

DESIGN

GRAPHIC STUDIO

שם העבודה: אפרת שמואלי- ניהול משרד גרפיקה

שם ביה"ס: תיכון בית יעקב רכסים

מגישה: אפרת שמואלי

ת.ז.: 327706123

שם המנחה: המורה אלה גליק

חלופה: שירותי אינטרנט, נתונים א-סינכרוניות

שנה: קיץ תשפ"ג

תאריך הגשה: יוני 2023

תוכן עניינים:

3.....	מבוא
4.....	ניתוח מערכת:
4.....	מסמך יזום:
7.....	ניתוח מצב קיים:
8.....	יעדי המערכת:
9.....	אילוצי המערכת:
24.....	בסיס הנתונים:
27.....	מבנה וארכיטקטורה:
27.....	מדריך למפתח:
30.....	שימוש ב- user control
31.....	מדריך למשתמש
44.....	רפלקציה:
45.....	ביבליוגרפיה:
46.....	נספחים

מבוא

שם הפרויקט: אפרת שמואלי- ניהול משרד גרפיקה.

הרקע לפרויקט:

שנים רבות עולם הגרפיקה והעיצוב מלווים אותנו. בילדות פגשנו אותם בעיקר בצורת כרטיסי ברכה, הזמנות לאירועים ואיחולי שנה טובה. כיום בעידן הטכנולוגיה, עולם הגרפיקה התפתח והשימוש בו גדל עם חלוף השנים. ע"י הגרפיקה ניתן לבטא אומנות, יצירתיות וטעם בצורה מושלמת.

בתור אחת שמחבבת גרפיקה, בחרתי לחבר בין עולם הגרפיקה והעיצוב לפרויקט שנדרשתי ליצור.

כאשר משתמשים ברישום ידני ביומן, דבר זה גורם לבעיות באימות נתונים שונים, כפילויות בפרטים וכן שגיאות בתאריכים.

מערכת ממוחשבת יכולה לעזור למשרד הגרפיקה לנהל את הזמן העבודה שלו בצורה יעילה.

מסיבה זו חשבתי לבנות את התוכנה הזו בכדי שהעבודה במשרד תתמקד בגרפיקה ותשאיר את משימות הניהול לתוכנה.

בנוסף לכך, רציתי לעסוק בנושא שאני אוהבת ויש לי ידע בו.

תהליך המחקר:

את המידע באיזו צורה לנהל את המשרד צברתי מחברתי הגרפיקאית שהסבירה לי את כל תהליך ניהול המשרד.

כאשר פיתחתי את התוכנה חיפשתי פתרון לבעיות שבהן נתקלה חברתי בעבר, וכן יישמתי עצות לייעול תהליך הניהול.

אתגרים:

אתגר שנתקלתי בו במהלך בניית התוכנה היה בבניית היומן, לא ידעתי באיזו צורה להציגו. לבסוף התקבלה החלטה על הרשאות הגישה לכל משתמש.

ניתוח מערכת:

מסמך יזום:

תיאור המערכת:

משרד הגרפיקה עוסק במתן שירותי עבודות גרפיקה שונות, כמו: מיתוג עסקים, עיבוד תמונות, וכן עיצוב במגוון אפשרויות: אלבומים, מודעות לעיתונות, קטלוגים, פליירים, הזמנות לאירועים ועוד...

עובדי המשרד הם: מנהלת המשרד, המזכירה והגרפיקאיות.

לעובדי המשרד ישנם ימי עבודה ולוח זמנים מתאים, לגרפיקאיות- לעבודת העיצוב, ולמנהלת- לפגישות מול הלקוחות.

לקוח המעוניין בעבודת גרפיקה רשאי לתאם פגישת הכרות והסבר (בתשלום) וזאת אף לפני ביצוע ההזמנה בפועל.

הפגישה מסייעת לתיאום ציפיות מושלם בין הלקוחה והמנהלת, כך הן תגענה לתוצאה הטובה ביותר. הפגישה תיערך במשך חצי שעה. במידה ויש צורך לפגישה ארוכה יותר- ניתן יהיה לקבוע שתי פגישות רצופות.

במידה והלקוח מעוניין בפגישת היכרות והוא אינו לקוח רשום במערכת, הפגישה תרשם על שם מנהל המשרד ורק אם הלקוח יבצע הזמנה לאחר הפגישה- מחיר הפגישה יתווסף להזמנתו והפגישה תירשם במערכת על שמו.

