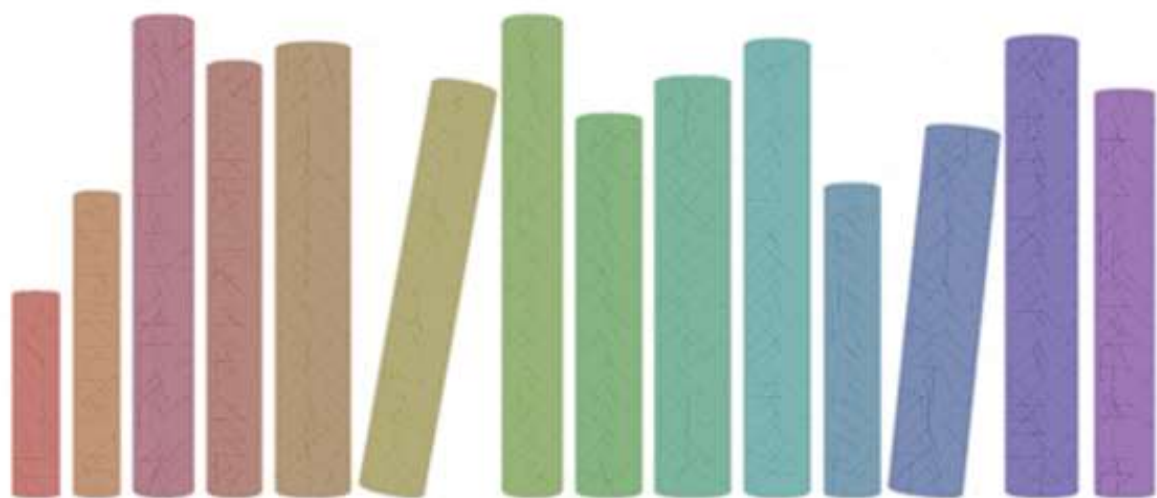


2/5/2022



# ספריה איזורית שדות נגב קורא כאן משהו טוב

בית הספר: נוות ישראל

שם העבודה: ספרייה

מגישה: שירה חדאד

ת.ז.: 212823736

המנחה: המורה מרים דבורקינג

חלופה: מערכות מנהליות

שנה: קיץ תשפ"ב



## תוכן עניינים:

3	תיאור הארגון:
3	מטרת העל של הארגון:
3	מטרות משניות:
3	תרשים מבנה ארגוני של החנות:
4	מהות המשימה:
4	תיחום המערכת:
4	גורמים מעורבים:
4	תיחום ארגוני:
4	תיחום תהליכי:
5	יעדים:
5	מצב קיים:
6	מטרות המערכת העתידית ומצב עתידי –
9	תרשים ERD:
10	אילוצי המערכת:
12	שכבת הנתונים
12	המחלקה Dal.cs
13	שכבת העסק - BLL
13	המחלקה GeneralTable
14	המחלקות nameTableTable:
17	המחלקה GeneralRow
17	מחלקות nameTableRow –
30	שכבת היישום GUI
63	קשרי גומלין:
63	טבלאות
66	תרשימי use case:
75	מסך פתיחה:
75	טופס הזדהות:
76	טופס ניווט:
76	טופס עובדים
77	טופס עובד יחיד
78	טופס לקוחות
80	טופס לקוח יחיד
80	טופס ספרים
81	טופס ספר יחיד



82	טופס השאלות.....
82	טופס השאלה יחידה.....
83	טופס השאלה יחידה.....
83	טופס סופרים.....
84	טופס סוגים.....



## מבוא

### תיאור הארגון:

הספרייה מציעה מגוון שירותים ומקום מזמין לצרכי למידה והעשרה, פנאי ותרבות, בילוי והנאה ומקום מפגש לקהילה. הספרייה פועלת מתוך רצון לתת מענה לצרכים המשתנים ולהשפיע לטובה על איכות חיי הפרט והקהילה.

הספרייה פתוחה בימים א'-ה' בין השעות 9:00 - 18:00.

בספרייה צוות מקצועי ומיומן השם דגש על שירות יעיל ומאיר פנים המסייע בהפעלתה. הספרייה משרתת ילדים, נוער ומבוגרים כאחד בכל שעות פעילותה במהלך הבוקר ואחר הצהריים.

מנהל הספרייה מעדכן את המלאי הקיים לפי רצונו, עבור כל ספר הוא בוחר את טווח גילאים אליו הוא מיועד.

הלקוחות המנויים יכולים לבחור ספרים מין המאגר לפי רצונם. כאשר לקוח רוצה ספר שאינו זמין במאגר כרגע (כי "נתפס" ע"י מנוי אחר) הלקוח יתווסף לרשימת הממתינים על אותו הספר.

אחרי החזרת ספר מהמנוי יש לבדוק האם יש לקוחות המחכים לספר זה ולהודיע להם.

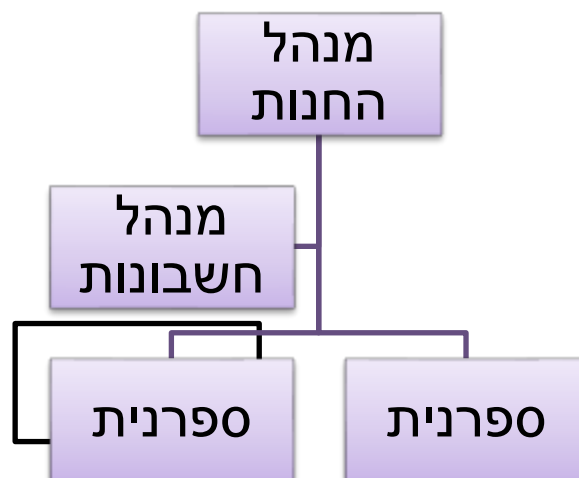
מכל ספר קיים לכל הפחות עותק אחד ממנו ולכן כאשר המנהל מוסיף ספר למערכת מתעדכנת גם כמות העותקים ממנו (במקרה והוא כבר קיים).

מטרת העל של הארגון: לספק מגוון שירותים ומקום מזמין לצרכי למידה והעשרה, פנאי ותרבות, בילוי והנאה ומקום מפגש לקהילה. הספרייה פועלת מתוך רצון לתת מענה לצרכים המשתנים ולהשפיע לטובה על איכות חיי הפרט והקהילה

### מטרות משניות:

- שמירה מסודרת של נתוני הספרים במאגר.
- שמירת הוגנות בין ממתינים לספר.
- יחס אדיב ללקוחות.
- חיבור בין מגזרי.

### תרשים מבנה ארגוני של החנות:





### מהות המשימה:

במערכת המידע הידנית של הספרייה יש בעיות אחדות:

1. אין מאגר מסודר של לקוחות וספקים ולכן כאשר רוצים להוסיף לקוח/ספק חדש יכולה להיווצר כפילות.
2. קשה לעקוב אחר החומרים המוזמנים ולכן יכולות להיווצר כפילויות בהזמנות הספרים.
3. קושי במעקב אחר המלאי בספרייה.
4. אין ללקוחות אפשרות לצפות בקטלוג מסודר ומעודכן של הספרים הקיימים.
5. קשה לערוך השאלה מסודרת עם חישוב הממתינים בתור. ולפעמים נוצרות טעויות במתן עדיפות לממתיין.

### תיחום המערכת:

גורמים מעורבים:

גורם	תיחום אחריות	רמת מעורבות
<b>מנהל החנות</b>	ניהול מאגר ספרים. הוספת ספקים חדשים. מעקב אחר קניות הלקוחות והזמנות מספקים	משתמש עיקרי
<b>מנהל חשבוניות</b>	מקבל דו"חות תשלומי פיגורים ופרטי הזמנות מספקים.	משתמש משני
<b>ספרנים/ות</b>	ניהול ההשאלה ללקוח הוספת לקוח. עדכון פרטי לקוח. הודעה ללקוח כאשר הספר שהמתין לו הגיע. העברת המלצות למנהל על הספרים להוספה/עדכון עפ"י בקשות הלקוחות.	משתמש עיקרי.

### תיחום ארגוני:

המערכת נועדה לפעילותם של מנהל החנות וספרנים.

### תיחום תהליכי:

1. סיוע בניהול מאגר הספרים.
2. עדכון מאגר הספרים.
3. ניהול מאגר הלקוחות.
4. עזרה במעקב אחר ההשאלות.



## יעדים:

שם היעד	תיאור
<b>ניהול מאגרים</b>	ניהול מאגרי המידע הבאים: לקוחות, עובדים, ספרים, השאלות, החזרות משאילים והמתנות.
<b>טיפול בפרטי לקוח</b>	<ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של לקוח והוספתם למאגר הלקוחות.</li> <li>עדכון פרטי לקוח לפי דרישת הלקוח או המנהל.</li> <li>הצגת פרטי לקוח מסוים.</li> <li>מציאת לקוח עפ"י שם</li> </ul>
<b>טיפול בפרטי עובד-ספרן</b>	<ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של העובד והוספתם למאגר העובדים.</li> <li>עדכון פרטי ספק לפי דרישת הספק או המנהל.</li> <li>הצגת פרטי עובד מסוים.</li> <li>מציאת עובד עפ"י שם</li> </ul>
<b>טיפול בפרטי ספר</b>	<ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של ספר והוספתם למאגר החומרים.</li> <li>עדכון פרטי ספר לפי דרישת המנהל.</li> <li>הצגת פרטי ספר מסוים.</li> <li>מציאת ספר להוספה עפ"י שם</li> </ul>
<b>טיפול בפרטי השאלה</b>	<ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של השאלה והוספתם למאגר ההשאלות.</li> <li>הצגת פרטי השאלה מסוימת.</li> <li>מציאת ספר עפ"י שם</li> </ul>
<b>טיפול בפרטי החזרת השאלה</b>	<ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של השאלה קיימת מתוך מאגר היצירות.</li> <li>עדכון פרטי ההשאלה לפי דרישת הלקוח.</li> <li>עדכון מצב הספר במאגר.</li> <li>שליחת הודעה לממתין שהספר פנוי.</li> <li>מחיקת המתנה.</li> </ul>
<b>טיפול בפרטי המתנה לספר</b>	<p>שליחת התעוררות ללקוח כשהספר מוחזר ע"י:</p> <ul style="list-style-type: none"> <li>קליטת נתונים פורמליים של המתנה לספר מסוים והוספתם למאגר המתנות.</li> <li>הצגת פרטי המתנה מסוימת.</li> <li>מחיקת ממתין.</li> </ul>

## מצב קיים

תהליך מס'	שם התהליך	תיאור קצר	ממוחשב (כן, לא, חלקית)	בעיות במצב הקיים
1. 1	השאלת ספר ללקוח	לקוח משאיל את הספר	חלקי	במערכת יש טעויות ברישום הספר
2. 2	החזרת השאלה מלקוח-מחיקת השאלה	רישום הספר כפנוי והודעה לממתין לו ומחיקת	חלקי	חוסר הגינות בהעברה ללקוח הממתין



		ההשאלה מהטבלה		
3.	הוספת לקוח חדש	לקוח חדש מתווסף למערכת	חלקי	בעיית כפילויות
4.	עדכון לקוח	שינוי פרטי לקוח קיים	חלקי	אין בדיקות תקינות
5.	חיפוש לקוח	חיפוש לקוח מסויים מהמאגר	לא	קשה ומייגע
6.	הוספת ספר חדש	ספר חדש מתווסף למערכת	חלקי	בעיות כפילות
7.	עדכון ספר	שינוי פרטי ספר קיים	חלקי	אין בדיקות תקינות
8.	חיפוש ספר	חיפוש ספר מסויים מהמאגר	לא	קשה ומייגע
9.	הוספת עובד חדש	עובד חדש מתווסף למערכת	חלקי	בעיות כפילות
10.	עדכון עובד	שינוי פרטי עובד קיים	חלקי	אין בדיקות תקינות
11.	חיפוש עובד	חיפוש עובד מסויים מהמאגר	לא	קשה ומייגע

### מטרות המערכת העתידית ומצב עתיד' -

#### מספר שם המטרה משתמש תגובת הערות המערכת

1.	אפשרות להוסיף לקוח חדש	עובד מנהל	או	מוסיפה לקוח	
2.	אפשרות לעדכן פרטי לקוח	עובד מנהל	או	מעדכנת פירטי לקוח	
3.	אפשרות לצפות ברשימת הלקוחות הקיימים	עובד מנהל	או	מציגה רשימת לקוחות	



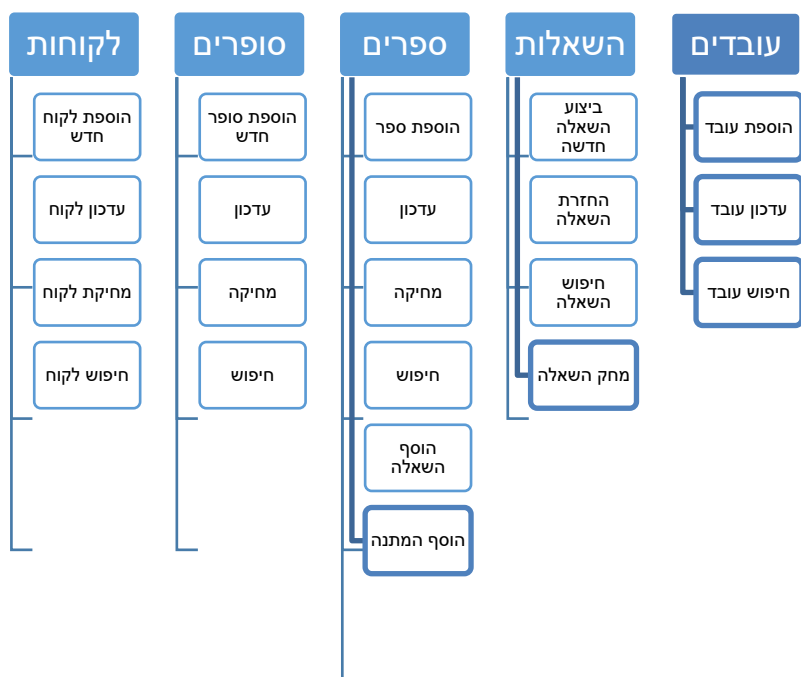
4.	אפשרות לחפש לקוח לפי שמו	עובד מנהל או	מחפשת לקוח	
5.	אפשרות להצגת נתונים על לקוח	עובד מנהל או	מציגה את פרטי הלקוח	
8.	אפשרות להוסיף ספר חדש	עובד מנהל או	מוסיף ספר חדש	
9.	אפשרות לעדכן פרטי ספר	עובד מנהל או	מעדכן פרטי ספר	
10.	אפשרות לצפות ברשימת הספרים הקיימים	עובד מנהל או	מציג רשימת הספרים הקיימים	את
11.	אפשרות לחפש ספר	עובד מנהל או	מחפש ספר	
12.	אפשרות להצגת נתונים על ספר מסוים	עובד מנהל או	מציג נתונים של ספר מסוים	
14.	אפשרות לשנות את הממתינים לספר	עובד מנהל או	מוסיף ממתין לאותו הספר	
15.	אפשרות להוסיף עובד חדש	עובד מנהל או	מוסיף עובד חדש	
16.	אפשרות לעדכן פרטי עובד	עובד מנהל או	מעדכן פרטי עובד	
17.	אפשרות לצפות ברשימת העובדים הקיימים	עובד מנהל או	מציג רשימת העובדים הקיימים	את
18.	אפשרות לחפש עובד	עובד מנהל או	מחפש עובד	





19.	אפשרות להצגת נתונים על עובד מסוים	עובד מנהל	או	מציג נתונים של ספר מסוים	
22.	אפשרות להוסיף השאלה חדשה	עובד מנהל	או	מוסיף השאלה חדש	
23.	אפשרות לעדכן פרטי השאלה	עובד מנהל	או	מעדכן פרטי השאלה	
24.	אפשרות לצפות ברשימת ההשאלות הקיימת	עובד מנהל	או	מציג את רשימת ההשאלות הקיימת	
26.	אפשרות להצגת נתונים על השאלה מסוימת	עובד מנהל	או	מציג נתונים של השאלה מסוימת	
27.	אפשרות להוסיף המתנה חדשה	עובד מנהל	או	מוסיף המתנה חדשה לספר מסוים	
28.	אפשרות לצפות ברשימת הממתינים לספר	עובד מנהל	או	מציג את רשימת הממתינים לספר	
30.	החזרת השאלה מסוימת	עובד מנהל	או	מוסיף את הספר למאגר בחזרה והודעה לממתין הבא	





### אילוצי המערכת:

- ☒ לו"ז- הגשת הפרויקט עד סוף יוני 2022.
- ☒ כ"א- פיתוח ומימוש התבצע ע"י תלמידת תיכון במסגרת פרויקט גמר.
- ☒ טכנולוגיה- שימוש במחשב PC.
- ☒ סביבת פיתוח- C#- visual studio 2019 , מסד נתונים Access 2016 בטכנולוגית Window Forms.

