קורס בדוטנט תשע"ז
מיני פרויקט

מבוא

במהלך הקורס אנו נכתוב פרויקט שבאמצעותו נתרגל את עקרונות שפת #C, עקרונות ארכיטקטורה של תוכנה, ויצירת ממשקי משתמש באמצעות תשתית הפתוח המודרנית WPF.
הערה: עבודה פושרת ולא יצירתית תגרור בעקבותיה גם ציון פושר. יש להדגיש שחלק נכבד מן המבחן הסופי יתבסס על נושאים תכנותיים שתתבקשו להתמודד איתם במהלך התרגול. חשוב להדגיש שציון מתחת ל-85 בקורס זה כמו בכל קורס מעשי אחר איננו משמעותי בעיניהם של המעסיקים בתעשייה...

העבודה תתבצע אך ורק בזוגות ! לא בבודדים ולא בשלישיות ! כל אחד מהשותפים חייב להיות שותף מלא בכל אחד מן השלבים ! הפרויקט חייב לרוץ על מחשב של כל אחד מן השותפים. לא תתקבל טענה ששותף אחד עשה שלב מסוים ואלו שותף אחר עשה את החלק האחר ! במקרה כזה הציון יהיה פרופורציונלי למספר השלבים שנעשו ע"י כל אחד מן השותפים... לא ניתן לעבור לשותף אחר במהלך הסמסטר. לא ניתן לעבור במהלך הסמסטר מקבוצה לקבוצה. לא ניתן להיות שותף עם חבר שרשום בקבוצה שונה. בכל אחד מן המקרים האלה התרגיל ייפסל.

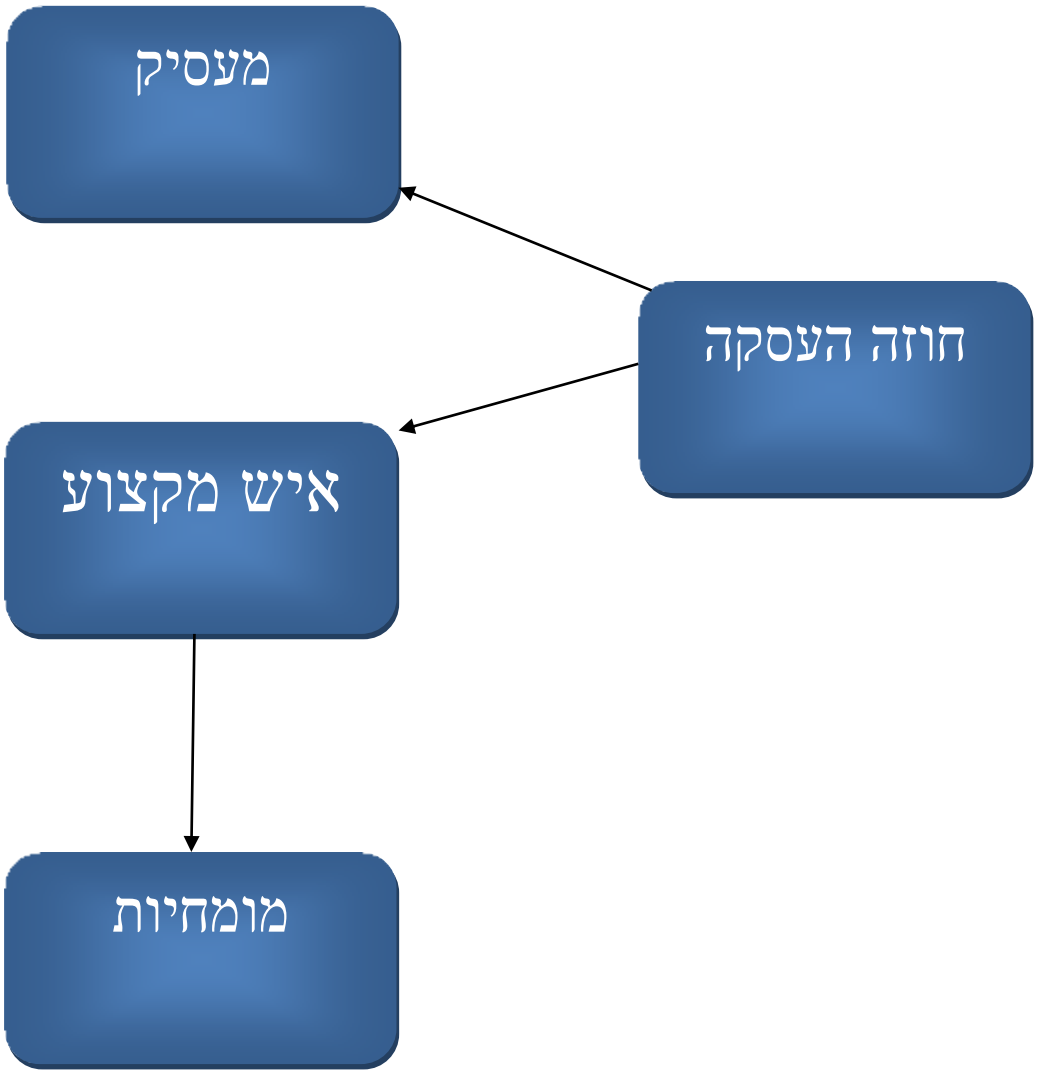
הבדיקה תיעשה בשלשה מועדים בלבד. כל שלב הגשה יימסר בתאריך היעד. השלבים ייבדקו בצורה כללית (בעיקר תיבדק הרצה תקינה, המרצה ישמור בקובץ אקסל הערות לגבי כל שלב כולל הערות איכות עבודה ועמידה מדויקת בלוח הזמנים הנדרש) ורק לבסוף עם הגשת השלב האחרון והמסכם תתבצע בדיקה יסודית של העבודה שתכלול גם הגנה של שני השותפים על העבודה כ"א בנפרד. הציון בהגנה יהיה שבר בין 0..1 ויוכפל בציון הכללי שישוקלל מבדיקת כל השלבים. הציונים הסופיים עבור כל אחד מהשלבים יינתנו רק בסוף הסמסטר. ציון המעבדה ישוקלל עם ציון המבחן רק בתנאי שהציון במבחן הוא לפחות 60%.