בזמן ביצוע ההזמנה- במידה והלקוח אינו רשום במערכת, המזכירה תמלא את פרטיו: פרטים אישיים ופרטים ליצירת קשר.

לאחר מכן הלקוח יבחר את סוגי הגרפיקה בהן הוא מעוניין וכן תאריך יעד לסיום הגשת ההזמנה.

כעת המערכת תחשב באופן אוטומטי את משך זמן העבודה של הזמנה זו, ותבדוק לאיזו גרפיקאית לתת לעצב את ההזמנה וזאת לפי לוח הזמנים הפנוי של הגרפיקאית.

במידה וקיימות מספר גרפיקאיות שיש להן זמן פנוי לעיצוב הזמנה זו, הלקוח רשאי לבחור את הגרפיקאית הרצויה לו מבין רשימת הגרפיקאיות.

בעת סגירת ההזמנה- הגשתה ללקוח, ההזמנה נדרש הלקוח לאשר אופן תשלום לבחירתו: תשלום באשראי או בהעברה בנקאית. (במידה ובחר בהעברה בנקאית- יש לצרף מספר אסמכתא לביצוע ההעברה).

ניתן לקבוע פגישות ייעוץ והכוונה עם המנהלת גם במהלך העבודה על ההזמנה.

התשלום יתבצע רק לאחר סיום העבודה, כיוון שיתכן ותהיינה פגישות במהלך העבודה וכן שעות עבודה נוספות.

מטרות המערכת:

מטרת על:

לאפשר למשרד הגרפיקה לקבוע הזמנות ופגישות בצורה קלה ויעילה ללא צורך ברישום ידני שגורם להתנהלות להיות קשה ומסובכת יותר.

מטרות נלוות:

- לאפשר רישום לקוחות בצורה מסודרת.
- ניהול יומן פגישות באופן מסודר.
- חלוקת עיצוב ההזמנות לגרפיקאיות על ידי חישוב הזמן הפנוי שלהן בצורה ממוחשבת ללא תקלות.
- הצגת המערכת בצורה נוחה ונעימה למראה.

היקף הפעילות:

הגרפיקאיות עובדות בימים א'-ה' בין השעות: 09:00-15:00 . במהלך העבודה הן תעשינה את העבודה הגרפית.

בזמן זה מנהלת המשרד תקיים פגישות הכרות / ייעוץ והכוונה עם הלקוחות.

מידי שבוע ישנן כ- 12 הזמנות מלקוחות. קבלת ההזמנות תתבצע לפי חישוב אוטומטי אם ישנו זמן פנוי בלוח הזמנים של הגרפיקאיות.

מבנה הארגון:

העסק מנוהל ע"י בעלת המשרד והמזכירה.

המזכירה קובעת את הפגישות עם הלקוחות.

בעלת המשרד מבצעת הזמנות מול לקוחות, עדכון מחירים, ופגישות עם לקוחות.

בניהול הכספי נעזרת בעלת המשרד- במשרד לניהול חשבונות.

הגרפיקאיות מבצעות את עבודות הגרפיקה שנקבעו להן בהתאם ללוח הזמנים ומועד הגשת העבודות.

אופי הפעילות:

הלקוח יוצר קשר טלפוני / במייל עם המזכירה, היא שולחת לו קטלוג ובו סוגי הגרפיקה שהגרפיקאיות מבצעות.

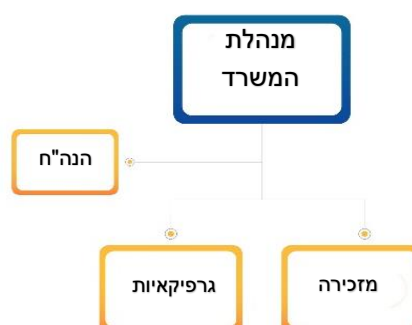
במידה והלקוח מעוניין בפגישת הכרות ותאום ציפיות עם המנהלת- המזכירה קובעת מועד לפגישה.

הלקוח בוחר את סוגי הגרפיקה בהם מעוניין, המזכירה מכניסה למערכת את פרטי ההזמנה, הכוללים גם פרטי לקוח- במידה ואינו קיים במאגר הלקוחות.

לאחר מכן היא שומרת פרטי תשלום, אך התשלום בפועל יתבצע רק בזמן סגירת ההזמנה- לאחר ביצועה.

כמו כן, ישנה אפשרות לפגישות ייעוץ והכוונה עם המנהלת במהלך העבודה.