### **מבנה וארכיטקטורה:**

הספרייה לניהול אחזקת והשאלת ספרים מתייחסת ל-2 תהליכים מרכזיים:

1. השאלת ספרים ללקוח, וניהול מאגר הספרים.
2. שליחת התעוררות ללקוח ממתינ כשהספר מוחזר.

התוכנה נכתבה בשפת C# בסביבת Visual Studio 2019.



מחלקות ועצמים, קלט/פלט, ממשקי משתמש ידידותיים, תמיכה בבסיס נתונים, הורשה, טפסים.

התוכנית נבנתה בצורת חשיבה של DNA – מודל שלושת השכבות.

התוכנית חולקה לשלוש שכבות כאשר כל שכבה מטפלת בתחום שונה ומכילה כמה מחלקות ו/או טפסים שונים. בתוך כל מחלקה הוגדרו משתנים ואובייקטים מתאימים המיוחדים למחלקה. כל מחלקה מטפלת בתחום מסוים ומתקשרת לטופס/המחלקה המתאימים לה. צורת כתיבה זו נועדה להקל על המתכנת הן בכתיבת הקוד והן בתחזוקה השוטפת של הקוד. כמובן שבין המחלקות והטפסים שבשכבות השונות קיים קשר, וקימת התייחסות מתוך מחלקה אחת לעצמים המוגדרים במחלקות אחרות אך הכל נעשה תוך שמירה על בטחון והבטחת הנתונים והפונקציות.

פרוט השכבות:

1. DAL Data Access Layer – שכבה המטפלת בקישור התוכנה עם בסיס הנתונים.
2. BLL Business Logic Layer – שכבה המטפלת בכל הפונקציות הקשורות לתוכנה עצמה.
3. GUI Graphic User Interface – שכבה המטפלת בממשק החיצוני של התוכנה.



## מדריך למפתח

### שכבת הנתונים

שכבה זו מטפלת בקישור התוכנה עם בסיס הנתונים.

ובה מחלקה לתקשורת עם ה Dal.cs. DataBase

המחלקה Dal.cs

היחידה ש"מדברת" עם ה- DataBase כל שאר המחלקות פונות ל-DataBase דרך מחלקה זו. במלקה זו יש מספר פונקציות ששולפות, מעדכנות, מוחקות ומוסיפות ל- DataBase.

```
class ClassDAL
{
    private DataSet ds; אובייקט המכיל עותק של מסד הנתונים
    private OleDbConnection con; מחרוזת הקישור לאקסס

    פעולה בונה () public ClassDAL
    {
        ds = new DataSet();
        con = new OleDbConnection(@"Provider=Microsoft.Jet.OLEDB.4.0;Data
Source="+Application.StartupPath+ @"\super.mdb");
    }
    פעולה המייבאת טבלה כלשהי מהאקסס ל public void AddTable(string tableName)
    {
        OleDbDataAdapter adapter = new OleDbDataAdapter("select * from " +
tableName, con);
        if (!ds.Tables.Contains(tableName))//חסרה שהייתה שורה
            adapter.Fill(ds, tableName);
    }
    public DataTable GetTable(string tableName)
    {
        if (!ds.Tables.Contains(tableName))
            AddTable(tableName);
        return ds.Tables[tableName];
    }
    פעולה המחזירה תוצאות של שאילתה public DataTable GetQuery(string sqlString)
    {
        OleDbDataAdapter adapter = new OleDbDataAdapter(sqlString, con);
        DataTable table = new DataTable();
        adapter.Fill(table);
        return table;
    }
    פעולה המעדכנת את קובץ האקסס מתוך האובייקט שמכיל את העותק public void Update(string tableName)
    {
        OleDbDataAdapter adapter = new OleDbDataAdapter("select * from " +
tableName, con);
        OleDbCommandBuilder builder = new OleDbCommandBuilder(adapter);
        adapter.InsertCommand = builder.GetInsertCommand();
        adapter.UpdateCommand = builder.GetUpdateCommand();
        adapter.DeleteCommand = builder.GetDeleteCommand();
        adapter.Update(ds, tableName);
    }
}
```



## שכבת העסק – BLL

שכבה המטפלת בכל הפונקציות והלוגיקה הקשורות לתוכנה עצמה.

המחלקות המוכלות בשכבה:

- מחלקת בסיס לטיפול בטבלה אחת מתוך ה `DataBase` – `GeneralTable`
- מחלקה עבור כל טבלה מה `DataBase` היורשת ממהמחלקה `GeneralTable`
- מחלקת בסיס לטיפול הרשומה אחת מתוך ה `DataBase` - `GeneralRow`
- מחלקה עבור כל רשומה מטבלה היורשת ממחלקת `GeneralRow`

### המחלקה `GeneralTable`

היא היחידה המתקשרת עם מחלקת `Dal`. מנהלת קשר עם טבלה בודדת מתוך ה `DataBase`. לא ניתן ליצור ממנה עצמים כיוון שהיא כללית ולא ספציפית לטבלה מסוימת ב `DataBase`. המחלקות היורשות ממנה הם מתייחסות כל אחת לטבלה אחרת מהמאגר.

תפקידן הוא לקשר בין שכבת הנתונים לבין שכבת הממשק החיצוני.

```
class GeneralTable
{
    protected static ClassDAL dal = new ClassDAL() מחלקת של מחלקת Dal מופע סטטי של
    protected DataTable table; אובייקט המכיל את הטבלה
    protected string tableName; שם הטבלה
    protected string key; שם המפתח הראשי
    public GeneralTable(string tableName, string key)
    {
        this.tableName = tableName;
        this.key = key;
        dal.AddTable(tableName);
        this.table = dal.GetTabel(tableName);
    }
    public DataTable GetTable() פעולה המחזירה את הטבלה
    {
        return this.table;
    }
    public void AddRow(DataRow item)
    {
        table.Rows.Add(item);
        Save();
    }
    public DataRow GetNewRow() מחזירה שורה חדשה במבנה של הטבלה הזו
    {
        return table.NewRow();
    }
    public void Save() שומרת את הנתונים באקסס
    {
        dal.Update(table.TableName);
    }
    public DataRow FindRow(object num) חיפוש שורה לפי עמודת מפתח
    {
        foreach (DataRow item in table.Rows)
        {
            if (item[this.key].Equals(num))
                return item;
        }
    }
}
```



```

    }
    return null;
}
//החדשה הרשומה של הקוד החזרת - חדש/
public int GetNewKey()
{
    DataTable dt = dal.GetQuery("select max(" + key + ") from " +
table.TableName);
    if (dt.Rows[0][0] is DBNull)
        return 1;
    return Convert.ToInt32(dt.Rows[0][0]) + 1;
}
}

```

### המחלקות nameTableTable

כל אחת מן המחלקות הבאות יורשת מהמחלקה **GeneralTable** ומנהלת את הטיפול בטבלה בודדת מתוך ה **DataBase**

מחלקת **AuthorsTable\_T**:

```

public class AuthorsTable_T:GeneralTable
{
    public AuthorsTable_T() : base("AuthorsTable", "CodeAuthor")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new AuthorsTable(item));
        }
    }

    public List<AuthorsTable> GetList()
    {
        return base.list.Cast<AuthorsTable>().ToList();
    }
}

```

מחלקת **BooksTable\_T**

```

class BooksTable_T:GeneralTable
{
    public BooksTable_T() : base("BooksTable", "codeBook")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new BooksTable(item));
        }
    }

    public List<BooksTable> GetList()
    {

```



```
        return base.list.Cast<BooksTable>().ToList();
    }
}
```

מחלקת :BorrowsBooksTable\_T

```
class BorrowsBooksTable_T:GeneralTable
{
    public BorrowsBooksTable_T() : base("BorrowsBooksTable",
"CodeBookinborrow")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new BorrowsBooksTable(item));
        }
    }

    public List<BorrowsBooksTable> GetList()
    {
        return base.list.Cast<BorrowsBooksTable>().ToList();
    }
}
```

מחלקת :BorrowTable\_T

```
public class BorrowTable_T:GeneralTable
{
    public BorrowTable_T() : base("BorrowTable", "CodeBorrow")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new BorrowTable(item));
        }
    }

    public List<BorrowTable> GetList()
    {
        return base.list.Cast<BorrowTable>().ToList();
    }
}
```

מחלקת ClientTable\_T

```
public class ClientTable_T:GeneralTable
{
    public ClientTable_T() : base("ClientTable", "IdClient")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new ClientTable(item));
        }
    }

    public List<ClientTable> GetList()
    {
        return base.list.Cast<ClientTable>().ToList();
    }
}
```





}

## מחלקת CopyTable\_T:

```
public class CopyTable_T:GeneralTable
{
    public CopyTable_T() : base("CopyTable", "CodeCopy")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new CopyTable(item));
        }
    }

    public List<CopyTable> GetList()
    {
        return base.list.Cast<CopyTable>().ToList();
    }
}
```

## מחלקת EmploiesTable\_T:

```
public class EmploiesTable_T:GeneralTable
{
    public EmploiesTable_T() : base("EmploiesTable", "IdEmploy")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new EmploiesTable(item));
        }
    }

    public List<EmploiesTable> GetList()
    {
        return base.list.Cast<EmploiesTable>().ToList();
    }
}
```

## מחלקת KindTable\_T:

```
public class KindTable_T:GeneralTable
{
    public KindTable_T() : base("KindTable", "CodeKind")
    {
        foreach (DataRow item in table.Rows)
        {
            list.Add(new KindTable(item));
        }
    }

    public List<KindTable> GetList()
    {
        return base.list.Cast<KindTable>().ToList();
    }
}
```

מחלקת `WaitingTable_T`:

```
public WaitingTable_T() : base("WaitingTable", "codeWaiting")
{
    foreach (DataRow item in table.Rows)
    {
        list.Add(new WaitingTable(item));
    }

    public List<WaitingTable> GetList()
    {
        return base.list.Cast<WaitingTable>().ToList();
    }
}
```

המחלקה GeneralRow

מחלקה כללית, המחייבת את כל המחלקות היורשות ממנה לממש את הפעולות המטפלות ברשומה בודדת בטבלה מתוך DataBase לתוך המחלקה המתאימה. מכילה אובייקט מסוג `DataRow`.

מחלקות `nameTableRow` –

כל מחלקה מהן, מטפלת ברשומה אחת מתוך טבלה מסוימת.

תכונותיה: משתנים המקבילים לשדות של אותה טבלה (גם בסוג הנתונים), אובייקט מסוג מחלקת DB השייכת לטבלה זו.

מחלקת `AuthorsTable`

```
public class AuthorsTable : GeneralRow
{
    private int CodeAuthor;

    public int Code_Author
    {
        get { return CodeAuthor; }
        set {
            CodeAuthor = value; }
    }

    private string NameAuthor;

    public string Name_Author
    {
        get { return NameAuthor; }
    }
}
```



```

        set {
            if (value == "")
                throw new Exception("חובה שדה");
            NameAuthor = value; }
    }

    public override void FillFields()//השורה מתוך העצם תכונות מילוי
    {
        this.CodeAuthor = Convert.ToInt32(drow["CodeAuthor"]);
        this.NameAuthor = drow["NameAuthor"].ToString();
    }

    public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
    {
        drow["CodeAuthor"] = this.CodeAuthor;
        drow["NameAuthor"] = this.NameAuthor;
    }

    public AuthorsTable()
    {
    }

    public AuthorsTable(DataRow dr)//שורה לפי סופר היוצרת בונה פעולה
    {
        drow = dr;
        FillFields();
    }

    public AuthorsTable(int id)//ז"ת לפי סופר היוצרת בונה פעולה
    {
        AuthorsTable_T author = new AuthorsTable_T();
        drow = author.Find(id);
        FillFields();
    }

    public List<BooksTable> GetCodeAuthor()
    {
        BooksTable_T reg = new BooksTable_T();
        return reg.GetList().Where(x => x.Code_Author ==
this.CodeAuthor).ToList();
    }

    public override string ToString()
    {
        return NameAuthor;//הסופר שם הדפסת פעולת
    }
}

```

#### מחלקת BooksTable

```

public class BooksTable : GeneralRow
{
    private int codeBook;

    public int code_Book
    {
        get { return codeBook; }
        set { if (value > 0) codeBook = value; }
    }
}

```



```

private int CodeAuthor;

public int Code_Author
{
    get { return CodeAuthor; }
    set { CodeAuthor = value; }
}

private int CodeKind;

public int Code_Kind
{
    get { return CodeKind; }
    set { CodeKind = value; }
}

private string NameBook;

public string Name_Book
{
    get { return NameBook; }
    set {
        if (value == " ")
            throw new Exception("חובה שדה");
        NameBook = value; }
}

public override void FillFields()//השורה מתוך העצם תכונות מילוי
{
    this.codeBook = Convert.ToInt32(drow["codeBook"]);
    this.CodeAuthor = Convert.ToInt32(drow["CodeAuthor"]);
    this.CodeKind = Convert.ToInt32(drow["CodeKind"]);
    this.NameBook = drow["NameBook"].ToString();
}

public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
{
    drow["codeBook"] = this.codeBook;
    drow["CodeAuthor"] = this.CodeAuthor;
    drow["CodeKind"] = this.CodeKind;
    drow["NameBook"] = this.NameBook;
}

public BooksTable()
{
}

public BooksTable(DataRow dr)//שורה לפי ספר היוצרת בונה פעולה
{
    drow = dr;
    FillFields();
}

public BooksTable(int id)//קוד לפי ספר היוצרת בונה פעולה
{
    BooksTable_T book = new BooksTable_T();
    drow = book.Find(id);
    FillFields();
}

```



```

    }
    public List<CopyTable> GetCodeBook()
    {
        CopyTable_T reg = new CopyTable_T();
        return reg.GetList().Where(x => x.Code_Book == this.codeBook).ToList();
    }

    public AuthorsTable GetAuthor()
    {
        AuthorsTable_T reg = new AuthorsTable_T();
        return reg.GetList().FirstOrDefault(x => x.Code_Author ==
this.CodeAuthor);
    }
    public KindTable GetKind()
    {
        KindTable_T reg = new KindTable_T();
        return reg.GetList().FirstOrDefault(x => x.Code_Kind ==
this.CodeKind);
    }
    public override string ToString()
    {
        return Name_Book; //הספר שם הדפסת פעולת
    }
}
}

```

```

-----

public class BorrowsBooksTable:GeneralRow                                מחלקתBorrowsBooksTable
{
    private int CodeBookinborrow;

    public int Code_Bookinborrow
    {
        get { return CodeBookinborrow; }
        set { CodeBookinborrow = value; }
    }

    private int CodeBorrow;

    public int Code_Borrow
    {
        get { return CodeBorrow; }
        set { CodeBorrow = value; }
    }

    private int CodeCopy;

    public int Code_Copy
    {
        get { return CodeCopy; }
        set { CodeCopy = value; }
    }

    public bool Status { get; set; }

    public override void FillFields()//השורה מתוך העצם תכונות מילוי
    {
        this.CodeBookinborrow = Convert.ToInt32(drow["CodeBookinborrow"]);
    }
}