הנחיות להגשה

העבודה בזוגות, יש להעלות קובץ zip עם הפתרון למודל (ראה הנחיות בתרגיל המבוא)
שם הקובץ יהיה dotNet5777_Project01_xxxx_yyyy
dotNet5777 = זה הקורס והשנה העברית
01 = זה השלב (1,2,3)
xxxx = זה 4 ספרות אחרונות בתעודת הזהות של בן הזוג הראשון
yyyy = זה 4 ספרות אחרונות בתעודת הזהות של בן הזוג השני
נא להקפיד על פורמט זה על מנת למנוע מצב של אי קבלת ציון על תרגיל מסוים.

תיאור הארגון

בפרויקט זה נכתוב מערכת (חלקית בלבד) לניהול חברת כח אדם למקצועות ההייטק. חברת כח האדם מנהלת אתר אינטרנטי, שבאמצעותו כל איש מקצוע יכול להציע את שירותיו (כלומר את תחומי מומחיותו). שמות אנשי המקצוע והתמחותם נגיש לקהל מעסיקים . המערכת מאפשרת למעסיק ליצור קשר עם איש מקצוע, לראיין אותו וכן לחתום אתו חוזה העסקה.
אמנם, את הפרויקט הזה עדיין לא נכתוב כאתר אינטרנטי בתבנית של שרת\לקוח (מפני שזה חורג מנושאי הקורס) אך נכתוב את רוב השכבות בצורה כזאת שהרוב יהיה חופף למערכת אינטרנטית מקבילה.



תיאור ישויות המערכת

מומחיות (Specialty)
מכילה פרטים כגון: מספר ייחודי לזיהוי המומחיות, שם התחום (enum), שם המומחיות, תעריף מינימלי מומלץ לשעת עבודה במומחיות זו, תעריף מקסימלי מומלץ לשעת עבודה במומחיות זו.

איש מקצוע
מכילה פרטים כגון: מספר ת.ז., שם משפחה, שם פרטי, תאריך לידה, טלפון, כתובת, תואר, מספר שנות ניסיון, המלצות כשדה טקסטואלי, מזהה מומחיות פרטי חשבון בנק (שם הבנק, סניף, עיר, מספר חשבון)

מעסיק
מכילה פרטים כגון: שדה בוליאני שיציין האם המעסיק הוא אדם פרטי או חברה בע"מ,., מספר חברה, שם משפחה, שם פרטי, שם חברה – אם יש, טלפון, כתובת, שם תחום (enum), תאריך הקמת העסק

חוזה העסקה
ישות המקשרת בין המעסיק לאיש-המקצוע ומכילה את הפרטים הבאים: מספר חוזה העסקה, מספר ת.ז. של מעסיק, מספר ת.ז. של איש-המקצוע, האם נחתם חוזה העסקה (שדה בוליאני)?, השכר לשעה ברוטו (לפני עמלה של חב' כח אדם, השכר לשעה נטו (אחרי עמלה של חב' כח אדם) תאריך תחילת העסקה, תאריך סוף העסקה

הערה: הישות חוזה העסקה **תחבר** בין המעסיק לאיש-המקצוע. מאחר ומעסיק יכול להעסיק הרבה אנשי מקצוע ואיש מקצוע יכול להיות מועסק אצל הרבה מעסיקים, לכן יש כאן רשימה מקשרת של NXM.