תרשים מבנה המערכת:



מהות המשימה:

כיום, המערכת עובדת באופן ידני. עובדה זו גורמת למספר קשיים. כגון:

- חוסר סדר בפרטי לקוחות, הזמנות, פגישות ועוד.
- קביעת הזמנות ופגישות באופן ידני גורם לחוסר תאום. דבר זה עלול ליצור מצב שבו: הזמנה מסוימת לא תהיה מוכנה במועד סיומה, נקבעה פגישה עם לקוח- בזמן של פגישה אחרת.
- אימות נתונים- אין אימות נתונים בפרטי לקוח, לדוג' מספר זהות, מספר פלאפון, כתובת מייל. וכך יכול להיווצר מצב של אי יכולת ליצור קשר עם לקוח.
- קושי בשליפת מידע נצרך, חיפוש נתונים. לדוג' הזמנות של לקוח, פגישות שבוצעו עם לקוח ועוד.
- טעויות בניהול היומן במקרה של ביטולים.
- אין מערכת מסודרת שמחשבת לוח זמנים של גרפיקאית, וכן מי פנויה לקבל עבודת גרפיקה שיש לסיימה עד לתאריך הסיום.

תיחום המערכת:

המערכת נועדה לפעילותם של: מנהלת המשרד, גרפיקאיות ומזכירה.

גורמים מעורבים:

גורם:	תחום אחריות:	רמת מעורבות:
מנהל המשרד	קבלת גרפיקאיות חדשות לעבודה, קביעת מחירים, אישור חופשות, ביצוע פגישות עם לקוחות, הוספת סוגי גרפיקה לקטלוג.	משתמש משני
גרפיקאיות	מקבלת עבודות גרפיקה, מבקשת חופשות.	צרכן
מזכירה	מתווכת בין הלקוח לגרפיקאיות, קובעת פגישות, מבצעת הזמנות, גובה מחירים.	משתמש עיקרי

תיחום תהליכים:

המערכת תטפל ב:

- ניהול מאגר הלקוחות.
- ניהול הזמנות של לקוחות.
- ניהול תשלומים.
- ניהול יומן עבודה לגרפיקאיות.

המערכת לא תטפל ב:

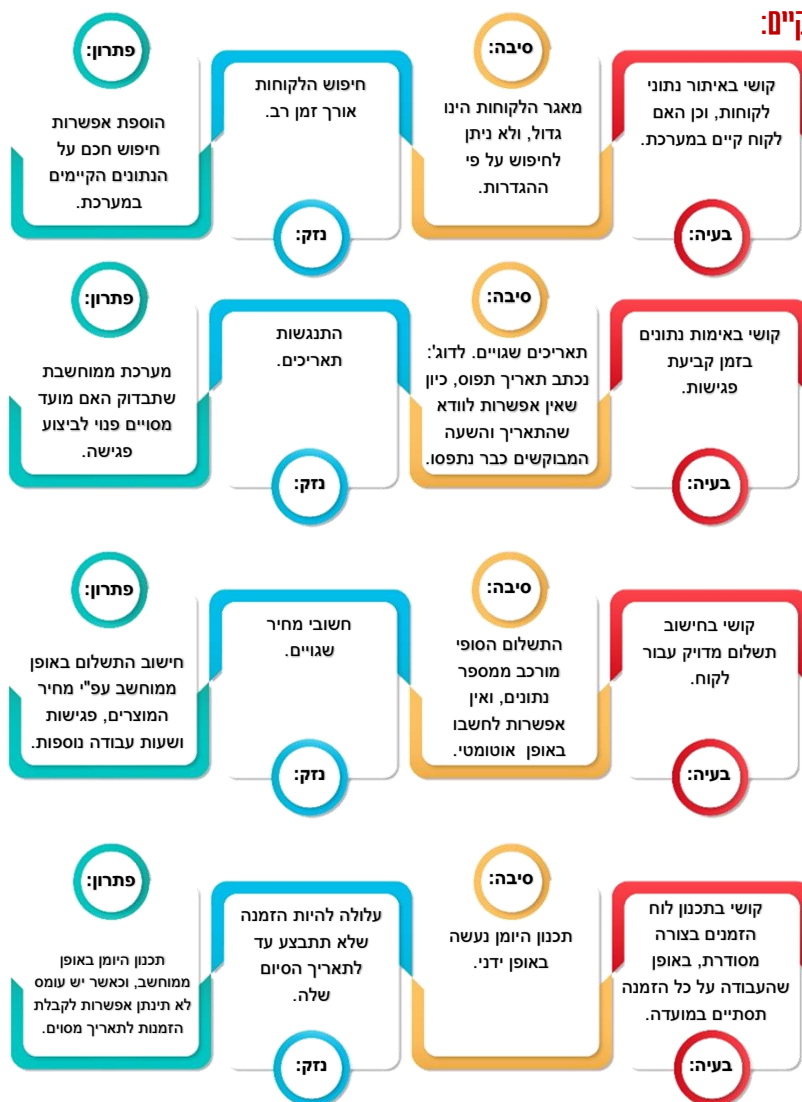
- הנהלת חשבונות.
- חישובי משכורות.