```



```
this.CodeBorrow = Convert.ToInt32(drow["CodeBorrow"]);
this.CodeCopy = Convert.ToInt32(drow["CodeCopy"]);
this.Status = Convert.ToBoolean(drow["Status"]);
}

public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
{
    drow["CodeBookinborrow"] = this.CodeBookinborrow;
    drow["CodeBorrow"] = this.CodeBorrow;
    drow["CodeCopy"] = this.CodeCopy;
    drow["Status"] = this.Status;
}
public BorrowsBooksTable()
{

}
public BorrowsBooksTable(DataRow dr)//שורה לפי בהמתנה ספר היוצרת בונה פעולה
{
    drow = dr;
    FillFields();
}
public BorrowsBooksTable(int id)//קוד לפי בהמתנה ספר היוצרת בונה פעולה
{
    BorrowsBooksTable_T borrowBook = new BorrowsBooksTable_T();
    drow = borrowBook.Find(id);
    FillFields();
}

public BorrowTable GetBorrow()
{
    BorrowTable_T reg = new BorrowTable_T();
    return reg.GetList().FirstOrDefault(x => x.Code_Borrow ==
this.Code_Borrow);
}
public CopyTable GetCopy()
{
    CopyTable_T reg = new CopyTable_T();
    return reg.GetList().FirstOrDefault(x => x.Code_Copy ==
this.Code_Copy);
}

}
```

-----

## מחלקת BorrowTable

```
public class BorrowTable:GeneralRow
{
    private int CodeBorrow;

    public int Code_Borrow
    {
        get { return CodeBorrow; }
        set { CodeBorrow = value; }
    }

    private int CodeClient;
```



```

public int Code_Client
{
    get { return CodeClient; }
    set { CodeClient = value; }
}

private int codeEmploy;

public int code_Employ
{
    get { return codeEmploy; }
    set {codeEmploy = value; }
}

private DateTime DateBorrow;

public DateTime Date_Borrow
{
    get { return DateBorrow; }
    set
    {
        if (value < DateTime.Today)
            throw new Exception("תקין לא התאריך");
        DateBorrow = value; }
}

public bool Status { get; set; }
public override void FillFields()//השורה מתוך העצם תכונות מילוי
{
    this.CodeBorrow = Convert.ToInt32(drow["CodeBorrow"]);
    this.CodeClient = Convert.ToInt32(drow["CodeClient"]);
    this.codeEmploy = Convert.ToInt32(drow["codeEmploy"]);
    this.DateBorrow = Convert.ToDateTime(drow["DateBorrow"]);
    this.Status = Convert.ToBoolean(drow["Status"]);
}

public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
{
    drow["CodeBorrow"] = this.CodeBorrow;
    drow["CodeClient"] = this.CodeClient;
    drow["codeEmploy"] = this.codeEmploy;
    drow["DateBorrow"] = this.DateBorrow;
    drow["Status"] = this.Status;
}

public BorrowTable()
{
}

public BorrowTable(DataRow dr)//שורה לפי השאלה היוצרת בונה פעולה
{
    drow = dr;
    FillFields();
}

public BorrowTable(int id)//קוד לפי השאלה היוצרת בונה פעולה
{
    BorrowTable_T borrow = new BorrowTable_T();
    drow = borrow.Find(id);
    FillFields();
}

public List<BorrowBooksTable> GetBorrowBooks()
{
    BorrowBooksTable_T reg = new BorrowBooksTable_T();

```



```

        return reg.GetList().Where(x => x.Code_Borrow ==
this.CodeBorrow).ToList();
    }
    public ClientTable GetClient()
    {
        ClientTable_T reg = new ClientTable_T();
        return reg.GetList().FirstOrDefault(x => x.Id_Client ==
this.CodeClient);
    }
    public EmploiesTable GetEmploies()
    {
        EmploiesTable_T reg = new EmploiesTable_T();
        return reg.GetList().FirstOrDefault(x => x.Id_Employ ==
this.codeEmploy);
    }
}
}

```

#### מחלקת ClientTable

```

public class ClientTable:GeneralRow
{
    private int IdClient;

    public int Id_Client
    {
        get { return IdClient; }
        set {
            IdClient = value; }
    }
    private string NameClient;

    public string Name_Client
    {
        get { return NameClient; }
        set {
            NameClient = value; }
    }
    private string mailClient;

    public string MailClient
    {
        get { return mailClient; }
        set {
            if (validates.CheackMail(value) == false)
                throw new Exception("שגוי מייל");
            mailClient = value; }
    }

    private string telClient;

    public string TelClient
    {
        get { return telClient; }
        set
        {
            if (validates.IsTelephone(value) == false||
validates.IsCellPhone(value) == false)
                throw new Exception("שגוי טלפון");
            telClient = value;
        }
    }
}

```





```

    }
    public override void FillFields()//השורה מתוך העצם תכונות מילוי
    {
        this.IdClient = Convert.ToInt32(drow["IdClient"]);
        this.NameClient = drow["NameClient"].ToString();
        this.MailClient = drow["MailClient"].ToString();
        this.TelClient = drow["TelClient"].ToString();
    }

    public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
    {
        drow["IdClient"] = this.IdClient;
        drow["NameClient"] = this.NameClient;
        drow["MailClient"] = this.MailClient;
        drow["TelClient"] = this.TelClient;
    }
    public ClientTable()
    {
    }
    public ClientTable(DataRow dr)//שורה לפי לקוח היוצרת בונה פעולה
    {
        drow = dr;
        FillFields();
    }
    public ClientTable(int id)//ז"ת לפי לקוח היוצרת בונה פעולה
    {
        ClientTable_T client = new ClientTable_T();
        drow = client.Find(id);
        FillFields();
    }
    public List<BorrowTable> GetBorrow()
    {
        BorrowTable_T reg = new BorrowTable_T();
        return reg.GetList().Where(x => x.Code_Client ==
this.Id_Client).ToList();
    }
    public List<WaitingTable> GetWaiting()
    {
        WaitingTable_T reg = new WaitingTable_T();
        return reg.GetList().Where(x => x.Code_Client ==
this.IdClient).ToList();
    }
    public override string ToString()
    {
        return NameClient;//הלקוח שם הדפסת פעולת
    }
}

```

#### מחלקת CopyTable

```

public class CopyTable:GeneralRow
{
    private int CodeCopy;

    public int Code_Copy
    {
        get { return CodeCopy; }
        set { CodeCopy = value; }
    }
}

```



```

    }
    private int CodeBook;

    public int Code_Book
    {
        get { return CodeBook; }
        set { CodeBook = value; }
    }
    public bool Isborrow { get; set; }

    public override void FillFields()//השורה מתוך העצם תכונות מילוי
    {
        this.CodeCopy = Convert.ToInt32(drow["CodeCopy"]);
        this.CodeBook = Convert.ToInt32(drow["CodeBook"]);
        this.Isborrow = Convert.ToBoolean(drow["Isborrow"]);
    }

    public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
    {
        drow["CodeCopy"] = this.CodeCopy;
        drow["CodeBook"] = this.CodeBook;
        drow["Isborrow"] = this.Isborrow;
    }
    public CopyTable()
    {
    }
    public CopyTable(DataRow dr)//שורה לפי עותק היוצרת בונה פעולה
    {
        drow = dr;
        FillFields();
    }
    public CopyTable(int id)//קוד לפי עותק היוצרת בונה פעולה
    {
        CopyTable_T copy = new CopyTable_T();
        drow = copy.Find(id);
        FillFields();
    }
    public List<BorrowsBooksTable> GetBorrowsBooks()
    {
        BorrowsBooksTable_T reg = new BorrowsBooksTable_T();
        return reg.GetList().Where(x => x.Code_Copy ==
this.CodeCopy).ToList();
    }
    public BooksTable GetBooks()
    {
        BooksTable_T reg = new BooksTable_T();
        return reg.GetList().FirstOrDefault(x => x.code_Book ==
this.CodeBook);
    }
}

```

מחלקת EmploiesTable

```

public class EmploiesTable:GeneralRow
{
    private int IdEmploy;

    public int Id_Employ

```



```

{
    get { return IdEmploy; }
    set { IdEmploy = value; }
}
private string NameEmploy;

public string Name_Employ
{
    get { return NameEmploy; }
    set {
        if (value == "")
            throw new Exception("חובה שדה");
        NameEmploy = value; }
}
private string MailEmploy;

public string Mail_Employ
{
    get { return MailEmploy; }
    set {
        if (validates.CheackMail(value) == false)
            throw new Exception("שגוי מייל");
        MailEmploy = value; }
}

private string TelEmploy;

public string Tel_Employ
{
    get { return TelEmploy; }
    set {
        if (validates.IsTelephone(value) == false ||
validates.IsCellPhone(value) == false)
            throw new Exception("שגוי טלפון");
        TelEmploy = value; }
}
private string PasswordClient;

public string Password_Client
{
    get { return PasswordClient; }
    set { PasswordClient = value; }
}

private string KindClient;

public string Kind_Client
{
    get { return KindClient; }
    set {
        if (value != "מנהל" && value != "משתמש")
            throw new Exception("בלבד מנהל או משתמש להזין ניתן");
        KindClient = value;
    }
}

public override void FillFields()//השורה מתוך העצם תכונות מילוי
{

```



```

        this.IdEmploy = Convert.ToInt32(drow["IdEmploy"]);
        this.NameEmploy = drow["NameEmploy"].ToString();
        this.MailEmploy = drow["MailEmploy"].ToString();
        this.TelEmploy = drow["TelEmploy"].ToString();
        this.PasswordClient = drow["Password_Client"].ToString();
        this.KindClient = drow["KindClient"].ToString();
    }

    public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
    {
        drow["IdEmploy"] = this.IdEmploy;
        drow["NameEmploy"] = this.NameEmploy;
        drow["MailEmploy"] = this.MailEmploy;
        drow["TelEmploy"] = this.TelEmploy;
        drow["Password_Client"] = this.PasswordClient;
        drow["KindClient"] = this.KindClient;
    }
    public EmploiesTable()
    {

    }
    public EmploiesTable(DataRow dr)//שורה לפי עובד היוצרת בונה פעולה
    {
        drow = dr;
        FillFields();
    }
    public EmploiesTable(int id)//ז"ת לפי עובד היוצרת בונה פעולה
    {
        EmploiesTable_T employe = new EmploiesTable_T();
        drow = employe.Find(id);
        FillFields();
    }

    public List<BorrowTable> GetBorrow()
    {
        BorrowTable_T reg = new BorrowTable_T();
        return reg.GetList().Where(x => x.code_Employ ==
this.IdEmploy).ToList();
    }
    public override string ToString()
    {
        return NameEmploy;//העובד שם הדפסת פעולת
    }

}
}

```

מחלקת KindTable

```

public class KindTable : GeneralRow
{
    private int CodeKind;

    public int Code_Kind
    {
        get { return CodeKind; }
        set { CodeKind = value; }
    }
    private string Namekind;

    public string Name_kind
    {
        get { return Namekind; }
    }
}

```



```
set {
    if (value == "")
        throw new Exception("חובה שדה");
    Namekind = value; }
}

public override void FillFields()//השורה מתוך העצם תכונות מילוי
{
    this.CodeKind = Convert.ToInt32(drow["CodeKind"]);
    this.Namekind = drow["Namekind"].ToString();
}

public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
{
    drow["CodeKind"] = this.CodeKind;
    drow["Namekind"] = this.Namekind;
}

public KindTable()
{
}

public KindTable(DataRow dr)//שורה לפי סוג היוצרת בונה פעולה
{
    drow = dr;
    FillFields();
}

public KindTable(int id)//קוד לפי סוג היוצרת בונה פעולה
{
    KindTable_T kind = new KindTable_T();
    drow = kind.Find(id);
    FillFields();
}

public List<BooksTable> GetBooks()
{
    BooksTable_T reg = new BooksTable_T();
    return reg.GetList().Where(x => x.Code_Kind== this.CodeKind).ToList();
}

public override string ToString()
{
    return Namekind;//הסוג שם הדפסת פעולת
}

}
```

---

## מחלקת WaitingTable

```
public class WaitingTable:GeneralRow
{
    private int codeWaiting;

    public int code_Waiting
    {
        get { return codeWaiting; }
        set { codeWaiting = value; }
    }

    private int CodeClient;

    public int Code_Client
```



```

    {
        get { return CodeClient; }
        set {CodeClient = value; }
    }
    private int CodeBook;

    public int Code_Book
    {
        get { return CodeBook; }
        set {CodeBook = value; }
    }
    public bool StatusWaiting { get; set; }

    public override void FillFields()//השורה מתוך העצם תכונות מילוי
    {
        this.codeWaiting = Convert.ToInt32(drow["codeWaiting"]);
        this.CodeClient = Convert.ToInt32(drow["CodeClient"]);
        this.CodeBook = Convert.ToInt32(drow["CodeBook"]);
        this.StatusWaiting = Convert.ToBoolean(drow["StatusWaiting"]);
    }

    public override void FillDataRow()//העצם תכונות מתוך השורה מילוי
    {
        drow["codeWaiting"] = this.codeWaiting;
        drow["CodeClient"] = this.CodeClient;
        drow["CodeBook"] = this.CodeBook;
        drow["StatusWaiting"] = this.StatusWaiting;
    }
    public WaitingTable()
    {
    }
    public WaitingTable(DataRow dr)//שורה לפי ממתין היוצרת בונה פעולה
    {
        drow = dr;
        FillFields();
    }
    public WaitingTable(int id)//ז"ת לפי ממתין היוצרת בונה פעולה
    {
        WaitingTable_T wait = new WaitingTable_T();
        drow = wait.Find(id);
        FillFields();
    }
    public ClientTable GetClient()
    {
        ClientTable_T reg = new ClientTable_T();
        return reg.GetList().FirstOrDefault(x => x.Id_Client ==
this.CodeClient);
    }
    public BooksTable GetBook()
    {
        BooksTable_T reg = new BooksTable_T();
        return reg.GetList().FirstOrDefault(x => x.code_Book ==
this.CodeBook);
    }
}

```



## שכתב היישום GUI

שכתב המטפלת בממשק החיצוני של התוכנה.