תיאור חלקי הפרויקט בקצרה

חלק ראשון:
נבנה את הפרויקט במודל השכבות, מקור הנתונים יופיע כפרויקט נפרד ויכיל רשימות של אובייקטים. הגישה למקור הנתונים תתבצע באמצעות שכבת ה- DAL וממשק המשתמש יהיה ממשק ויזואלי גרפי. תשתית הפתוח הנדרשת של הממשק הוויזואלי הינה WPF.

חלק שני:
נשנה את שכבת ה- DAL שייגש למקור נתונים שהפעם יתבסס על קבצי xml , שם נעבוד עם האובייקטים שלנו וכן עם אתר חיצוני המספק ערכי xml אמיתיים אותם נמיר לפרויקט שלנו.

חלק א – הגדרת הפרויקטים עבור מודל השכבות ומימושן

בחלק זה של הפרויקט נגדיר את כל אחת מן השכבות שיהוו יחד את המיני פרויקט שיאפיין את חברת כח האדם.

לחלק זה 4 שלבים כדלהלן:

בשלב א' נגדיר את המחלקות של הישויות, כלומר את שכבת ה- BE.
בשלב ב' נגדיר את דרך העבודה מול מקור הנתונים, כלומר את שכבת ה- DAL.
בשלב ג' נגדיר את הלוגיקה של המיני פרויקט. כלומר את שכבת ה- BL
בשלב ד' בחלק זה נממש את ממשק המשתמש של המיני פרויקט, כלומר שכבת ה- UI.

שלב א - ישויות

א. הגדר פרויקט **מסוג Class Library** בשם **BE** ובו תגדיר את המחלקות הבאות. חובה להגדיר כל מחלקה בקובץ נפרד.

אנו נפרט שדות שיכללו במחלקות. כל אחד מוזמן **וחייב** להוסיף שדות כפי שנראה לו מתאים.

מבנה המתאר חשבון בנק המכיל:

- מספר בנק
- שם הבנק
- מספר סניף
- כתובת הסניף
- עיר הסניף
- מספר חשבון

מחלקה בשם: Specialization שתייצג מומחיות ותכלול:

א. מאפיין שמציין מספר מומחיות. **(קוד רץ ייחודי בן 8 ספרות)**
ב. מאפיין enum שמציין את שם התחום.(בסיסי נתונים, תקשורת, אבטחת מידע, תכנות צד שרת, תכנות מובייל, עיצוב ממשקי משתמש...)
ג. מאפיין שמציין את שם המומחיות
ד. מאפיין שמציין את התעריף המינימלי עבור שעת עבודה במומחיות זו.
ה. מאפיין שמציין את התעריף המקסימלי עבור שעת עבודה במומחיות זו.
ו. מאפיינים נוספים לפי הצורך.
ז. *ToString*

מחלקה בשם: Employee שמייצגת איש מקצוע ותכלול:

א. מאפיין שמציין את מספר ת.ז. של האיש מקצוע.
ב. מאפיין שמציין את שם המשפחה של איש המקצוע.
ג. מאפיין שמציין את השם הפרטי של איש המקצוע
ד. מאפיין שמציין את תאריך הלידה של איש המקצוע
ה. מאפיין שמציין את הטלפון של איש המקצוע.
ו. מאפיין שמציין את הכתובת של איש המקצוע
ז. מאפיין enum שמציין את התואר של איש המקצוע (תעודה בלבד, תואר ראשון, תואר שני, תואר שלישי, סטודנט)
ח. שדה בוליאני המציין איש המקצוע הוא בוגר צבא.
ט. מאפיין המכיל את פרטי חשבון הבנק של איש המקצוע (מסוג המבנה שהוגדר לעיל)
י. מאפיין שמכיל מזהה של המומחיות (כמו המספר מזהה במומחיות).
יא. מאפיינים נוספים לפי הצורך.
יב. *ToString*.