אילוצי המערכת:

- **לוח זמנים** - מועד הגשת הפרויקט עד סוף יוני 2023.
- **סביבת פיתוח** - פיתוח בסביבת visual studio 2019 C#, מסד נתונים Access 2002, בטכנולוגיית Windows Form 2003.
- **כוח אדם** - המערכת תפותח ותתוכנת ע"י תלמידת כיתה י"ב במגמת הנדסת תוכנה.
- **תקציב** - הסכום המוקצה לצידוד ולרכישת תוכנה לא יעלה על 10,000 ₪. וכן הסכום המוקצה לפיתוח התוכנה לא יעלה על 7,000 ₪.

ניתוח מצב קיים:

בעיות במצב הקיים:



יעדי המערכת:

רשימת יעדים:

שם היעד	תיאור
ניהול מאגרים	<ul style="list-style-type: none"> ניהול מאגרי המידע הבאים: לקוחות, עובדים, קטלוג סוגי גרפיקה, קביעת פגישות, תשלומים, ימי חופשה לעובדים, יומן ניהול לגרפיקאיות.
טיפול בפרטי לקוח	<ul style="list-style-type: none"> קליטת נתונים של לקוח והוספתם למאגר הלקוחות. עדכון פרטי לקוח. מציאת פרטי לקוח בחיפוש חכם.
טיפול בפרטי גרפיקאית	<ul style="list-style-type: none"> קליטת נתונים של גרפיקאית והוספתם למאגר העובדים. עדכון פרטי גרפיקאית. מציאת פרטי גרפיקאית בחיפוש חכם.
טיפול בפרטי קביעת פגישה	<ul style="list-style-type: none"> קליטת נתונים של קביעת פגישה והוספתם למאגר הפגישות. עדכון פרטי פגישה. מציאת פרטי פגישה עם לקוח מסוים.
טיפול בפרטי חופשה לגרפיקאית	<ul style="list-style-type: none"> קליטת נתונים של חופשה והוספתם למאגר החופשות. עדכון פרטי ימי החופשה לעובד. הצגת פרטי חופשה מסוימת. חיפוש ימי חופשה לעובד מסוים.
טיפול בפרטי יומן עבודה לגרפיקאית	<ul style="list-style-type: none"> יצירת נתונים של מערכת שעות עבודה והוספתם למאגר המערכות. עדכון פרטי מערכת לפי דרישות הגרפיקאית / המנהל / חופשות של הגרפיקאית. הצגת פרטי מערכת שעות לגרפיקאית.

אילוצי המערכת:

דרישות פונקציונאליות:

ניהול לקוחות:

- הצגת לקוחות
- הוספת לקוח
- עדכון לקוח
- חיפוש לקוח
- מחיקת לקוח

ניהול עובדים:

- הצגת עובדים
- הוספת עובד
- עדכון עובד
- חיפוש עובד
- מחיקת עובד

ניהול קטלוג:

- הצגת קטלוג
- הוספת פריט לקטלוג
- עדכון פריט בקטלוג
- מחיקת פריט בקטלוג

ניהול פגישות:

- הצגת פגישות
- הוספת פגישה
- עדכון פגישה
- חיפוש פגישה

ניהול הזמנות:

- הצגת הזמנות
- הוספת הזמנה
- עדכון הזמנה
- חיפוש הזמנה
- הצגת פרטי הזמנה

- מחיקת הזמנה

ניהול תשלומים:

- הצגת תשלומים

- הוספת תשלום

- עדכון תשלום

- חיפוש תשלום

- מחיקת תשלום

ניהול ביטולים:

- הצגת ביטולים

- הוספת ביטול

- עדכון ביטול

- מחיקת ביטול

- חיפוש ביטול

- אישור ביטול

דרישות לא פונקציונאליות:

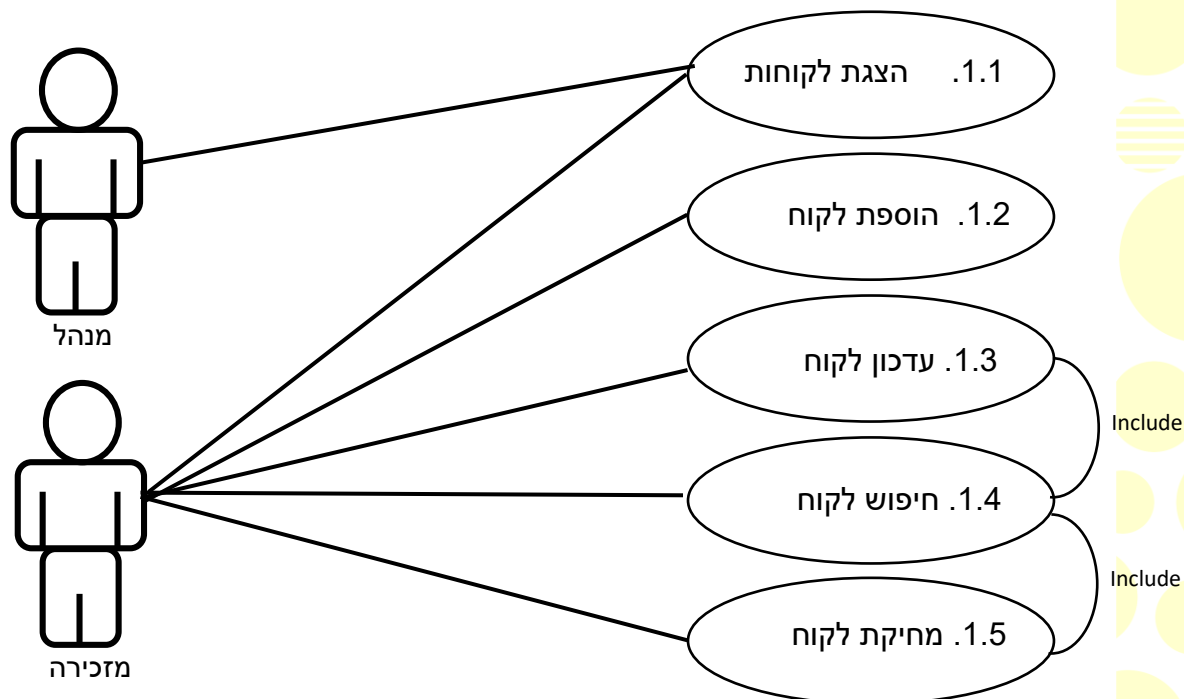
המערכת תכלול הרשאות גישה שונות לפי רמת המשתמש, זאת בכדי למנוע גישה לא מורשית למידע ולביצוע שינויים ללא פיקוח.

דרישות לפי משתמשים:

מנהל	<ul style="list-style-type: none"> - הצגת לקוחות - הוספת עובד - הצגת קטלוג - הוספת פריט לקטלוג - עדכון פריט בקטלוג - הצגת הזמנות 	<ul style="list-style-type: none"> - עדכון פגישה - חיפוש פגישה - הוספת הזמנה - חיפוש ביטול - אישור ביטול
מזכירה	<ul style="list-style-type: none"> - הצגת לקוחות - הוספת לקוח - עדכון לקוח - חיפוש לקוח - מחיקת לקוח - הצגת עובדים - עדכון עובד - חיפוש עובד - מחיקת עובד - הצגת קטלוג - הוספת פריט לקטלוג - עדכון פריט בקטלוג - מחיקת פריט בקטלוג - הצגת פגישות - הוספת פגישה - עדכון פגישה - חיפוש פגישה 	<ul style="list-style-type: none"> - הוספת הזמנה - עדכון הזמנה - חיפוש הזמנה - הצגת הזמנות - הצגת פרטי הזמנה - מחיקת הזמנה - הצגת תשלומים - הוספת תשלום - עדכון תשלום - חיפוש תשלום - מחיקת תשלום - הצגת ביטולים - הוספת ביטול - עדכון ביטול - מחיקת ביטול - חיפוש ביטול
גרפיקאית	<ul style="list-style-type: none"> - הצגת ביטולים - הוספת ביטול - עדכון ביטול - מחיקת ביטול - חיפוש ביטול 	<ul style="list-style-type: none"> - הצגת קטלוג - הצגת הזמנות - הצגת פרטי הזמנה - עדכון הזמנה