טופס הוספה + עדכון סופרים + חיפוש

טופס זה מחולק לשני חלקים

1. הצגת הסופרים בטבלה, פקדים לחיפוש סופר מסוים ע"י שם

2. הוספה, עדכון ומחיקת סופר

```
public partial class frmAuthorTable : Form
{
    BLL.AuthorsTable_T aTable;
    BLL.AuthorsTable a;
    public frmAuthorTable()
    {
        InitializeComponent();
        aTable = new BLL.AuthorsTable_T();
        a = new BLL.AuthorsTable();
        rbadd.Checked = true;
        refresh();
    }

    private void frmAuthorTable_Load(object sender, EventArgs e)
    {
    }

    public void refresh()
    {
        aTable = new BLL.AuthorsTable_T();
        dgvAuthors.DataSource = null;
        dgvAuthors.DataSource = aTable.GetList().Select(x => new {
x.Code_Author, x.Name_Author }).ToList();
        ChangeHeaders();
    }

    private void ChangeHeaders()
    {
        dgvAuthors.Columns["Code_Author"].HeaderText = "סופר קוד";
        dgvAuthors.Columns["Name_Author"].HeaderText = "סופר שם";
    }

    private void rbadd_CheckedChanged(object sender, EventArgs e)
    {
        if (rbadd.Checked == true)
        {
            gbMatzav.Text = "הוספה";
            txtname.Text = "";
        }
    }

    private void rbedkun_CheckedChanged(object sender, EventArgs e)
    {
    }
}
```



```

        if (rbedkun.Checked == true)
        {
            gbMatzav.Text = "עדכון";
            txtname.Text = "";
        }
    }

    public bool Check()
    {
        bool ok = true;
        try
        {
            a.Name_Author = txtname.Text;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            ok = false;
        }
        return ok;
    }

    private void btnok_Click(object sender, EventArgs e)
    {
        if (aTable.GetList().Any(x => x.Name_Author == txtname.Text) ==
true)
            MessageBox.Show("קיים הסופר");
        else
            if (Check())
            {
                if (rbadd.Checked == true)
                {
                    a.Code_Author = aTable.GetNewKey();
                    aTable.Add(a);
                    a = new BLL.AuthorsTable();
                    MessageBox.Show("בהצלחה נוסף הסופר");
                }
                if (rbedkun.Checked == true)
                {
                    aTable.Update(a);
                    MessageBox.Show("בהצלחה עודכן הסופר");
                }
                txtname.Text = "";
                refresh();
            }
    }

    private void dgvAuthors_RowHeaderMouseDoubleClick(object sender,
DataGridViewCellMouseEventArgs e)
    {
        if (rbedkun.Checked == true)
        {
            int code =
Convert.ToInt32(dgvAuthors.SelectedRows[0].Cells[0].Value);
            a = aTable.GetList().FirstOrDefault(x => x.Code_Author ==
code);
            txtname.Text = a.Name_Author;
        }
    }

```





```
private void txtfind_TextChanged(object sender, EventArgs e)
{
    dgvAuthors.DataSource = aTable.GetList().Where(x =>
x.Name_Author.Contains(txtfind.Text)).Select(x => new { x.Code_Author,
x.Name_Author }).ToList();
}

}
```

~~~~~

טופס הוספה + עדכון ספרים + הצגה ובנוסף הוספת השאלה

טופס זה מחולק לשלושה חלקים

1. הצגת הספרים בטבלה.
2. הוספה, עדכון ומחיקת ספר.
3. הוספת השאלה

```
public partial class frmBooksTable : Form
{
    BooksTable_T bTable;
    BooksTable b;
    public frmBooksTable()
    {
        InitializeComponent();
        b = new BooksTable();
        refresh();
    }
    public void refresh()
    {
        dgvBook.DataSource = null;
        bTable = new BooksTable_T();
        dgvBook.DataSource = bTable.GetList().Select(x => new {
x.code_Book, x.GetAuthor().Name_Author, x.GetKind().Name_kind, x.Name_Book
}).ToList();
        changeHeaders();
    }
    private void changeHeaders()
    {
        dgvBook.Columns["code_Book"].HeaderText = "ספר קוד";
        dgvBook.Columns["Name_Author"].HeaderText = "שם";
        dgvBook.Columns["Name_Kind"].HeaderText = "קטגוריה";
        dgvBook.Columns["Name_Book"].HeaderText = "ספר שם";
    }

    private void frmBooksTable_Load(object sender, EventArgs e)
    {
    }

    private void button1_Click(object sender, EventArgs e)
    {
    }
}
```



```

        frmOneBook frm = new frmOneBook();
        frm.ShowDialog();
        refresh();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        if (b.code_Book == 0)
            MessageBox.Show("ספר נבחר לא");
        else
        {
            frmOneBook frm = new frmOneBook(b, eState.עדכון);
            frm.ShowDialog();
            refresh();
        }
    }

    private void button3_Click(object sender, EventArgs e)
    {
        if (b.code_Book == 0)
            MessageBox.Show("ספר נבחר לא");
        else
        {
            frmOneBook frm = new frmOneBook(b, eState.הצגה);
            frm.ShowDialog();
            refresh();
        }
    }

    private void dgvClient_RowHeaderMouseDoubleClick(object sender,
DataGridViewCellMouseEventArgs e)
    {
        int code =
Convert.ToInt32(dgvBook.SelectedRows[0].Cells[0].Value);
        b = bTable.GetList().FirstOrDefault(x => x.code_Book == code);

    }

    private void button4_Click(object sender, EventArgs e)
    {
        if (b.code_Book == 0)
            MessageBox.Show("ספר נבחר לא");
        else
        {
            frmOneBook frm = new frmOneBook(b, eState.השאלה_הוסף);
            frm.ShowDialog();
            refresh();
        }
    }
}

```

~~~~~

טופס הוספה + עדכון לקוחות + הצגה+חיפוש על פי שם

טופס זה מחולק לשלושה חלקים



1. הצגת הלקוחות בטבלה
2. הוספה, עדכון והצגת לקוח .
3. חיפוש לקוח עפ"י שם

```
public partial class frmClientTable : Form
{
    ClientTable c;
    ClientTable_T cTable;
    public frmClientTable()
    {
        InitializeComponent();
        c = new ClientTable();
        cTable = new ClientTable_T();
        refresh();
    }

    private void refresh()
    {
        dgvClient.DataSource = null;
        cTable = new ClientTable_T();
        dgvClient.DataSource = cTable.GetList().Select(x => new {
x.Id_Client, x.Name_Client, x.MailClient, x.TelClient }).ToList();
        changeHeaders();
    }

    private void changeHeaders()
    {
        dgvClient.Columns["Id_Client"].HeaderText = "ז.י.";
        dgvClient.Columns["Name_Client"].HeaderText = "שם";
        dgvClient.Columns["MailClient"].HeaderText = "מייל";
        dgvClient.Columns["TelClient"].HeaderText = "טלפון";
    }

    private void button1_Click(object sender, EventArgs e)
    {
        frmOneClient frm = new frmOneClient();
        frm.ShowDialog();
        refresh();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        if (c.Id_Client == 0)
            MessageBox.Show("לקוח נבחר לא");
        else
        {
            frmOneClient frm = new frmOneClient(c, eState.עדכון);
            frm.ShowDialog();
            refresh();
        }
    }

    private void button3_Click(object sender, EventArgs e)
    {
        if (c.Id_Client == 0)
            MessageBox.Show("לקוח נבחר לא");
    }
}
```



```

        else
        {
            frmOneClient frm = new frmOneClient(c, eState.הצגה);
            frm.ShowDialog();
        }
    }

    private void dgvClient_RowHeaderMouseDoubleClick(object sender,
DataGridViewCellEventArgs e)
    {
        int code =
Convert.ToInt32(dgvClient.SelectedRows[0].Cells[0].Value);
        c = cTable.GetList().FirstOrDefault(x => x.Id_Client == code);
    }

    private void txtfind_TextChanged(object sender, EventArgs e)
    {
        dgvClient.DataSource = cTable.GetList().Where(x =>
x.Name_Client.ToString().Contains(txtfind.Text)).Select(x => new { x.Id_Client,
x.Name_Client, x.MailClient, x.TelClient }).ToList();
    }
}

```

~~~~~

טופס זה מחולק לשני חלקים

1. הוספה ועדכון סוג
2. חיפוש סוג מהמאגר.

```

public partial class frmKindTable : Form
{
    BLL.KindTable_T kTable;
    BLL.KindTable k;
    public frmKindTable()
    {
        InitializeComponent();
        kTable = new BLL.KindTable_T();
        k= new BLL.KindTable();
        rbadd.Checked = true;
        refresh();
    }

    private void frmKindTable_Load(object sender, EventArgs e)
    {
    }

    public void refresh()
    {
        dgvKind.DataSource = kTable.GetList().Select(x => new {
x.Code_Kind, x.Name_kind }).ToList();
        ChangeHeaders();
    }

    private void ChangeHeaders()
    {
        dgvKind.Columns["Code_Kind"].HeaderText = "סוג קוד";
        dgvKind.Columns["Name_kind"].HeaderText = "סוג שם";
    }
}

```



```

    }
    private void rbadd_CheckedChanged(object sender, EventArgs e)
    {
        if (rbadd.Checked == true)
        {
            gbMatzav.Text = "הוספה";
            txtname.Text = "";
        }
    }
    private void rbedkun_CheckedChanged(object sender, EventArgs e)
    {
        if (rbedkun.Checked == true)
        {
            gbMatzav.Text = "עדכון";
            txtname.Text = "";
        }
    }

    public bool Check()
    {
        bool ok = true;
        try
        {
            k.Name_kind = txtname.Text;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            ok = false;
        }
        return ok;
    }
    private void btnok_Click(object sender, EventArgs e)
    {
        if (kTable.GetList().Any(x => x.Name_kind == txtname.Text) == true)
            MessageBox.Show("קיים הסוג");
        else
            if (Check())
            {
                if (rbadd.Checked == true)
                {
                    k.Code_Kind = kTable.GetNewKey();
                    kTable.Add(k);
                    k = new BLL.KindTable();
                    MessageBox.Show("בהצלחה נוסף הסוג");
                }
                if (rbedkun.Checked == true)
                {
                    kTable.Update(k);
                    MessageBox.Show("בהצלחה עודכן הסוג");
                }
                txtname.Text = "";
                refresh();
            }
    }
    private void dgvAuthors_RowHeaderMouseDoubleClick(object sender,
    DataGridViewCellMouseEventArgs e)
    {

```



```

        if (rbedkun.Checked == true)
        {
            int code =
Convert.ToInt32(dgvKind.SelectedRows[0].Cells[0].Value);
            k = kTable.GetList().FirstOrDefault(x => x.Code_Kind == code);
            txtname.Text = k.Name_kind;
        }
    }

    private void txtfind_TextChanged(object sender, EventArgs e)
    {
        dgvKind.DataSource = kTable.GetList().Where(x =>
x.Name_kind.Contains(txtfind.Text)).Select(x => new { x.Code_Kind, x.Name_kind
}).ToList();
    }
}

~~~~~

```

טופס תפריט וניתוב אל החלונות הבאים:

עובדים ספרים, לקוחות, הוספת לקוח חדש, השאלות ועוד

```

public partial class frmNav : Form
{
    EmploiesTable e;
    public frmNav()
    {
        InitializeComponent();
    }
    public frmNav(EmploiesTable e)
    {
        InitializeComponent();
        this.e = e;
        //if(e.Kind_Client=="משתמש")

    }

    private void frmNav_Load(object sender, EventArgs e)
    {

    }

    private void employee_Click(object sender, EventArgs e)
    {
        frmEmployeeTable frm = new frmEmployeeTable();
        frm.ShowDialog();
    }

    private void books_Click(object sender, EventArgs e)
    {
        frmBooksTable frm = new frmBooksTable();
        frm.ShowDialog();
    }

    private void clients_Click(object sender, EventArgs e)
    {

```



```

        frmClientTable frm = new frmClientTable();
        frm.ShowDialog();
    }

    private void newClient_Click(object sender, EventArgs e)
    {
        frmOneClient frm = new frmOneClient();
        frm.ShowDialog();
    }

    private void waitings_Click(object sender, EventArgs e)
    {
        frmBrrowTable frm = new frmBrrowTable();
        frm.ShowDialog();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        frmKindTable frm = new frmKindTable();
        frm.ShowDialog();
    }

    private void button1_Click(object sender, EventArgs e)
    {
        frmAuthorTable frm = new frmAuthorTable();
        frm.ShowDialog();
    }
}

```

~~~~~

טופס ספר יחיד:

טופס זה מחולק לשלושה חלקים:

1. טופס למילוי פרטי ספר חדש להוספה

2. שניים מהחלקים לא יפתחו תמיד אלא רק במצב של השאלה חדשה:  
רשימת הממתינים עבור כל ספר

3. אפשרות להוסיף ממתין לאחד מהספרים.

```

public partial class frmOneBook : Form
{
    BooksTable b;
    BooksTable_T bTble;
    WaitingTable_T wTable;
    WaitingTable w;
    ClientTable_T cTable;
    eState state;
    public frmOneBook()
    {
        InitializeComponent();
        b = new BooksTable();
        bTble = new BooksTable_T();
        wTable = new WaitingTable_T();
        w = new WaitingTable();
        state = eState.חדש;
    }
}

```



```

cTable = new ClientTable_T();

//אוטומטי מספור הוא הראשי המפתח אם רק - הראשי המפתח הוספת
b.code_Book = bTble.GetNewKey();
//בטופס החדש הקוד הצגת
lblCode.Text = b.code_Book.ToString();

//אוטומטי מספור הוא הראשי המפתח אם רק - הראשי המפתח הוספת
w.code_Waiting = wTable.GetNewKey();
w.StatusWaiting = true;

KindTable_T bt = new KindTable_T();
cmbCodeKind.DataSource = bt.GetList();
AuthorsTable_T at = new AuthorsTable_T();
cmbCodeAuthor.DataSource = at.GetList();

}

public frmOneBook(BooksTable b, eState s)
{
    InitializeComponent();
    this.b = b;
    state = s;

    bTble = new BooksTable_T();
    wTable = new WaitingTable_T();
    w = new WaitingTable();

    cTable = new ClientTable_T();

    //הפרטים את לראות כדי רק לטופס מגיע
    if (s == eState.הצגה)
    {
        fillTextBox();
        fillEnabled();
    }
    else if (s == eState.עדכון)
    {
        fillTextBox();

        KindTable_T kt = new KindTable_T();
        cmbCodeKind.DataSource = kt.GetList();
        AuthorsTable_T at = new AuthorsTable_T();
        cmbCodeAuthor.DataSource = at.GetList();
        ClientTable_T ct = new ClientTable_T();
        comboBox1.DataSource = ct.GetList();
    }
    else if (s == eState.השאלה הוסף)
    {
        fillTextBox();
        fillEnabled();
        ClientTable_T ct = new ClientTable_T();
        comboBox1.DataSource = ct.GetList();
        listView1.Visible = true;
        comboBox1.Visible = true;
        button3.Visible = true;
        foreach (var item in
wTable.GetList().Where(x=>x.Code_Book==b.code_Book&&x.StatusWaiting==true))

```





```

        {

            ListViewItem lvi = new ListViewItem(new string[] {

                item.code_Waiting.ToString(),

                cTable.GetList().FirstOrDefault(x => x.Id_Client ==
item.Code_Client).Name_Client

            });
            listView1.Items.Add(lvi);

        }

    }

    // שינוי אפשרות לללא הפקדים כל את שמגדירה פעולה//
    public void fillEnabled()
    {
        lblCode.Enabled = false;
        cmbCodeKind.Enabled = false;
        cmbCodeAuthor.Enabled = false;
        textBox1.Enabled = false;
        button1.Enabled = false;

    }

    // עדכון של במצב בטופס הפקדים את שממלאה פעולה//
    public void fillTextBox()
    {
        lblCode.Text = b.code_Book.ToString();

        textBox1.Text = b.Name_Book;
        cmbCodeKind.Text = b.GetKind().Name_kind;
        cmbCodeAuthor.Text = b.GetAuthor().Name_Author;

        List<string> k = new List<string>();
        foreach (KindTable item in cmbCodeKind.Items)
        {
            k.Add(((KindTable)item).Name_kind);
        }
        List<string> a = new List<string>();
        foreach (AuthorsTable item in cmbCodeAuthor.Items)
        {
            k.Add(((AuthorsTable)item).Name_Author);
        }
        // cmbCodeKind.SelectedIndex = k.IndexOf(b.GetKind().Name_kind);

    }

    // המחלקה לתוך מהטופס הפקדים את שמעדכנת פעולה//
    public bool Check()
    {
        bool ok = true;
        errorProvider1.Clear();

        try
        {
            b.Name_Book = textBox1.Text;

```



```

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox1, ex.Message);
        ok = false;
    }
    try
    {
        b.code_Book = Convert.ToInt32(lblCode.Text);

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(lblCode, ex.Message);
        ok = false;
    }
    try
    {
        b.Code_Author =
        ((AuthorsTable)(cmbCodeAuthor.SelectedItem)).Code_Author;

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(cmbCodeAuthor, ex.Message);
        ok = false;
    }
    try
    {
        b.Code_Kind =
        ((KindTable)(cmbCodeKind.SelectedItem)).Code_Kind;

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(cmbCodeKind, ex.Message);
        ok = false;
    }

    return ok;
}

private void frmOneBook_Load(object sender, EventArgs e)
{
}

private void button1_Click(object sender, EventArgs e)
{
    if (Check() == true)
    {
        CopyTable_T ctable = new CopyTable_T();
        CopyTable c = new CopyTable();
        c.Code_Copy = ctable.GetNewKey();
        c.Code_Book = b.code_Book;
    }
}

```