מחלקה בשם Employer שמייצגת מעסיק ותכלול:

א. מאפיין שמציין מספר מזהה של החברה. (אם הוא אדם פרטי זה יהיה הת"ז שלו)
ב. מאפיין בוליאני שיציין האם המעסיק הוא אדם פרטי או חברה בע"מ.
ג. מאפיין שיציין את שם משפחת המעסיק.
ד. מאפיין שיציין את השם הפרטי של המעסיק
ה. מאפיין שיציין את שם החברה (יכול להיות ריק אם הוא אדם פרטי)
ו. מאפיין שיציין את מספר הטלפון של המעסיק.
ז. מאפיין שמציין את הכתובת של המעסיק.
ח. מאפיין enum שיציין את תחום ההתעסקות של המעסיק, מקביל לתחום שיש במומחיות.
ט. מאפיין שמציין את תאריך הקמת העסק.
י. מאפיינים נוספים לפי הצורך.
יא. *ToString*.

מחלקה בשם: Contract - שמייצגת חוזה העסקה (כלומר את הקשר בין איש המקצוע למעסיק) ותכלול:

א. מאפיין שמציין את מספר חוזה העסקה. **(קוד רץ ייחודי בן 8 ספרות)**
ב. מאפיין שמציין מספר מזהה של המעסיק.(חייב להופיע כזה)
ג. מאפיין שמציין מספר מזהה של המועסק.(חייב להופיע כזה)
ד. מאפיין בוליאני שמציין האם נערך ראיון ? (כן/לא)
ה. מאפיין בוליאני שמציין האם נחתם חוזה העסקה ? (כן/לא)
ו. מאפיין שמציין את השכר לשעת עבודה ברוטו. (לפני עמלה של חב' כח אדם, לדוגמה עמלה של - 10%)
ז. מאפיין שמציין את השכר לשעת עבודה נטו. (לשים לב לבדוק את הקשר לשכר המצוין עבור המומחיות אצל המועסק בשכבה הרלוונטית)
ח. מאפיין שמציין את תאריך תחילת ההעסקה.
ט. מאפיין שמציין את תאריך סוף ההעסקה.
י. מאפיינים נוספים לפי הצורך.
יא. מאפיין שמציין את מספר שעות ההעסקה בחוזה.
יב. *ToString*

הערה: כל תקלה, כתוצאה מפעולה כלשהי תגרור אחריה טיפול בחריגות לפי מנגנון החריגות הקיים ב- #C. לדוגמא: ייתכן שבחוזה העסקה יצוין שכר לשעה שאיננו נמצא בטווח השכר עבור התמחות ספציפית זאת.

<u>הערות:</u>

- יש להוסיף שדות ומאפיינים במידת הצורך.
- **יש להימנע משימוש בפונקציות public במחלקות (מלבד הtostring)**, הלוגיקה המרכזית מתבצעת בשכבת ה- BL כפי שיפורט בהמשך.

שלב ב' – DAL במימוש רשימות

הוסף פרויקט מסוג class library בשם DAL ובו בצע את הפעולות הבאות:

א. הגדר (interface) ממשק בשם Idal .
בממשק הנ"ל הגדר חתימה של פונקציות שימושיות עבור האפליקציה כגון:

- הוספת מומחיות.
- מחיקת מומחיות.
- עדכון פרטי מומחיות קיימת.

- הוספת איש מקצוע.
- מחיקת איש מקצוע.
- עדכון איש מקצוע קיים.

- הוספת מעסיק.
- מחיקת מעסיק.
- עדכון מעסיק.

- הוספת חוזה העסקה
- עדכון חוזה העסקה
- מחיקת חוזה העסקה

- קבלת רשימת כל המומחיות.
- קבלת רשימת כל אנשי המקצוע.
- קבלת רשימת כל המעסיקים.
- קבלת רשימת כל החוזי העסקה.

- קבלת רשימת כל סניפי הבנק הקיימים בארץ (מחזיר אוסף של סניפים מסוג המבנה המתאים שהוגדר ב-BE)

ב. צור פרויקט חדש בשם: DS והגדר בתוכו מחלקה בשם DataSource שתכיל את הנתונים (רשימות) של הישויות שלנו.
מחלקה זו תכיל 4 רשימות סטטיות של המחלקות הנמצאות ב- BE.

ג. הגדר מחלקה בשם: Dal_imp אשר מממשת את ממשק ה- Idal הנ"ל.
הפונקציות של המחלקה הזאת יעבדו מול האוספים מסוג list<> הנמצאים במחלקה DataSource.