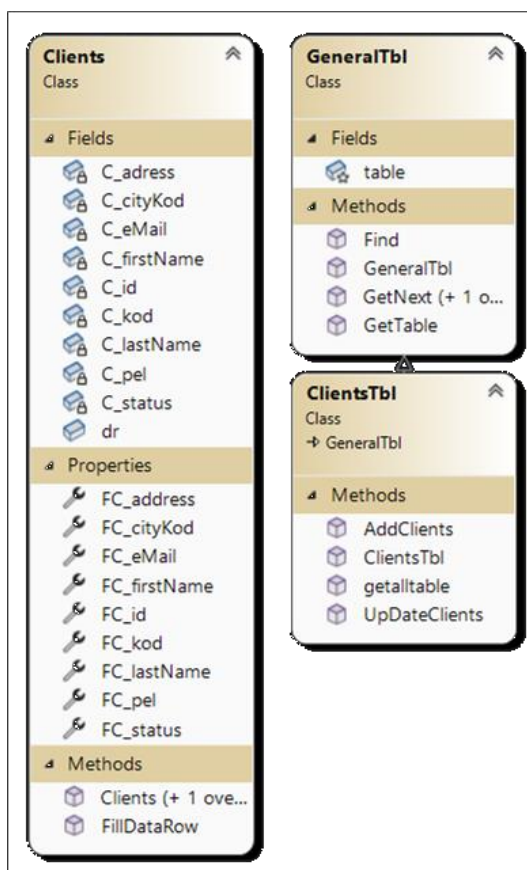
תורשים Use Case1:

ניהול לקוחות:



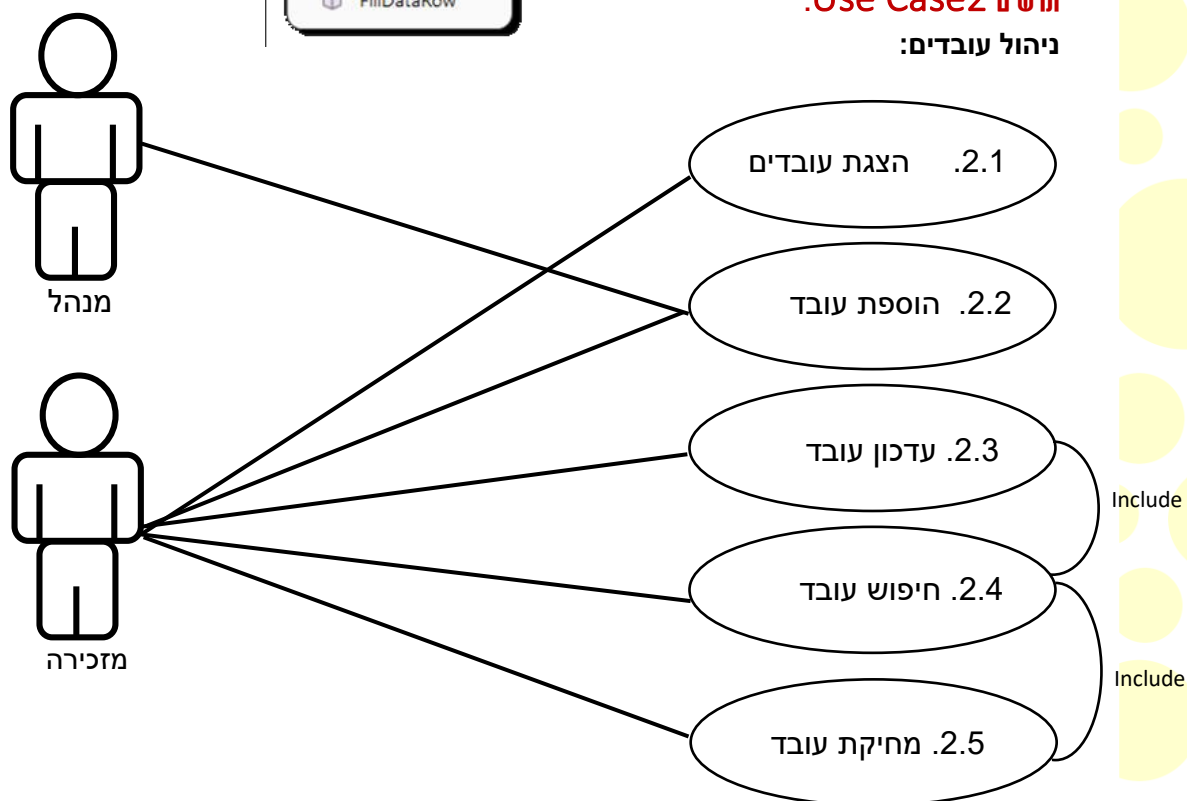
שם ה-UseCase	עדכון לקוח. UseCase 1.3
תיאור קצר	אפשרות לעדכון לקוח במערכת
השחקנים	מזכירה
תנאי קדם	לקוח נמצא במאגר
היזנק	צורך בהוספת לקוח
תיאור מהלך	המערכת מעדכנת לקוח
חריגים	אין
תנאי סופי	עדכון לקוח