```

c.Isborrow = false;
if (state == eState.חדש)
{
    try
    {
        bTble.Add(b);
        ctable.Add(c);
        MessageBox.Show("בהצלחה נשמר הספר");
        this.Close();
    }
    catch
    {
        MessageBox.Show("הספר בשמירת בעייה");
    }
}
else if (state == eState.עדכון)
{
    try
    {
        bTble.Update(b);
        MessageBox.Show("בהצלחה עודכן הספר");

        this.Close();
    }
    catch (Exception e1)
    {
        MessageBox.Show("הספר בעדכון בעייה");
    }
}
else if (state == eState.השאלה_הוסף)
{
    try
    {
        wTable.Add(w);
        MessageBox.Show("בהצלחה נוסף ממתין");

        this.Close();
    }
    catch (Exception e1)
    {
        MessageBox.Show("ממתין בהוספת בעיה");
    }
}
}

}

private void button3_Click(object sender, EventArgs e)
{
    try
    {
        w.Code_Client =
        ((ClientTable)(comboBox1.SelectedItem)).Id_Client;
        w.Code_Book = Convert.ToInt32(lblCode.Text);
        w.StatusWaiting = true;
        wTable.Add(w);
        MessageBox.Show("בהצלחה נוסף ממתין");
        this.Close();
    }
}

```



## טופס עבור הוספת לקוח חדש:

43



```

        // אוטומטי מספור הוא הראשי המפתח אם רק - הראשי המפתח הוספת
        c.Id_Client = cTble.GetNewKey();
        // בטופס החדש הקוד הצגת
        lblCode.Text = c.Id_Client.ToString();

    }

    public frmOneClient(ClientTable em, eState s)
    {
        InitializeComponent();
        c = new ClientTable();
        cTble = new ClientTable_T();
        c = em;
        state = s;

        // הפרטים את לראות כדי רק לטופס מגיע
        // הפרטים את לראות כדי רק לטופס מגיע
        if (s == eState.הצגה)
        {
            fillTextBox();
            fillEnabled();
        }
        else
            fillTextBox();

    }
    // שינוי אפשרות לללא הפקדים כל את שמגדירה פעולה
    public void fillEnabled()
    {
        lblCode.Enabled = false;
        textBox1.Enabled = false;
        textBox2.Enabled = false;
        textBox3.Enabled = false;

    }
    // עדכון של במצב בטופס הפקדים את שממלאת פעולה
    public void fillTextBox()
    {
        lblCode.Text = c.Id_Client.ToString();
        textBox1.Text = c.Name_Client;
        textBox2.Text = c.MailClient;
        textBox3.Text = c.TelClient;

    }
    // המחלקה לתוך מהטופס הפקדים את שמעדכנת פעולה
    public bool Check()
    {
        bool ok = true;
        errorProvider1.Clear();

        try
        {
            c.Id_Client = Convert.ToInt32( lblCode.Text);

        }
        catch (Exception ex)
        {
            errorProvider1.SetError(lblCode, ex.Message);
        }
    }

```



```

        ok = false;
    }
    try
    {
        c.Name_Client = textBox1.Text;

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox1, ex.Message);
        ok = false;
    }
    try
    {
        c.MailClient = textBox2.Text;

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox2, ex.Message);
        ok = false;
    }
    try
    {
        c.TelClient = textBox3.Text;
    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox3, ex.Message);
        ok = false;
    }

    return ok;
}

private void btnOk_Click(object sender, EventArgs e)
{
    if (Check() == true)
    {
        if (state == eState.חדש)
        {
            try
            {
                cTble.Add(c);

                MessageBox.Show("בהצלחה נשמר הלקוח");
                this.Close();
            }
            catch
            {
                MessageBox.Show("הלקוח בשמירת בעייה");
            }
        }
        if (state == eState.עדכון)
        {
            try
            {
                cTble.Update(c);
                MessageBox.Show("בהצלחה עודכן הלקוח");
            }
            catch
            {
                MessageBox.Show("הלקוח בשמירת בעייה");
            }
        }
    }
}

```



```

        this.Close();
    }
    catch (Exception e1)
    {
        MessageBox.Show("הלוקח בעדכון בעייה");
    }
}

}

private void frmOneClient_Load(object sender, EventArgs e)
{
}

private void textBox2_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsLetter(e.KeyChar) && !char.IsDigit(e.KeyChar)&&
e.KeyChar != ' ' && e.KeyChar != '\b' && e.KeyChar != '@' && e.KeyChar != '.')
        e.Handled = true;
}

private void textBox3_KeyPress(object sender, KeyPressEventArgs e)
{
    if (char.IsLetter(e.KeyChar) && e.KeyChar != '-' && e.KeyChar !=
'\b')
        e.Handled = true;
}

}

```

~~~~~

טופס להוספת עובד חדש למערכת:

```

public partial class frmOneEmployee : Form
{
    EmploiesTable e;
    EmploiesTable_T eTble;
    eState state;
    public frmOneEmployee()
    {
        InitializeComponent();
        e = new EmploiesTable();
        eTble = new EmploiesTable_T();
        state = eState.חדש;

        //אוטומטי מספור הוא הראשי המפתח אם רק - הראשי המפתח הוספת //
        e.Id_Employ = eTble.GetNewKey();
        // בטופס החדש הקוד הצגת //
        lblCode.Text = e.Id_Employ.ToString();
    }
    public frmOneEmployee(EmploiesTable em, eState s)
    {
        InitializeComponent();
        e = em;
        eTble = new EmploiesTable_T();
        state = s;
        //הפרטים את לראות כדי רק לטופס מגיע //
        if (s == eState.הצגה)
        {

```



```
fillTextBox();
fillEnabled();

}
else if (s == eState.עדכון)
fillTextBox();
}
// שינוי אפשרות לללא הפקדים כל את שמגדירה פעולה
public void fillEnabled()
{
    lblCode.Enabled = false;

    textBox1.Enabled = false;
    textBox2.Enabled = false;
    textBox3.Enabled = false;
    textBox4.Enabled = false;
    cmbkind.Enabled = false;
    btnOk.Enabled = false;

}
// עדכון של במצב בטופס הפקדים את שממלאה פעולה
public void fillTextBox()
{
    lblCode.Text = e.Id_Employ.ToString();
    textBox1.Text = e.Name_Employ;
    textBox2.Text = e.Mail_Employ;
    textBox3.Text = e.Tel_Employ;
    textBox4.Text = e.Password_Client;
    // textBox4.UseSystemPasswordChar = false;
    cmbkind.Text = e.Kind_Client;

}
// המחלקה לתוך מהטופס הפקדים את שמעדכנה פעולה
public bool Check()
{
    bool ok = true;
    errorProvider1.Clear();

    try
    {
        e.Name_Employ = textBox1.Text;

    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox1, ex.Message);
        ok = false;
    }

    try
    {
        e.Mail_Employ = textBox2.Text;
    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox2, ex.Message);
        ok = false;
    }
    try
    {

```





```

        e.Tel_Employ = textBox3.Text;
    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox3, ex.Message);
        ok = false;
    }
    try
    {
        e.Password_Client = textBox4.Text;
    }
    catch (Exception ex)
    {
        errorProvider1.SetError(textBox4, ex.Message);
        ok = false;
    }
    try
    {
        e.Kind_Client = cmbkind.Text;
    }
    catch (Exception ex)
    {
        errorProvider1.SetError(cmbkind, ex.Message);
        ok = false;
    }

    return ok;
}

private void frmOneEmployee_Load(object sender, EventArgs e)
{
}

private void btnOk_Click_1(object sender, EventArgs e)
{
    if (Check() == true)
    {
        if (state == eState.חדש)
        {
            try
            {
                eTble.Add(this.e);

                MessageBox.Show("בהצלחה נשמר העובד");
                this.Close();
            }
            catch (Exception e1)
            {
                MessageBox.Show("העובד בשמירת בעייה");
            }
        }
        if (state == eState.עדכון)
        {

```



```

        try
        {
            eTble.Update(this.e);
            MessageBox.Show("בהצלחה עודכן העובד");

            this.Close();
        }
        catch (Exception e1)
        {
            MessageBox.Show("העובד בעדכון בעייה");
        }
    }

}

private void textBox3_KeyPress(object sender, KeyPressEventArgs e)
{
    if (char.IsLetter(e.KeyChar) && e.KeyChar != '-' && e.KeyChar !=
'\b')
        e.Handled = true;
}

private void textBox2_KeyPress_1(object sender, KeyPressEventArgs e)
{
    if (!char.IsLetter(e.KeyChar) && !char.IsDigit(e.KeyChar) &&
e.KeyChar != ' ' && e.KeyChar != '\b' && e.KeyChar != '@' && e.KeyChar != '.')
        e.Handled = true;
}

private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsLetter(e.KeyChar) && e.KeyChar != ' ' && e.KeyChar !=
'\b')
        e.Handled = true;
}

}
~~~~~

```

טופס מסך פתיחה לאתר:

```

public partial class frmOpenLibrary : Form
{
    public frmOpenLibrary()
    {
        InitializeComponent();
        timer1.Start();
        timer2.Start();
    }

    private void timer1_Tick(object sender, EventArgs e)
    {
        timer2.Stop();

        timer1.Stop();
        frmPassword frm = new frmPassword();
        frm.Show();
        this.Hide();
    }
}

```



```
}  
  
private void timer2_Tick(object sender, EventArgs e)  
{  
    if (progressBar1.Value+5 < 100)  
        progressBar1.Value += 5;  
    else  
        progressBar1.Value = 100;  
}  
}
```

~~~~~

טופס כניסה באמצעות סיסמא.

מחולק לשניים:

1. כניסת משתמש ואופצייה לשינוי לאחר מכן.
2. כניסת מנהל ע"י שם וסיסמא

```
//יחיד מחלקת מסוג  
EmploiesTable e;  
//רבים מחלקת מסוג  
EmploiesTable_T eTable;  
  
public frmPassword()  
{  
    InitializeComponent();  
    e = new EmploiesTable();  
    eTable = new EmploiesTable_T();  
}  
  
private void frmPassword_Load(object sender, EventArgs e)  
{  
  
}  
  
private void picOpen1_Click(object sender, EventArgs e)  
{  
    //הסיסמה את תראה  
    txtpassword1.UseSystemPasswordChar = false;  
    //הסגורה העין את תציג  
    picClose1.Visible = true;  
    //הפתוחה העין את תסתיר  
    picOpen1.Visible = false;  
}  
  
private void picClose1_Click(object sender, EventArgs e)  
{  
    //הסיסמה את תסתיר  
    txtpassword1.UseSystemPasswordChar = true;  
    //הפתוחה העין את תציג  
    picOpen1.Visible = true;  
    //הסגורה העין את תסתיר  
    picClose1.Visible = false;  
}  
}
```



```

private void btnOk1_Click(object sender, EventArgs e1)
{
    if (eTable.GetList().FirstOrDefault(x => x.Name_Employ ==
txtname1.Text && x.Password_Client == txtpassword1.Text) != null)
    {
        e = eTable.GetList().FirstOrDefault(x => x.Name_Employ ==
txtname1.Text && x.Password_Client == txtpassword1.Text);
        frmNav n = new frmNav(e);
        n.ShowDialog();
    }
    else
        MessageBox.Show("שגויים הסיסמה או משתמש השם");
}

private void btnOk2_Click(object sender, EventArgs e)
{
    // הסיסמה את ונשנה לעובד נישגש
    // סיסמה חוקי לפי תקינה שהסיסמה בודקת
    if (validates.checkPassword(txtNewPassword.Text) == true)
        // לסיסמה שווה סיסמה שאימות בודקת
        if (txtNewPassword.Text != txtVarifyPass.Text)
            MessageBox.Show("תואמות לא הסיסמאות");
        else
        {
            // שונתה שהסיסמה העובד את מעדכן תקין שהכל
            this.e.Password_Client = txtNewPassword.Text;
            eTable.Update(this.e);
            MessageBox.Show("בהצלחה עודכנה הסיסמה");
        }
}

private void btnAdd_Click_1(object sender, EventArgs e)
{
    this.e = eTable.GetList().FirstOrDefault(x => x.Name_Employ ==
txtname1.Text && x.Password_Client == txtpassword1.Text);
    if (this.e != null && this.e.Kind_Client == "מנהל")
    {
        gbAddPass.Visible = true;
        gbPassword.Visible = false;
    }
    else
        MessageBox.Show("המערכת מנהל ידי על רק עובד הוספת");
}

private void btnOk3_Click(object sender, EventArgs e)
{
    // למערכת חדש עובד של הוספה
    if (validates.checkPassword(txtPassword3.Text) == true)
    {
        // אוטומטי מספור
        this.e.Id_Employ = eTable.GetNewKey();
        this.e.Name_Employ = txtname2.Text;
        this.e.Password_Client = txtPassword3.Text;
        this.e.Kind_Client = cmbkind.Text;
        eTable.Add(this.e);
        MessageBox.Show("בהצלחה נוסף העובד");
        txtname2.Text = "";
        txtPassword3.Text = "";
    }
}

```



```

        cmbkind.Text = "";
    }

}

private void btnUser_Click(object sender, EventArgs e)
{
    gbAddPass.Visible = false;
    gbChangePass.Visible = false;
    gbPassword.Visible = true;
    btnAdd.Visible = false;
}

private void btnMennege_Click(object sender, EventArgs e)
{
    gbAddPass.Visible = true;
    gbChangePass.Visible = false;
    gbPassword.Visible = true;
    btnAdd.Visible = true;
}

private void picOpen2_Click(object sender, EventArgs e)
{
    //הסיסמה את תראה
    txtpassword2.UseSystemPasswordChar = false;
    //הסגורה העין את תציג
    picClose2.Visible = true;
    //הפתוחה העין את תסתיר
    picOpen2.Visible = false;
}

private void picClose2_Click(object sender, EventArgs e)
{
    //הסיסמה את תסתיר
    txtpassword2.UseSystemPasswordChar = true;
    //הפתוחה העין את תציג
    picOpen2.Visible = true;
    //הסגורה העין את תסתיר
    picClose2.Visible = false;
}

private void picOpen3_Click(object sender, EventArgs e)
{
    //הסיסמה את תראה
    txtNewPassword.UseSystemPasswordChar = false;
    //הסגורה העין את תציג
    picClose3.Visible = true;
    //הפתוחה העין את תסתיר
    picOpen3.Visible = false;
}

private void picOpen4_Click(object sender, EventArgs e)
{
    //הסיסמה את תראה
    txtVarifyPass.UseSystemPasswordChar = false;
    //הסגורה העין את תציג
    picClose4.Visible = true;
    //הפתוחה העין את תסתיר
    picOpen4.Visible = false;
}

private void picClose4_Click(object sender, EventArgs e)

```



```

{
    //הסיסמה את תסתיר/
    txtVarifyPass.UseSystemPasswordChar = true;
    //הפתוחה העין את תציג/
    picOpen4.Visible = true;
    //הסגורה העין את תסתיר/
    picClose4.Visible = false;
}

private void picClose3_Click(object sender, EventArgs e)
{
    //הסיסמה את תסתיר/
    txtNewPassword.UseSystemPasswordChar = true;
    //הפתוחה העין את תציג/
    picOpen3.Visible = true;
    //הסגורה העין את תסתיר/
    picClose3.Visible = false;
}

private void picOpen5_Click(object sender, EventArgs e)
{
    //הסיסמה את תראה/
    txtPassword3.UseSystemPasswordChar = false;
    //הסגורה העין את תציג/
    picClose5.Visible = true;
    //הפתוחה העין את תסתיר/
    picOpen5.Visible = false;
}

private void picClose5_Click(object sender, EventArgs e)
{
    //הסיסמה את תסתיר/
    txtPassword3.UseSystemPasswordChar = true;
    //הפתוחה העין את תציג/
    picOpen5.Visible = true;
    //הסגורה העין את תסתיר/
    picClose5.Visible = false;
}

private void btnChange_Click(object sender, EventArgs e)
{
    gbChangePass.Visible = true;
}
}