ד. ממש את הפונקציה שמחזירה את רשימת הסניפים
בשלב זה ממש כך שהיא תחזיר רשימה של 5 סניפים שתגדיר בתוך הפונקציה.
הערה: בחלק הבא של הפרויקט הפונקציה תחזיר את המידע על הסניפים בארץ ישירות מהמידע שמופיע בבנק ישראל ולכן אין צורך להמציא יותר מ 5 סניפים בשלב זה.
(מיותר לציין שבניגוד לפונקציונאליות עבור שאר במחלקות, במחלקה סניף נרצה רק לקבל סניפים קיימים ממאגר המידע שלנו ולא להוסיף, לעדכן או למחוק סניפים)

הערה:
שכבת ה- DAL היא האחראית על כך שהמפתח בכל אחת מהמחלקות יהיה קביל, כך שבמקרה שהמספר הייחודי (שיתקבל בישות שמעוניינים להוסיף) הוא 0 שכבת ה- dal אחראית לתת קוד ייחודי בעצמה, במידה והמספר הייחודי (שיתקבל בישות שמעוניינים להוסיף) שונה מ- 0 עליה לבדוק שאינו קיים כבר.
כמו כן בישויות חוזה העסקה והתמחות שכבת הDAL היא זו שמוסיפה לישות את המספר הרץ.

שלב ג' – שכבת ה- BL

הגדר ממשק (interface) בשם: IBL שבו החתימות של השיטות בדיוק כמו ב- IDAL

על שכבת ה BL לבצע את הלוגיקה הבאה:

- אין אפשרות להוסיף איש מקצוע מתחת לגיל 18
- הוספת חוזה העסקה תתאפשר רק ע"י בדיקה שאכן המעסיק הזה קיים במאגר המעסיקים, ושהעובד קיים במאגר העובדים
- אין אפשרות לחתום חוזה העסקה עם חברה חדשה שגילה קטן משנה.
- חישוב השכר נטו לשעה נעשה ע"פ נוסחה כלשהי שלוקחת בחשבון את מספר העסקות שיש כבר **לאיש מקצוע זה מטעם החברה**. (הרעיון הוא שעבור כל העסקה נוספת שאיש המקצוע הזה מועסק באמצעות חב' כח אדם אז שיעור העמלה קטן)
- חישוב השכר נטו לשעה נעשה ע"פ נוסחה כלשהי שלוקחת בחשבון את מספר **העסקות שיש כבר למעסיק זה מטעם החברה**. (הרעיון הוא שעבור כל העסקה נוספת שהחברה מעסיקה באמצעות חב' כח אדם אז שיעור העמלה קטן וזה בעצם מהווה צ'ופר לעובד וכמובן זה לטובת המעסיק)

ובנוסף להוסיף את חתימת הפונקציות הבאות:

- פונקציה שיכולה להחזיר את **כל חוזי ההעסקה** שמתאימים לתנאי מסוים (הכוונה שהפונקציה מקבלת delegate שמתאים לפונקציות שמקבלות חוזה העסקה ומחזירות bool וכך מוגדר התנאי)
- פונקציה שמחזירה את **מספר** חוזי ההעסקה שמתאימים לתנאי מסוים (הכוונה שהפונקציה מקבלת delegate שמתאים לפונקציות שמקבלות חוזה העסקה ומחזירות bool וכך מוגדר התנאי)

הגדר פונקציות המחזירות את הקבוצות הבאות (ע"י שימוש ב-Grouping)
כל אחת מהפונקציות הבאות תקבל משתנה בוליאני שיציין האם להחזיר את התוצאות בצורה ממוינת (איך למיין זה לפי שיקול דעתכם) ערך ברירת מחדל של פרמטר זה יהיה false (לא ממוין)

- חוזים של חב' כח האדם מקובצים (Grouping) ע"פ סוגי התמחויות.
- חוזים של חב' כח האדם מקובצים (Grouping) ע"פ אזור מגורים.
- רווחים של חב' כח האדם מקובצים (Grouping) ע"פ תקופות זמן, לשיקול דעתכם האם להגדיר תקופת זמן לפי חודשים, שנים, ימים וכו.. בכל מקרה יהיה ערך מספרי אחד בהתאם לכל אחת מהקבוצות של הזמן שחילקתם.

חובה לעשות שימוש בכישרון היצירתיות שלכם ולהוסיף לפחות עוד 6 פונקציות שמתאימות להיות בשכבת ה-BL ועובדות על הנתונים המוחזרים מה- DAL.
הגדר מחלקה חדשה בשכבת ה- BL שתממש את הממשק IBL הנ"ל המימוש יכלול שימוש ב Linq to object וביטויי למבדא.