שם ה-UseCase	הצגת לקוחות UseCase 1.1
תיאור קצר	אפשרות להצגת לקוחות במערכת
השחקנים	מזכירה, מנהל
תנאי קדם	לקוח נמצא במאגר
היזנק	צורך בהצגת לקוחות
תיאור מהלך	המערכת מציגה את הלקוחות
חריגים	אין
תנאי סופי	הצגת לקוחות



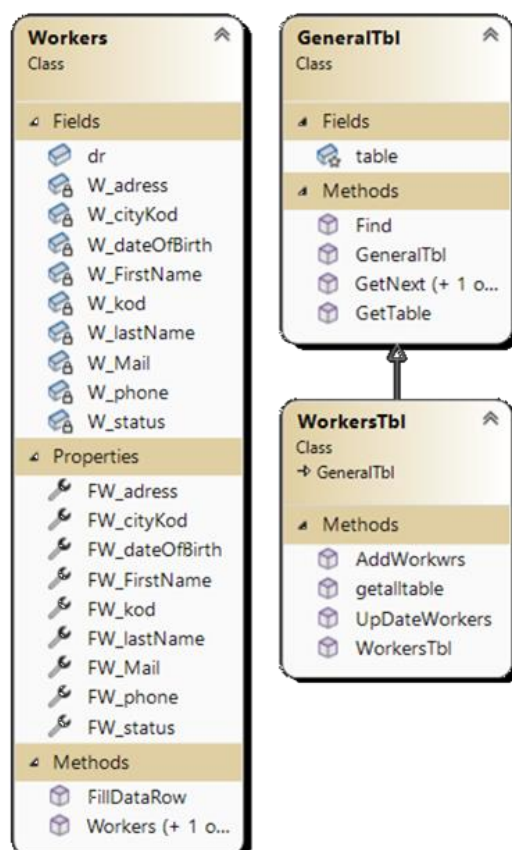
תורשים Use Case2:

ניהול עובדים:



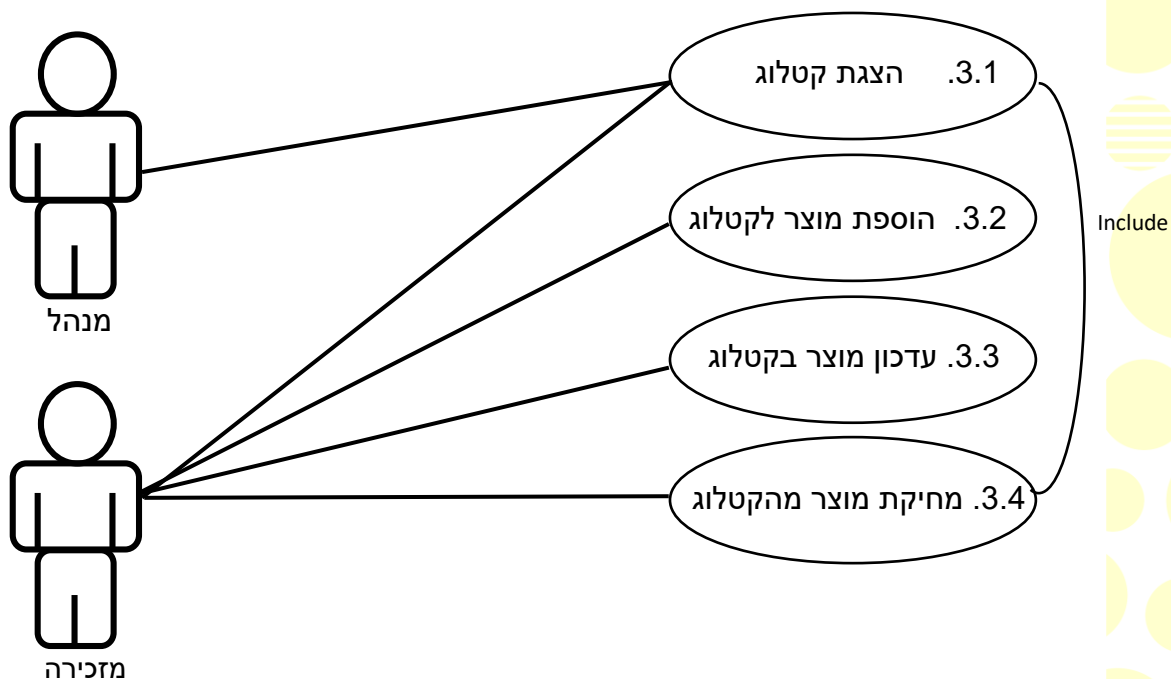
שם ה-UseCase	עדכון עובד UseCase 2.3.
תיאור קצר	אפשרות לעדכון עובד במערכת
השחקנים	מזכירה
תנאי קדם	עובד נמצא במאגר
היזנק	צורך בעדכון עובדים
תיאור מהלך	המערכת מעדכנת עובד
חריגים	אין
תנאי סופי	עדכון עובד