```

טופס הוספה + הצגת השאלה+ החזר

טופס זה מחולק לשני חלקים

1. הצגת ההשאלות בטבלה
2. הוספה, החזרה והצגת השאלה.

```

public partial class frmBorrowTable : Form
{
    //אובייקטים 2 הגדרת/

```



```

BorrowTable b;
BorrowTable_T bTable;
CopyTable_T cTable = new CopyTable_T();
public frmBorrowTable()
{
    InitializeComponent();
    b = new BorrowTable();
    bTable = new BorrowTable_T();
    refresh();
}
//ה את ממלאאדג
public void refresh()
{

    dgvBorrow.DataSource = null;
    bTable = new BorrowTable_T();
    dgvBorrow.DataSource =
bTable.GetList().Where(x1=>x1.Status==true).Select(x => new { x.Code_Borrow, x.
GetClient().Name_Client, x.GetEmploies().Name_Employ, x.Date_Borrow
}).ToList();
    changeHeaders();
}

private void changeHeaders()
{
    dgvBorrow.Columns["Code_Borrow"].HeaderText = "השאלה קוד";
    dgvBorrow.Columns["Name_Client"].HeaderText = "לקוח שם";
    dgvBorrow.Columns["Name_Employ"].HeaderText = "עובד שם";
    dgvBorrow.Columns["Date_Borrow"].HeaderText = "השאלה נתוני";
}

private void dgvBorrow_RowHeaderMouseDoubleClick(object sender,
DataGridViewCellMouseEventArgs e)
{
    if (dgvBorrow.SelectedRows[0].DefaultCellStyle.BackColor ==
Color.Pink)
        MessageBox.Show("כבר הוחזרה ההשאלה");
    else
    {
        int code =
Convert.ToInt32(dgvBorrow.SelectedRows[0].Cells[0].Value);
        b = bTable.GetList().FirstOrDefault(x => x.Code_Borrow ==
code);
    }
}

private void button1_Click(object sender, EventArgs e)
{
    frmBorrow frm = new frmBorrow();
    frm.ShowDialog();
    refresh();
}

private void button3_Click(object sender, EventArgs e)
{
    if (b.Code_Borrow == 0)
        MessageBox.Show("השאלה נבחרה לא");
}

```



```

else
{
    frmBorrow frm = new frmBorrow(b, eState.הצגה);
    frm.ShowDialog();
    refresh();
}
}

private void frmBorrowTable_Load(object sender, EventArgs e)
{
}

private void button4_Click(object sender, EventArgs e)
{
    BorrowsBooksTable_T bbTable = new BorrowsBooksTable_T();
    //BorrowTable b = new BorrowTable();
    CopyTable_T ccTable = new CopyTable_T();
    WaitingTable_T wTable = new WaitingTable_T();
    ClientTable_T cTable = new ClientTable_T();
    BooksTable_T boTable = new BooksTable_T();
    if (b.Code_Borrow == 0)
        MessageBox.Show("השאלה נבחרה לא");

    else
    {
        foreach (var item in bbTable.GetList().Where(x => x.Code_Borrow
== b.Code_Borrow&& x.Status==true))
        {
            CopyTable c = ccTable.GetList().FirstOrDefault(x =>
x.Code_Copy == item.Code_Copy);
            c.Isborrow = false;
            ccTable.Update(c);

            MessageBox.Show("למאגר הוחזר הספר");
            int code = ccTable.GetList().FirstOrDefault(x =>
x.Code_Copy == item.Code_Copy).Code_Book;
            WaitingTable W = wTable.GetList().FirstOrDefault(x =>
x.StatusWaiting == true && code == x.Code_Book);
            if (W!=null) {
                MessageBox.Show("ללקוח להתקשר יש" +
cTable.GetList().FirstOrDefault(x => x.Id_Client ==
W.Code_Client).TelClient.ToString()
+ "פנוי לו שחיכה שהספר להודיע"
);
                W.StatusWaiting = false;
                wTable.Update(W);
            }
        }
        try
        {
            foreach (var item in
bbTable.GetList().Where(x=>x.Code_Borrow==b.Code_Borrow&&x.Status==true))
            {
                item.Status = false;
                bbTable.Update(item);
            }
            b.Status = false;
            bTable.Update(b);
            MessageBox.Show("נמחקה ההשאלה");
        }
    }
}

```





```

        refresh();
    }
    catch
    {
        MessageBox.Show("ההשאלה במחיקת בעיה");
    }
}

}
}

```

טופס השאלה יחידה מחולק ל:

1. נען טופס השאלה יחיד עם פרטי ההשאלה
2. רשימה בה אפשר להוסיף ספרים לאותה השאלה

```

BorrowTable b;
BorrowTable_T bTable;
eState state;
ClientTable_T cTable;
ClientTable c;
public frmBorrow()
{
    InitializeComponent();
    b = new BorrowTable();
    bTable = new BorrowTable_T();
    cTable = new ClientTable_T();
    c = new ClientTable();
    state = eState.חדש;
    button1.Visible = false;

    //אוטומטי מספור הוא הראשי המפתח אם רק - הראשי המפתח הוספת
    b.Code_Borrow = bTable.GetNewKey();
    //בטופס החדש הקוד הצגת
    lblCode.Text = b.Code_Borrow.ToString();

    comboBox1.DataSource = cTable.GetList();
    EmploiesTable_T et = new EmploiesTable_T();
    comboBox2.DataSource = et.GetList();
}

public frmBorrow(BorrowTable b, eState s)
{
    InitializeComponent();
    this.b = new BorrowTable();
    bTable = new BorrowTable_T();
    this.b = b;

    cTable = new ClientTable_T();
    c = new ClientTable();
    state = s;
    comboBox1.DataSource = cTable.GetList();
    EmploiesTable_T et = new EmploiesTable_T();
    comboBox2.DataSource = et.GetList();
}

```



```
// הפרטים את לראות כדי רק לטופס מגיע
if (s == eState.הצגה)
{
    fillTextBox();
    fillEnabled();
}

}
// שינוי אפשרות לללא הפקדים כל את שמגדירה פעולה
public void fillEnabled()
{
    lblCode.Enabled = false;
    comboBox1.Enabled = false;
    comboBox2.Enabled = false;
    dateTimePicker1.Enabled = false;
    button1.Enabled = false;
    button3.Enabled = false;
}
// עדכון של במצב בטופס הפקדים את שממלאת פעולה
public void fillTextBox()
{
    CopyTable_T ccTable = new CopyTable_T();
    BooksTable_T bTable = new BooksTable_T();
    //WaitingTable_T wTable = new WaitingTable_T();
    BorrowsBooksTable_T bbTable = new BorrowsBooksTable_T();

    foreach (var item in bbTable.GetList().Where(x => x.Code_Borrow ==
b.Code_Borrow).ToList())
        //השאלה שבאותה הספרים כל את מביא
        {
            ListViewItem lvi = new ListViewItem(new string[] {
                item.Code_Copy.ToString(),
                bTable.GetList().FirstOrDefault(x => x.code_Book ==
(ccTable.GetList().FirstOrDefault(x1 => x1.Code_Copy ==
item.Code_Copy).Code_Book)).Name_Book.ToString(),

            });
            listView1.Items.Add(lvi);
        }
    lblCode.Text = b.Code_Borrow.ToString();
    comboBox1.Text = b.GetClient().Name_Client;
    comboBox2.Text = b.GetEmploies().Name_Employ;

    dateTimePicker1.Text = b.Date_Borrow.ToString();

    List<string> c = new List<string>();
    foreach (ClientTable item in comboBox1.Items)
    {
        c.Add(((ClientTable)item).Name_Client);
    }
    List<string> e = new List<string>();
    foreach (EmploiesTable item in comboBox2.Items)
    {
        e.Add(((EmploiesTable)item).Name_Employ);
    }
    // cmbCodeKind.SelectedIndex = k.IndexOf(b.GetKind().Name_kind);
}
```



```

    }
    // המחלקה לתוך מהטופס הפקדים את שמערכנת פעולה
    public bool Check()
    {
        bool ok = true;
        errorProvider1.Clear();

        try
        {
            b.Code_Borrow = Convert.ToInt32( lblCode.Text);

        }
        catch (Exception ex)
        {
            errorProvider1.SetError(lblCode, ex.Message);
            ok = false;
        }
        try
        {
            b.code_Employ =
            ((EmploiesTable)(comboBox2.SelectedItem)).Id_Employ;

        }
        catch (Exception ex)
        {
            errorProvider1.SetError(comboBox2, ex.Message);
            ok = false;
        }
        try
        {
            b.Code_Client =
            ((ClientTable)(comboBox1.SelectedItem)).Id_Client;

        }
        catch (Exception ex)
        {
            errorProvider1.SetError(comboBox1, ex.Message);
            ok = false;
        }
        try
        {
            b.Date_Borrow = ((DateTime)(dateTimePicker1.Value));

        }
        catch (Exception ex)
        {
            errorProvider1.SetError(dateTimePicker1, ex.Message);
            ok = false;
        }

        return ok;
    }

    private void frmOneBook_Load(object sender, EventArgs e)
    {

```



```

    }

    private void button1_Click_1(object sender, EventArgs e)
    {

        if (Check() == true)
        {

            if (state == eState.חדש)
            {
                BorrowsBooksTable_T bbTable = new BorrowsBooksTable_T();
                BorrowsBooksTable bb = new BorrowsBooksTable();
                CopyTable_T ccTable = new CopyTable_T();
                BooksTable_T boTable = new BooksTable_T();
                BorrowTable_T bTable = new BorrowTable_T();

                try
                {
                    for (int i = 0; i < listView1.Items.Count; i++)
                    {
                        bb.Code_Bookinborrow = bbTable.GetNewKey();
                        bb.Code_Borrow = Convert.ToInt32(lblCode.Text);
                        bb.Code_Copy =
Convert.ToInt32(listView1.Items[i].SubItems[0].Text);
                        bb.Status = true;

                        bbTable.Add(bb);
                        CopyTable c = ccTable.GetList().FirstOrDefault(x =>
x.Code_Copy == bb.Code_Copy);
                        c.Isborrow = true;
                        ccTable.Update(c);
                    }
                }
                catch (Exception)
                {
                    MessageBox.Show("ההשאלה בשמירת בעייה");
                }

                try
                {
                    b.Status = true;
                    bTable.Add(b);

                    MessageBox.Show("בהצלחה נוספה ההשאלה");
                    this.Close();
                }
                catch
                {
                    MessageBox.Show("ההשאלה בשמירת בעייה");
                }
            }

        }
    }
}

```



טופס הוסף השאלה מחולק ל-3:

1. טבלה ובה כל הספרים שאינם מושאלים עדיין
2. רשימה בה יהיו כל הספרים שהמשתמש יבחר להשאיל מהטבלה.
3. חיפוש ספר מסוים או ספרים המכילים את הכיתוב שיכניס הלקוח בתיבת חיפוש



```

public partial class frmAddBorrow : Form
{
    BooksTable b;
    BooksTable_T bTable;
    CopyTable_T cTable;
    public frmAddBorrow()
    {
        InitializeComponent();
        b = new BooksTable();
        bTable = new BooksTable_T();
        cTable = new CopyTable_T();
        refresh();
    }

    private void refresh()
    {
        dgvBorrow.DataSource = null;
        BooksTable_T bTable = new BooksTable_T();
        List<CopyTable> cList = cTable.GetList().Where(x => x.Isborrow ==
false).ToList();

        List<BooksTable> boList = new List<BooksTable>();

        foreach (var item in bTable.GetList())
        {
            if (cList.Any(x => x.Code_Book == item.code_Book))
                boList.Add(item);
        }

        dgvBorrow.DataSource = boList.Select(x => new { x.code_Book,
Code_Author= x.GetAuthor().Name_Author, Code_Kind= x.GetKind().Name_kind,
x.Name_Book }).ToList();
        changeHeaders();
    }
    private void changeHeaders()
    {
        dgvBorrow.Columns["code_Book"].HeaderText = "ספר קוד";
        dgvBorrow.Columns["Code_Author"].HeaderText = "שם";
        dgvBorrow.Columns["Code_Kind"].HeaderText = "קטגוריה";
        dgvBorrow.Columns["Name_Book"].HeaderText = "ספר שם";
    }

    private void button1_Click(object sender, EventArgs e)
    {
        if(b.code_Book==0)
            MessageBox.Show("ספר נבחר לא");
        else
        {
            cTable = new CopyTable_T();

            ListViewItem lvi = new ListViewItem(new string[] {
                cTable.GetList().First(x=>x.Code_Book==b.code_Book&&
x.Isborrow==false).Code_Copy.ToString(),

                b.Name_Book,
            });
            listView1.Items.Add(lvi);
            dgvBorrow.SelectedRows[0].DefaultCellStyle.BackColor =
Color.Pink;
        }
    }
}

```



```

    }

    private void dgvBorrow_RowHeaderMouseDoubleClick(object sender,
DataGridViewCellEventArgs e)
    {
        if (dgvBorrow.SelectedRows[0].DefaultCellStyle.BackColor ==
Color.Pink)
            MessageBox.Show("מושאל הספר");
        else
        {
            int code =
Convert.ToInt32(dgvBorrow.SelectedRows[0].Cells[0].Value);
            b = bTable.GetList().FirstOrDefault(x => x.code_Book == code);
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        dgvBorrow.SelectedRows[0].DefaultCellStyle.BackColor = Color.White;

        listView1.Items.Remove(listView1.SelectedItems[0]);
    }

    private void button3_Click(object sender, EventArgs e)
    {
        this.Close();
    }

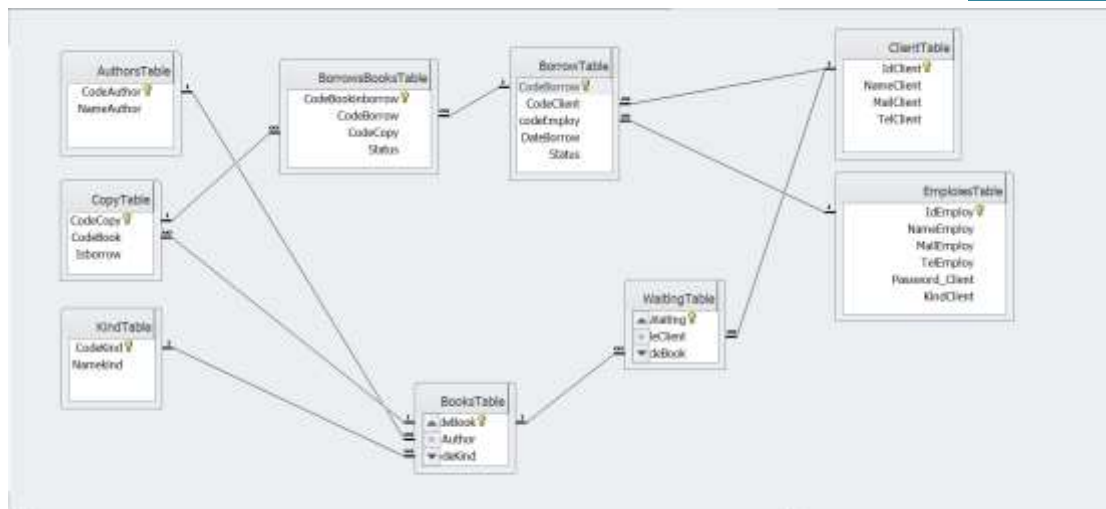
    private void txtfind_TextChanged(object sender, EventArgs e)
    {
        dgvBorrow.DataSource = bTable.GetList().Where(x =>
x.Name_Book.Contains(txtfind.Text)).Select(x => new { x.code_Book, Code_Author
= x.GetAuthor().Name_Author, Code_Kind = x.GetKind().Name_kind, x.Name_Book
}).ToList();
    }
}
}

```



## בסיס הנתונים

### קשרי גומלין



### טבלאות

#### טבלת סופרים

AuthorsTable		קשרי גומלין
סוג נתונים	שם שדה	
מספור אוטומטי	CodeAuthor	🔑
טקסט	NameAuthor	

#### טבלת ספרים

BooksTable		קשרי גומלין
סוג נתונים	שם שדה	
מספור אוטומטי	codeBook	🔑
מספר	CodeAuthor	
מספר	CodeKind	
טקסט	NameBook	