יש לעשות שימוש בלפחות 4 ביטויי Linq בשכבת ה- BL- וה- DAL כל אחת. יש לעשות שימוש בלפחות 4 ביטויי למבדה בנוסף לביטויי Linq דלעיל.

הערות:

על שכבה זו לוודא שמתקיימים חוקים לוגיים בסיסים כמו לדוגמה:

- אם מוסיפים חוזה העסקה מסוים אזי יש לוודא שאיש המקצוע וכן המעסיק שמוכלים בחוזה העסקה אכן קיימים.
- יש לוודא שפרטי חשבון של איש מקצוע מכילים פרטי בנק קיימים
- וכו...

כאן תתבצע ההגנה הראשונה – אפשר בממשק קונסול או גרפי פשוט.

לשם הבדיקה של הדברים אנו ניצור פרויקט זמני בשם PL.
תוכל לממש אותו או בעזרת console application או ממשק גרפי WPF פשוט . אפשר בשלב זה לעשות הכל בחלון אחד.
בכל מקרה עליך לקרוא לפונקציות הנמצאות ב- BL ולבדוק אותן.

הערה:

כדי לעבוד בשלבים מומלץ בהתחלה ליצור משהו שבודק את ה- BE אח"כ את הפונקציונליות ב- DAL כדי לראות שאכן זה עובד ואז לחבר את ה- BL.

שלב ד' – שכבת ה- UI

בשלב זה ניצור ממשק גרפי לפרויקט באמצעות תשתית הפתוח WPF.

ניצור פרויקט חדש מסוג : WPF
ונקרא לו **PLWPF**,.
כמובן שיהיה עליך ליצור לו reference מתאים ל-BL וכן ל- BE .
עליך להגדיר ישות בשם: BL מטיפוס: IBL כפי שראית בדוגמה בהרצאה. לדוגמא:

```
IBL bl
 public Window1()
{
 bl = FactoryBL.getBL();
}
```

למעשה עליך לתכנן את המסכים אשר יקראו ויאפשרו למשתמש לגשת לפונקציונאליות הנמצאת ב- BL.

עליך לתכנן מסך לכל ישות בדומה למה שראינו בכיתה עם "הסטודנטים והקורסים", כאשר יהיו לפחות המסכים הבאים:

- מסך הוספת, עדכון, מחיקה של התמחות.
- מסך הוספת, עדכון, מחיקה של איש מקצוע.
- מסך הוספת, עדכון , מחיקה של מעסיק.
- מסך הוספת, עדכון , מחיקה של חוזה העסקה.
- בנוסף – לפחות עוד 3 מסכים המציגים שאילתות שמימשתם ב- BL.

התכנית תיפתח במסך ראשי המפנה לאפשרויות השונות. יש לשים לב שאנו לא יוצרים כל פעם מופע חדש של הBL (איך אפשר לעשות זאת?)
הערה: כאשר ישנו מסך המאפשר הוספת ערך כמו התמחות, איש מקצוע וכדו' הקיימים במערכת או עדכון, יש לאפשר בחירה של שדה זה ע"י פקד ComboBox שבו נקבל רשימה ממנה יש לבחור.
כמו כן כאשר אנו מאפשרים עדכון , לאחר בחירה מתוך רשימת הישויות לעדכון, יופיע המפתח על המסך ויתמלאו יתר השדות באופן אוטומטי, כאשר לא ניתן יהיה כמובן לשנות את ערך המפתח.
ישנה אפשרות שהוספה עדכון ומחיקה יהיו למעשה אותו מסך, בהתאם לצורך באותו רגע.

(אין להוסיף פונקציונאליות מעבר ל- CRUD בסיסי ל- DAL !)

כמו כן עליך לדאוג להוסיף זריקה של חריגות במקומות המתאימים ותפיסה של החריגות כדי שהתכנית לא "תעוף" במקרה שפעולה מסוימת נכשלה בזמן הרצה.

לאחר השלמת לימוד כל הWPF יש צורך לדאוג שכל האלמנטים שנלמדו בWPF מוטמעים בפרויקט. היכן ואיך לממש, זה תלוי בכל אחד ואחת לפי יצירתיותם.

הערה חשובה: יש לעשות שימוש ב:
Resources, data binding, converter, data context, dependency property, trigger,

כאן תתבצע הגנה השניה במספר.

