובים שלא נפרט כאן....

* השתמשנו בשביל בדיקת תקינות קלט כתובת האימייל ב RegularExpressions , עשינו זאת כפונקציית הרחבה ל- string , פונקציה זו נקראת אצלנו: IsValidEmail, והיא משתמשת ב Regex.Replace()-
* שימוש ב BLTest ב- string.Join()
* מחיקה לוגית (סימון עם שדה נוסף "פעיל" או דומה), ניהול אובייקטים שנמחקו ואפשרות לשחזר (סטודנטים)

הוספנו בישויות הלוגיות של שכבת הנתונים שדה נוסף- IsDeleted מסוג bool המשמש מעין דגל להגיד האם הישות "נמחקה" או שהיא קיימת.  
היתרון בכך הוא שהמידע לא באמת נמחק ואם יש צורך לשחזר נתונים זה אפשרי.

* הוספת אופציה למחיקת הזמנה שכבר הקלטה במערכת

void CancleOrder\_forM(int id);

שלב 3:

* השתמשנו בstyle עבור פקדים חוזרים כמו תיבות טקסט כותרות וכפתורים

שלב 4:

* לבונוס - ניתן לעשות סינגלטון שיהיה Thread Safe ועם Lazy Initialization מירבי –

עשינו זאת בשלב זה עבור מחלקת dal-List.

שלב 5:

* שימוש בישויות po לכל אורך שכבת ה- pl ,(תוך שמירה על הפרדה בין השכבות )ישויות שהשדות שלהם הן dip prop
* שימוש רב ב-biding עבור רשימות,תיבות טקסט,וכו' וכן שימוש נרחב ב converters
* שימוש ב-list view עבור קטלוג המוצרים(כולל תמונה)
* שימוש ב data template עבור תבניות חוזרות ברשימות וכו'..כמו כפתורים, stack panel ועוד.
* טעינת קבצי material designin ושימוש נרחב בעיצובים שלהם כגון אייקונים,עיצוב כפתורים ועוד.. בכדי לשדרג את רמת העיצוב בפרוייקט.
* הןספת אופציה לשמירת ספרים אהובים
* כניסת מנהל עם סיסמא (הסיסמא היא: giveus100)
* הוספת פונקציות שמגבילות קלט בעזרת regex (למספרים וכו')
* שימוש בreletive source במספר מקומות
* סינון מוצרים לפי קטגוריה בעזרת שימוש בגרופינג
* לאחר בחירת קטגוריה אפשרות לסינון לפי מחיר מהנמוך לגבוה או להפך
* חיפוש ספר בעזרת הקלדה חופשית שבודקת הכלה של שם הספר
* עבור המנהל יש קיבוץ לפי קטגוריות של הספר
* עבור המנהל יש חיפוש הזמנה חופשי לפי שם או מספר הזמנה במקביל
* ארכיון עבור כל הישויות שנמחקו במהלך התכנית
* יצירת פקד בשם NumericUpDownControl שמאפשר הגדלת והקטנת מספר מוצרים בהזמנה שמקושרים לכמויות הנכונות של התכונה InStoke של המוצר

שלב 7:

* עדכון דינמית תצוגת חלונות פתוחים של רשימת הזמנות והזמנה (מנהל), של מעקב הזמנה (לקוח\קונה) בעזרת שימוש בתבנית אובזרבר ופועל רקע (**עד 3 נק'**)
  + ה-observable אלו אירועים בשכבת BL - עדכון\הוספה\[מחיקה] של הזמנות
* עדכון כל שניה על התקדמות מעבר ההזמנה ממצב למצב בעזרת progress bar שכלל אינטראקציה עם סימולטור
* שימוש בsound לאחר הצלחת הסימולטור