#### טבלת ספרים בהשאלה

BorrowsBooksTable		
סוג נתונים	שם שדה	
מספור אוטומטי	CodeBookinborrow	🔑
מספר	CodeBorrow	
מספר	CodeCopy	
כן/לא	Status	





## טבלת השאלות

BorrowTable	
סוג נתונים	שם שדה
מספור אוטומטי	CodeBorrow
מספר	CodeClient
מספר	codeEmploy
תאריך/שעה	DateBorrow
כן/לא	Status

## טבלת לקוחות

ClientTable	
סוג נתונים	שם שדה
מספור אוטומטי	IdClient
טקסט	NameClient
טקסט	MailClient
טקסט	TelClient

## טבלת עותקים

CopyTable	
סוג נתונים	שם שדה
מספור אוטומטי	CodeCopy
מספר	CodeBook
כן/לא	Isborrow

## טבלת עובדים

EmploiesTable	
סוג נתונים	שם שדה
מספור אוטומטי	IdEmploy
טקסט	NameEmploy
טקסט	MailEmploy
טקסט	TelEmploy
טקסט	Password_Client
טקסט	KindClient

## טבלת סוגים

KindTable	
סוג נתונים	שם שדה
מספור אוטומטי	CodeKind
טקסט	Namekind

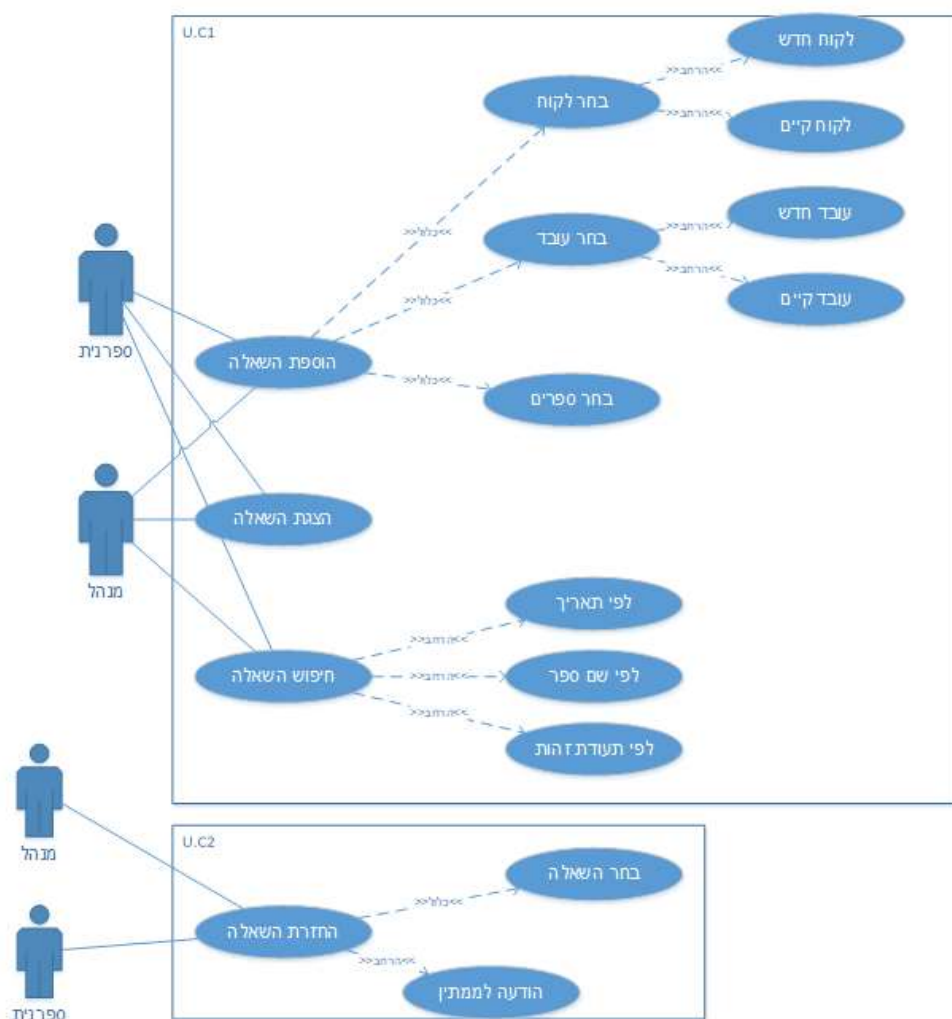
## טבלת ממתנים



WaitingTable	
סוג נתונים	שם שדה
מספור אוטומטי	codeWaiting
מספר	CodeClient
מספר	CodeBook
כן/לא	StatusWaiting



תרשימי use case:



U.C1	השאלה חדשה
הספרנית(קיימת או חדשה) מוסיף השאלת ספר ללקוח קיים או חדש.	תיאור קצר:
שההשאלה הזו חדשה,הלקוח בחר ספר/ים והספר אינו מושאל כבר למישהו אחר.	תנאי קודם:
עובר / מנהל	שחקנים מעורבים:
הלקוח מעוניין להשאל את הספר	הזנק:



תנאי סיום:	הספר הושאל ללקוח
תדירות:	מספר פעמים ביום

U.C1	הצגת השאלה
תיאור קצר:	הספרנית מציגה את פרטי השאלה מסוימת
תנאי קודם:	ההשאלה קיימת במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	הספרנית מעוניינת לצפות בפרטים
תנאי סיום:	פרטי ההשאלה מוצגים על המסך
תדירות:	מספר פעמים ביום

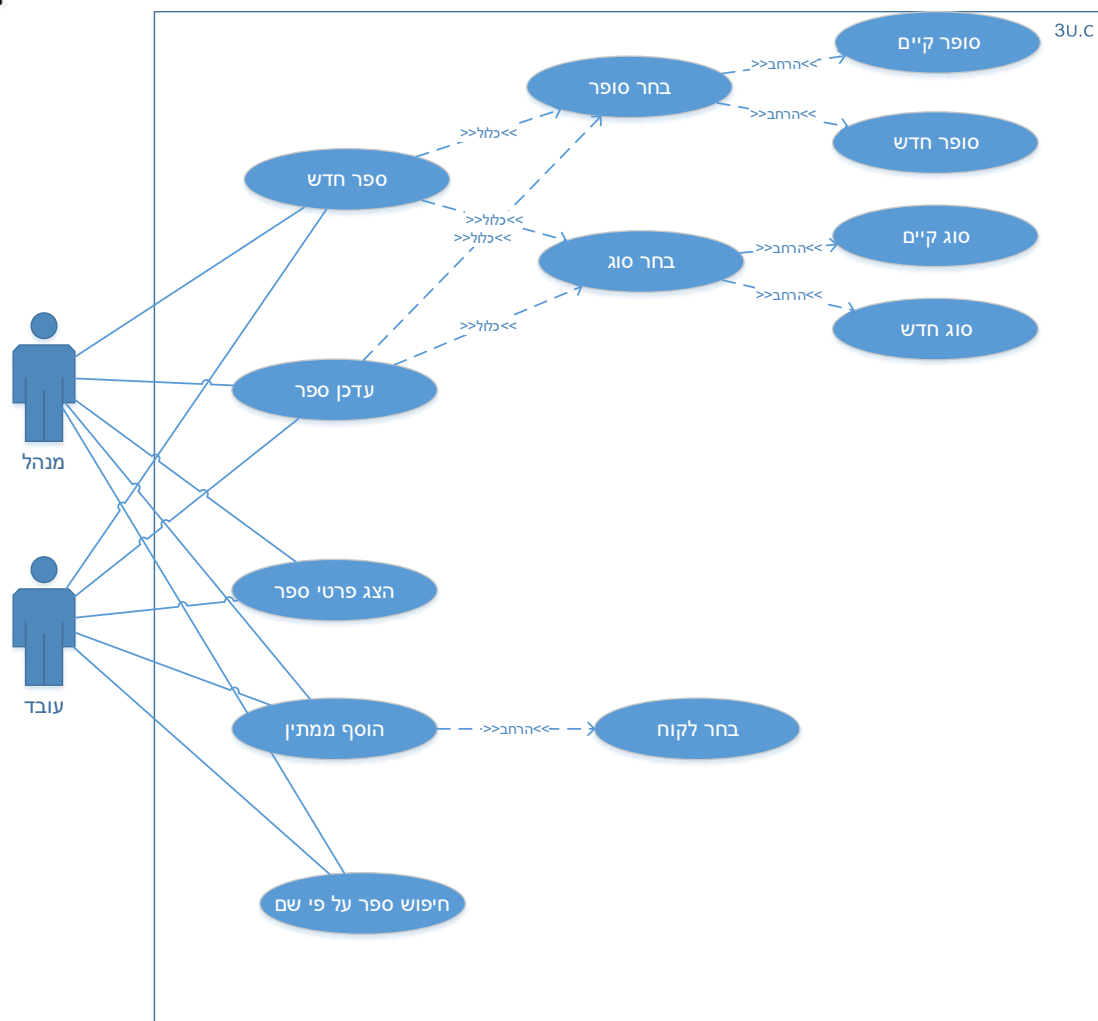
U.C1	חיפוש השאלה
תיאור קצר:	העובד מחפש השאלה מסוימת לפי פרמטרים מסוימים
תנאי קודם:	ההשאלה קיימת במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	העובד מעוניין למצוא השאלה מהמאגר
תנאי סיום:	ההשאלה המבוקשת נמצאה
תדירות:	מספר פעמים ביום

U.C2	הודעה לממתין
תיאור קצר:	הספרנית תודיע ללקוח שהמתין לספר שחזר כי הוא פנוי
תנאי קודם:	יש ממתנים על הספר שהוחזר
שחקנים מעורבים:	עובד / מנהל



<b>הזנק:</b>	הלקוח מעוניין לדעת שהספר פנוי להשאלה עבור
<b>תנאי סיום:</b>	הלקוח ידע שהספר פנוי להשאלה עבורו
<b>תדירות:</b>	מספר פעמים ביום

<b>החזרת השאלה</b>	U.C2
<b>תיאור קצר:</b>	הלקוח מחזיר את הספר שלקח
<b>תנאי קודם:</b>	הלקוח ביצע השאלה על אותו הספר
<b>שחקנים מעורבים:</b>	עובד / מנהל
<b>הזנק:</b>	הלקוח מעוניין להחזיר את הספר שלקח
<b>תנאי סיום:</b>	הספר הוחזר למאגר
<b>תדירות:</b>	מספר פעמים ביום



U.C3	ספר חדש
תיאור קצר:	הספרנית תרשום את הספר כתפוס אצל הלקוח
תנאי קודם:	הספר קיים במאגר ולא מושאל כבר
שחקנים מעורבים:	עובד / מנהל
הזנק:	הלקוח מעוניין לקחת את הספר
תנאי סיום:	הספר יצא מהמאגר
תדירות:	מספר פעמים ביום



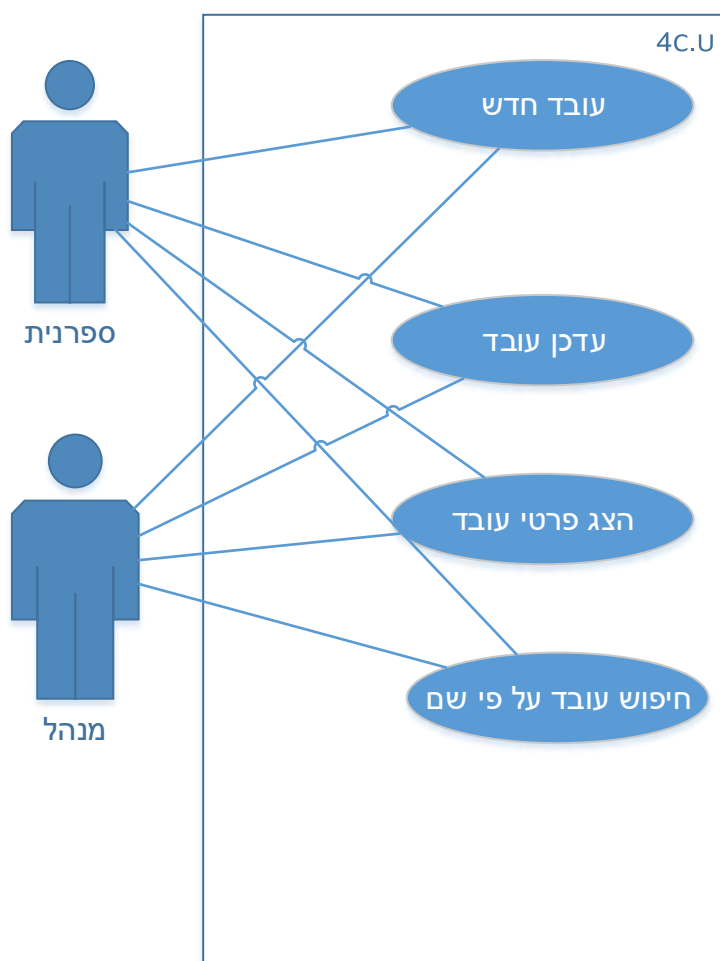
U.C3	הצג פרטי ספר
הספרנית מעוניינת לצפות בפרטי הספר הקיימים	תיאור קצר:
הספר קיים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
לצפות בפרטים הקיימים	הזנק:
פרטי הספר יוצגו על המסך	תנאי סיום:
מספר פעמים ביום	תדירות:

U.C3	הוסף ממתין לספר
הספרנית תוסיף לאחד הספרים המושאלים ממתין בתור	תיאור קצר:
הספר והלקוח קיימים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
אדם מעוניין להשאיל ספר שמושאל כבר למישהו אחר יועבר להמתנה	הזנק:
המתין נוסף לאותו ספר	תנאי סיום:
מספר פעמים ביום	תדירות:

U.C3	חיפוש ספר
הספרנית מעוניינת למצוא ספר מסוים	תיאור קצר:
הספר קיים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
הלקוח מעוניין למצוא ספר מסוים	הזנק:



תנאי סיום:	הספר מופיע בטבלה
תדירות:	מספר פעמים ביום



U.C4	עובד חדש
הספרנית תוסיף עובד חדש למאגר	תיאור קצר:
העובד לא קיים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
עובד חדש הגיע וצריך להוסיף אותו למאגר	הזנק:
העובד התווסף למאגר	תנאי סיום:



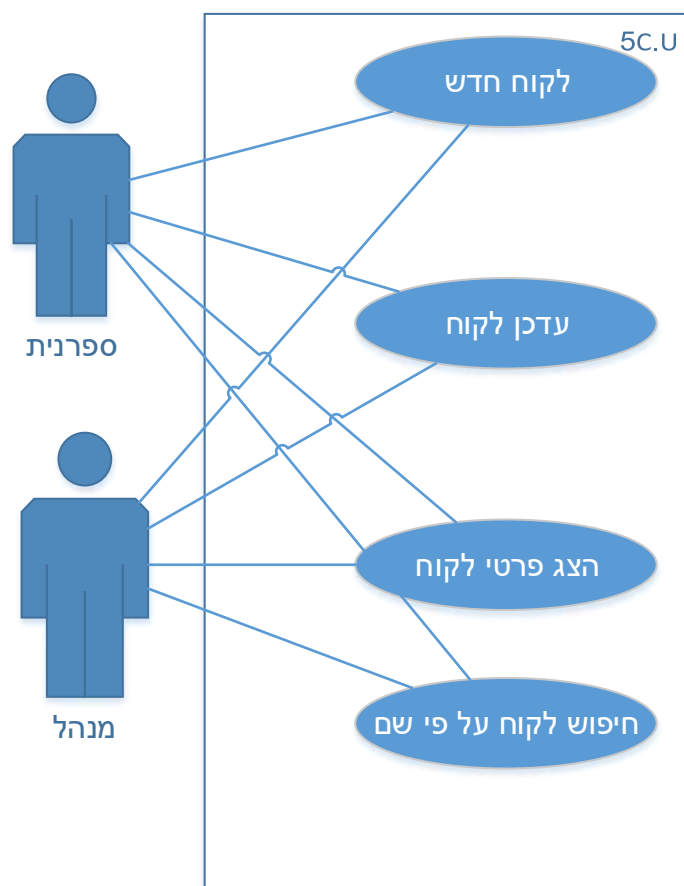


תדירות:	מספר פעמים ביום
---------	-----------------

עדכן עובד	U.C4
תיאור קצר:	הספרנית תעדכן את פרטי העובד במאגר
תנאי קודם:	העובד קיים במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	פרטי עובד השתנו
תנאי סיום:	פרטי העובד עודכנו במאגר
תדירות:	מספר פעמים ביום