חלק ב – עדכון שכבת ה- DAL

מימוש באמצעות Linq-To-XML

הוספה ומימוש מחלקה נוספת ב-DAL הממשת את ה- IDAL באמצעות Linq – to –XML:

עליך להוסיף לפרויקט **DAL** מחלקה נוספת בשם Dal_XML_imp

יש להכין קבצי xml שיחליפו את האוספים שיצרנו כבר (כלומר יחליפו את מקור הנתונים) אחד לכל אחת מהישויות, אשר ייכתבו בפורמט המתאים למבנה הישות אותה הוא מייצג. קבצים אלו יהיו בתוך תיקיה נפרדת ב- solution.

יש לעשות אותו תהליך שעשינו בכיתה כדי ליצור את האתחולים, אפשרות שמירה וטעינה של קובץ וכן אפשרות תשאול ע"י שאילתת Linq.
לאחר מכן יש לממש את כל המתודות של הממשק של ה- IDAL.

עבור קבלת רשימת סניפי הבנק הקיימים בארץ, נשתמש במסמך ה XML הבא שנמצא באתר בנק ישראל :
http://www.boi.org.il/he/BankingSupervision/BanksAndBranchLocations/Lists/BoiBankBranchesDocs/atm.xml

מכיוון שיש זמנים שבהם האתר הנ"ל מושבת, יש לכם גיבוי של כל המידע (נכון ל 09.12.16) גם בכתובת הבאה: http://www.jct.ac.il/~coshri/atm.xml

כיצד להוריד את הקובץ ?
ניתן להשתמש במחלקה WebClient של דוט נט כך:

```
const string xmlLocalPath = @"atm.xml";

WebClient wc = new WebClient();
try
{
    string xmlServerPath =
@"http://www.boi.org.il/he/BankingSupervision/BanksAndBranchLocations/Lists/Boi
BankBranchesDocs/atm.xml";
    wc.DownloadFile(xmlServerPath, xmlLocalPath);

}
catch (Exception)
{
    string xmlServerPath = @"http://www.jct.ac.il/~coshri/atm.xml";
    wc.DownloadFile(xmlServerPath, xmlLocalPath);
}
finally
{
    wc.Dispose();
}
```

הסבר:
המשתנה `xmlLocalPath` מכיל את הנתיב לשמירת הקובץ במחשב
שימו לב שמנסים להוריד מהאתר של בנק ישראל ובמידה ולא מצליחים מנסים להוריד מאתר השני.

אפשרות נוספת:
אם נעביר לפונקציה Load של XElement את כתובת האינטרנט, להפתעתנו הרבה גם זה יעבוד ונקבל XElement שתואם לקובץ ה xml
אבל, בכל אופן לענ"ד כדאי להוריד את הקובץ למחשב ואותו לטעון ב-XElement

עליך, להסתכל במבנה קובץ ה XML של בנק ישראל ולשלוף ממנו את המידע המתאים לרשימת הבנקים האפשריים.
למען האמת מה שמופיע בקובץ זה מידע על כל הכספומטים של הבנקים.
אבל, מתוך הנחה שלכל סניף יש לפחות כספומט אחד, נוכל באמצעות הקובץ לקבל את כל הסניפים הקיימים בארץ (וכמובן כדאי להוריד כפילויות למקרה שיש לסניף יותר מכספומט אחד)
שים לב שהאלמנטים אמנם כתובים בעברית אבל זה לא מפריע לנו לקרוא בצורה זו את הנתונים במערכת הדוט-נט.

שים לב שייקח זמן עד שהקובץ ירד ובזמן זה התוכנית לא תוכל להגיב
ולכן, מומלץ שהבנאי של ה DAL יבצע הורדה מהשרת לקובץ המקומי בתהליכון נפרד.
שאר הפונקציות שצריכות לגשת לקובץ זה יבדקו האם ההורדה הסתיימה ורק אם היא הסתיימה יאפשרו את הפעלת הפונקציה, אחרת תיזרק שגיאה בהתאם.
(ניתן להגדיר משתנה בוליאני שיציין האם ההורדה הסתיימה)

לגבי עבודה עם קבצי XML מקומיים, ניתן להיעזר בהדרכה להלן:

עבור אתחול קבצי הXML:

לדוגמה:
studentRoot = new XElement("students");
כעת נוכל כל פעם להוסיף \להוריד ולעדכן אלמנטים בקובץ
במידה והקובץ כבר קיים, פשוט לא ניצור אותו.
הקוד יהיה בערך כך:

```
public XmlSample()        {
            if (!File.Exists(FPath))
                CreateFiles();
        }

        private void CreateFiles()
        {
            studentRoot = new XElement("students");
        }
```

עבור המספר הרץ של חלק מהמחלקות ועוד כמה הגדרות:

הבעיה:
כשמימשנו את המספר הרץ השתמשנו במשתנה סטטי שגדל במהלך התוכנית.
כעת שנשמור את הנתונים ב- xml בפעם הבאה שנפתח את התוכנית אותו משתנה יתאפס שוב ואז המספר הרץ יתחיל מ- 0

הפתרון:
נגדיר קובץ XML בשם config.xml ושם נגדיר את המספר הרץ ועוד הגדרות שנרצה
כל פעם שנשמור אובייקט שמשתמש במספר הרץ נעדכן גם את קובץ ה config.xml
בפעם הראשונה שה dal נוצר הוא ידאג לעדכן זאת במחלקה.