הצג פרטי עובד	U.C4
תיאור קצר:	הספרנית מעוניינת לצפות בפרטי עובד
תנאי קודם:	העובד קיים במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	לצפות בפרטים הקיימים
תנאי סיום:	פרטי העובד יוצגו על המסך
תדירות:	מספר פעמים ביום

חיפוש עובד	U.C3
תיאור קצר:	הספרנית מעוניינת למצוא עובד מסוים על פי שמו
תנאי קודם:	העובד קיים במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	העובד/המנהל מעוניין למצוא פרטי עובד מסוים
תנאי סיום:	העובד מופיע בטבלה
תדירות:	מספר פעמים ביום



U.C5	לקוח חדש
הספרנית תוסיף לקוח חדש למאגר	תיאור קצר:
הלקוח לא קיים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
לקוח חדש הגיע וצריך להוסיף אותו למאגר	הזנק:
הלקוח התווסף למאגר	תנאי סיום:
מספר פעמים ביום	תדירות:

U.C5	עדכן לקוח
הספרנית תעדכן את פרטי הלקוח במאגר	תיאור קצר:
הלקוח קיים במאגר	תנאי קודם:
עובד / מנהל	שחקנים מעורבים:
פרטי לקוח השתנו	הזנק:



תנאי סיום:	פרטי הלקוח עודכנו במאגר
תדירות:	מספר פעמים ביום

U.C5	הצג פרטי עובד
תיאור קצר:	הספרנית מעוניינת לצפות בפרטי לקוח
תנאי קודם:	הלקוח קיים במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	לצפות בפרטים הקיימים
תנאי סיום:	פרטי הלקוח יוצגו על המסך
תדירות:	מספר פעמים ביום

U.C3	חיפוש לקוח
תיאור קצר:	הספרנית מעוניינת למצוא לקוח מסוים
תנאי קודם:	הלקוח קיים במאגר
שחקנים מעורבים:	עובד / מנהל
הזנק:	העובד/המנהל מעוניין למצוא לקוח מסוים
תנאי סיום:	פרטי הלקוח יופיעו בטבלה
תדירות:	מספר פעמים ביום

## מדריך למשתמש



ברוך הבא לתוכנת "הספרייה".

התוכנה נועדה לשרת את הספרנית בתחזוק הספרייה, בניהול מסודר של נתוני הספרים והלקוחות ובכך מקלה באופן משמעותי מהעומס הרב. האופציות העומדות בפני הספרנית הן רבות ובהן יש את האפשרות לעקוב אחר הספרים המושאלים והמערכת דואגת באופן אוטומטי שלא יהיה ניתן להשאיל עותק פעמיים, כמו כן המערכת דואגת להגינות בין הלקוחות הממתינים שלא יעוכבו זמן רב.

כדי להשתמש נכון בתוכנה עליך להכיר את הממשק הגרפי ולקרוא את ההוראות הבאות.

מאחלים לך בהצלחה ובהנאה, לך וללקוחותיך!!

### מסך כתיחה:



### טופס הזדהות:

משתמש ברמת מנהל חייב להכניס סיסמה ויהיה לו גישה לכל הטפסים, משתמש אחר ללא סיסמה מוגבל בגישה לחלק מהטפסים.

טופס פתיחה – טופס הקדמה, הטופס הבא יהיה באופן אוטומטי

בטופס זה המערכת מזהה את סוג המשתמש ובהתאם לכך נותנת לו גישות לפעולות שונות. כדי להתחבר עליך לבחור את סוג המשתמש ולהקיש את הסיסמא, בנוסף יש באפשרותך להחליף את הסיסמא.



### מופס ניווט:

אפשרות כניסה לכל הטפסים, טופס עובדים ספרים, לקוחות, הוספת לקוח חדש, השאלות... מסך זה מאפשר לך לנווט בין דפי התוכנה ולהגיע לכל פעולה רצויה. למעבר לעמודים אחרים יש להקיש על הכפתורים המתאימים.

### מופס עובדים

טופס המציג את כל הלקוחות שלנו, ניתן לסנן, לחפש, ניתן להוסיף לקוח חדש, לעדכן.



- בטבלה מוצגים כל העובדים הנמצאים במאגר הלקוחות.
- הוספת לקוח: ללחוץ על "חדש" -> למלא פרטים -> ללחוץ "שמור"
- עדכון – יש לסמן א השורה הרצויה -> ללחוץ "עדכן" -> למלא פרטים עדכניים -> ללחוץ "שמור"
- הצג – יש לסמן א השורה הרצויה -> ללחוץ "הצג"
- חיפוש עובש על פי השם המוכנס בתיבת החיפוש.

frmEmployeeTable

עובדים

ת.י	שם	מייל	טלפון	סוג
1	שירה	a0504173725@g	0504173725	מנהל
2	נוי שמחה	n@gmail.com	0504172070	משחמש
3	בן	b@gmail.com	0564321759	משחמש
4	אברהם	...av2324@gmail.c	0527156012	משחמש
5	גיל	g@gmail.com	0564326785	משחמש
6	בן שבע	b@gmail.com	0564326785	משחמש
7	יהודי	y@gmail.com	0548573245	מנהל

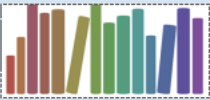
חפש עובד

חדש עדכן חזק

### טופס עובד יחיד

אל מסך זה מגיעים מהמסך הקודם ובו ממלאים את הפרטים או מעדכנים. כמו כן בהצגת הנתונים יפתח החלון הזה גם.





## עובד

<input type="text"/>	קוד
<input type="text"/>	שם פרטי
<input type="text"/>	מייל
<input type="text"/>	טלפון
<input type="text"/>	סיסמא
<input type="text"/>	סוג

גם כאן באופן דומה יעבדו גם מסכי עובד יחיד, ספר והשאלה.

### טופס לקוחות

- בטבלה מוצגים כל הלקוחות הנמצאים במאגר הלקוחות.
- הוספת לקוח: ללחץ על "חדש" -> למלא פרטים -> ללחץ "שמור"
- עדכון – יש לסמן א השורה הרצויה -> ללחץ "עדכן" -> למלא פרטים עדכניים -> ללחץ "שמור"
- הצג – יש לסמן א השורה הרצויה -> ללחץ "הצג"
- חיפוש לקוח על פי השם המוכנס בתיבת החיפוש.



frmClientTable

לְקוּחוֹת

ח.ז	שם	חייך	טל
2	נוי	n@gmail.com	0569874356
3	יראח	y@gmail.com	0456783223
7	מימי	m@gmail.com	0567654312
8	אביטל	a@gmail.com	0534108038
9	חווילת	t@gmail.com	0583239441
10	לי	l@gmail.com	0546732456
16	פיני	p@gmail.com	0546783245

חפש לקוח

חדש

עדכן

הצג





### טופס לקוח יחיד

אל מסך זה מגיעים מהמסך הקודם ובו ממלאים את הפרטים או מעדכנים. כמו כן בהצגת הנתונים יפתח החלון הזה גם.

### טופס ספרים

- בטבלה מוצגים כל הספרים הנמצאים במאגר הלקוחות.
- חיפוש: יש להכניס את השם המבוקש והספרים המתאימים יכללו בטבלה
- הוספת ספר: ללחוץ על "חדש" -> למלא פרטים -> ללחוץ "שמור"
- עדכון – יש לסמן א השורה הרצויה -> ללחוץ "עדכן" -> למלא פרטים עדכניים -> ללחוץ "שמור"
- הצג – יש לסמן א השורה הרצויה -> ללחוץ "הצג"
- הוסף ממתין-יש לסמן את השורה הרצויה-> ללחוץ על הוסף ממתין
- חיפוש ספר-יש לכתוב את השם המבוקש והתוצאות המתאימות יופיעו בטבלה.



frmBooksTable

**ספרים**

טוד ספר	Code_Author	Code_Kind	שם ספר
1	חיים נחמן ביאליק	ילדים	איסחור
14	דוד רפפורט	ילדים	21 בבית 1
15	דוד רפפורט	ילדים	21 בבית 1
17	יונה ספיר	נוער	מגן אנשי
18	יונה ספיר	מחנ	מגן אנשי
20	מאיה סינר	ילדים	לוחלוח כל סמס

חפש ספר

1

חדש עדכן חצב הוסף מחצין

### טופס ספר יחיד

אל מסך זה מגיעים מהמסך הקודם .

בצד הימני של החלון ממלאים את הפרטים או מעדכנים. כמו כן בהצגת הנתונים יפתח החלון הזה גם אך לא נראה את הצד השמאלי.

כאשר נגיע לכאן בשביל להוסיף ממתין לספר נראה גם את הצד השמאלי ונצטרך לבחור את הלקוח הממתין ->ללחץ על "הוסף ממתין".

frmBooksTable

**ספר**

קוד ספר

קוד סוג

קוד סוג

קוד סוג

שם

איסור

הוסף מחצין

קוד המחנה

שם לקוח

הוסף מחצין



### טופס השאלות

- בטבלה מוצגים כל ההשאלות הנמצאים במאגר הלקוחות.
- הוספת השאלה: ללחוץ על "חדש" -> למלא פרטים -> ללחוץ "שמור"
- הצג – יש לסמן א השורה הרצויה -> ללחוץ "הצג"
- החזר השאלה-יש לסמן את השורה הרצויה-> ללחוץ על הוסף השאלה

### טופס השאלה יחידה

אל מסך זה מגיעים מהמסך הקודם .

בצד הימני של החלון ממלאים את הפרטים. כמו כן בהצגת הנתונים יפתח החלון הזה .

כאשר נגיע לכאן בשביל להוסיף השאלה נצטרך ללחוץ על "הוסף ספר" שיפתח לנו את החלון הבא ואח"כ נלחץ על "הוסף".



## מופס השאלה יחידה

אל מסך זה מגיעים מהמסך הקודם .

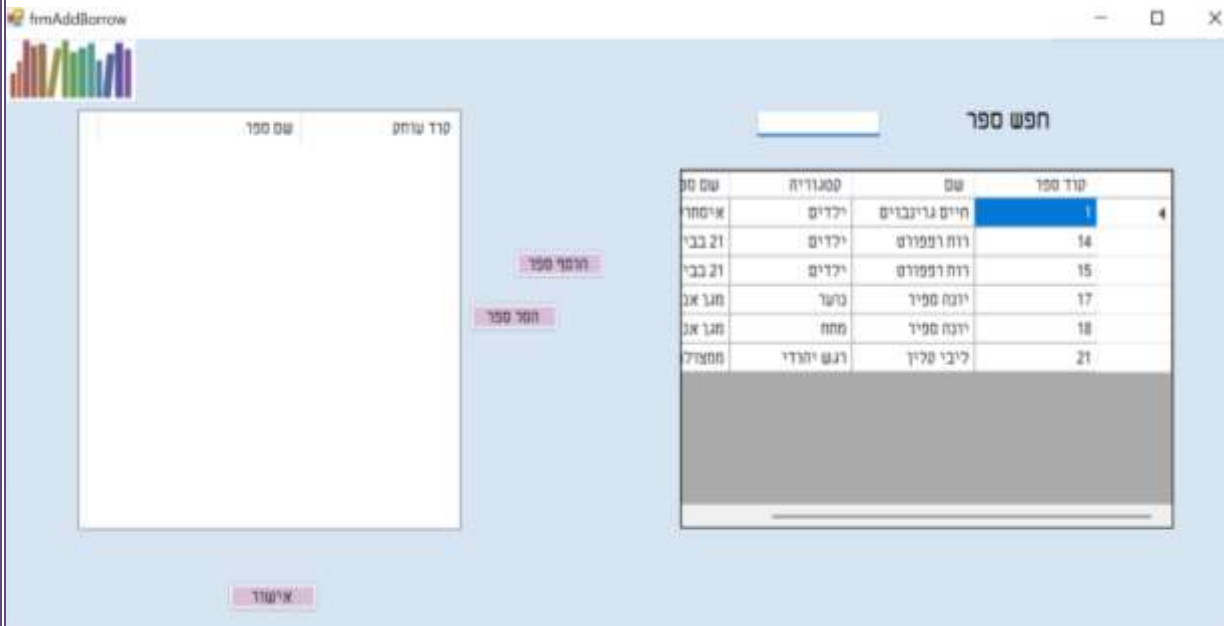
בצד הימני של החלון נראה טבלה ובה יוצגו כל הספרים שאינם מושאלים .

כדי להוסיף ספר להשאלה : נלחץ פעמיים בראש השורה בטבלה של הספר - < נלחץ על הוסף ספר. הספר יוצג בטבלה השמאלית.

כדי להסיר ספר: נלחץ פעמים על ראש השורה של הטבלה השמאלית המכילה את פרטי הספר שאנו רוצים להסיר - < נלחץ על "הסר ספר".

כשסיימנו לבחור ספרים להשאלה נלחץ על "אישור" ונגיע למסך הקודם.

על מנת לחפש ספר מסוים יש לכתוב את שמו בריבוע החיפוש.



## מופס סופרים

טופס המציג את כל הסופרים שלנו, ניתן לסנן , ניתן להוסיף סופר חדש, ולעדכן.

כדי להוסיף סופר נלחץ על "הוספה" - < נמלא את שם הסופר בצד - < נלחץ על "אישור".

כדי לעדכן סופר: נלחץ על "עדכון" - < נלחץ בראש השורה המכילה את פרטי הסופר המבוקש בטבלה - < נשנה את שם הסופר - < נלחץ על "אישור".

ניתן גם לחפש סופר מסוים עפ"י שמו.



### מופס סוגים

טופס המציג את כל הסוגים שלנו, ניתן לסנן, ניתן להוסיף סוג חדש, ולעדכן.

כדי להוסיף סוג נלחץ על "הוספה" - > נמלא את שם הסוג בצד - > נלחץ על "אישור".

כדי לעדכן סוג: נלחץ על "עדכון" - > נלחץ בראש השורה המכילה את פרטי הסוג המבוקש בטבלה - > נשנה את שם הסוג - > נלחץ על "אישור".

ניתן גם לחפש סוג מסוים עפ"י שמו.



## רפלקציה

לאורך כל מהלך הפרויקט הייתה אירה נהדרת של מחקר ועשייה פורה. לצד העבודה הקשה והשעות הרבות שמורים גם רגעים מהנים של סיפוק בתוצאות העשייה.

מאחר והפרויקט אכן גדול וישבנו עליו שעות לא מעטות, הייתי לוקחת איתי את תחושת הסיפוק והלמידה המהנה שיוצרת עשיית פרויקט שכזה, ששונה מלמידה אך מלמד לא פחות...

כגודל ההנאה כך גם היו מעט קשיים כמו עמידה במטלות רבות בזמן לחץ ולכן כמסקנה אילו הייתי מתחילה שוב הייתי מנסה בתחילת הזמן כשהתחלנו ממש את הפרויקט להתקדם עוד לבד כדי לעמוד בקצב ולא להיות בלחץ של הרגע האחרון...

לעניות דעתי ניתן בהחלט לתת לתלמידים לעשות את הפרויקט בזוגות מאחר והעבודה בו רבה ניתן בהחלט לחלק אותו ל-2 ואף אחד לא יישב משועמם. אך מה שברור הוא שהלמידה דרך פרויקט אישי שכזה הנה משמעותית, חויתית ומחכימה ביותר!!!

תודה רבה רבה!!!