הערה:
עבור קובץ ה- config ועוד קובץ אחד של אחת המחלקות, יש להשתמש ב- linq to xml
עבור כל פעולות ההוספה עדכון ומחיקה ושליפה.
במימוש של שאר המחלקות ניתן להשתמש ב- serialize, כלומר לעבוד עם הנתונים ב- list
ובסופו של דבר לשמור את זה ל- XML באמצעות xmlSerialize

כאן תתבצע בעז"ה ההגנה הסופית

בהצלחה !

אלו הן אמות הבדיקה לבדיקת הפרויקט ב- C#

עבור פרויקט טוב, אשר עונה על כל הקריטריונים שנתבקשו לענות, ניתן לקבל עד ציון 90.
כדי לקבל 100 יש לעמוד ב100% בכל מה שהמופיע בהמשך וכן להוסיף מעבר לכך.

א. ב- DAL וב- BL יש לבדוק: שימוש ב- Factory Method עבור DAL ועבור BL. רצוי מאוד ש- DAL יהיה בתבנית סינגלטון, שימוש מגוון ב- Linq, שימוש ב- new select, שימוש ב- Grouping, שימוש ב- Lambda, שימוש בanonymous delegates, שימוש ב- predicates ב- FUNC. רצוי שימוש ב- let בתוך שאילתת Linq.

ב. ב- BL יש לבדוק יצירתיות של בדיקות. כמו למשל אכיפת כללים המיוחדים לארגון הזה ע"פ הבנת התלמיד. יש לבדוק תוספות על אפשרויות ה bl שנכתבו בפרויקט, כלומר האם יש פונקציות נוספות שונות זו מזו אשר נוספו על המתבקש בפרויקט באופן מפורש.

ג. חריגות שעולות מ- DAL ל- BL ומ- BL ל- UI כאשר לכל שכבה יש גם חריגות מובנות משלה !

ד. ב- XML לבדוק שימוש נכון ב- LinQ To XML, כמו כן לסגור אפליקציה ולחזור אליה ולבדוק ששינויים אכן נשמרו. זריקת החריגות מ- DAL צריכה להיבדק.

ה. ב- UI לבדוק שימוש מגוון בפקדים. וחריגות שהגיעו משכבות קודמות שמטופלות וכן חריגות שנזרקות ומטופלות ששייכות לשכבת UI בעצמה כמו למשל שמשתמש לא מילא את כל השדות הנידרשים או שמילא שדות נומריים בתווים וכו'.

ו. יש לבדוק נתינת שמות משמעותיים למשתנים.

ז. חובה לבדוק שימוש בכל הנושאים שנלמדו בWPF ובכללן הנושאים שפורטו לעיל: resources, data binding, converter, data context, dependency property, trigger,

ח. כמו כן יש לבדוק תיעוד, ואלו הם הכללים לבדיקת התיעוד:
- ✓ מעל הפונקציות באינטרפסים של IDAL ו- IBL תוך שימוש ב \\\
- ✓ להוסיף תיעוד עבור מימושים רק מעל פונקציות יצירתיות ב- BL שאוכפות נהלים מסוימים שהמצאתם כמו למשל "מצא את כל העובדים מירושלים שמומחים בבסיסי נתונים" בקיצור, תיעוד רק עבור דברים שאינם טריוויאליים או שימוש בשאילתות מורכבות. כל השאר אין צורך לתעד.

כאן תתבצע ההגנה הסופית.

בהצלחה !

תאריכי הגשות:

הגשה 1 – חלק א עד שלב ג. תאריך הגשה: ז' טבת, 5.12.2016
הגשה 2 – חלק א שלב ד. תאריך הגשה: כ"א טבת, 19.1.2017
הגשה 3 – לאחר חלק ג' סופי – **תאריך הגשה** : מפגש אחרון בסמסטר ו' שבט, 2.2.2017
(ההגנה בשבוע האחרון בשעות המעבדה). לאחר סיום הסמסטר לא יתבצעו בדיקות.