שם ה-UseCase	הוספת עובד UseCase 2.2.
תיאור קצר	אפשרות להוספת עובד למערכת
השחקנים	מזכירה, מנהל
תנאי קדם	עובד נמצא במאגר
היזנק	צורך בהוספת עובד
תיאור מהלך	המערכת מוסיפה עובד
חריגים	אין
תנאי סופי	נוסף עובד



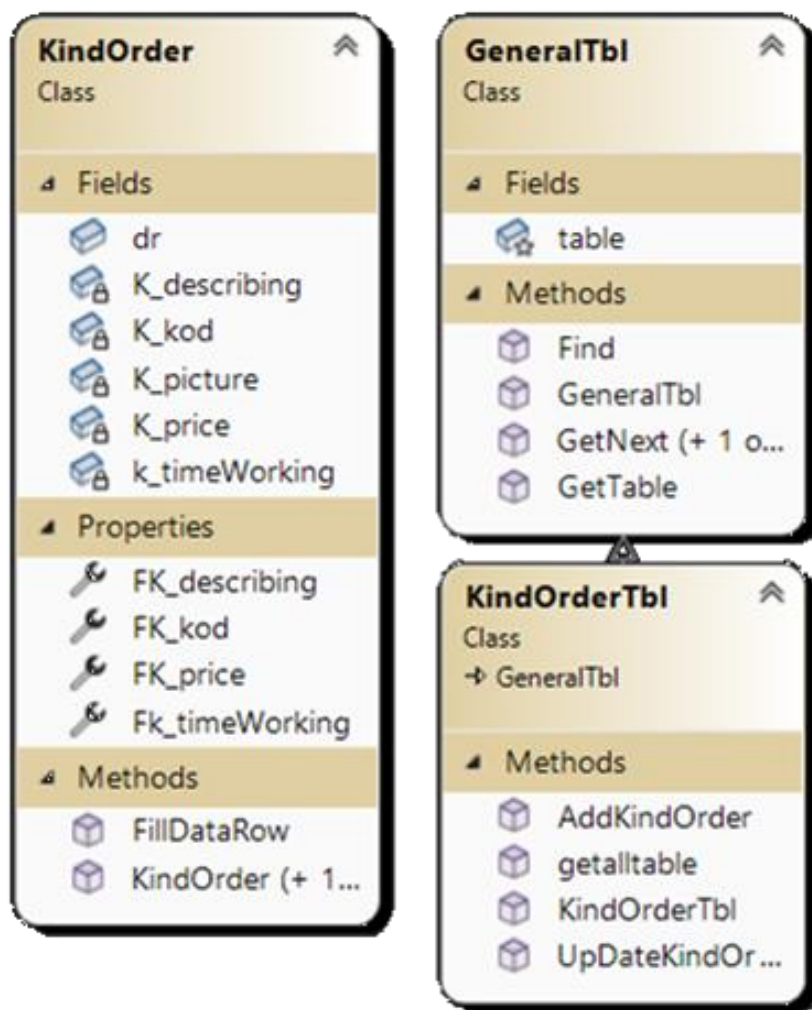
תורשים Use Case 3:

ניהול קטלוג:



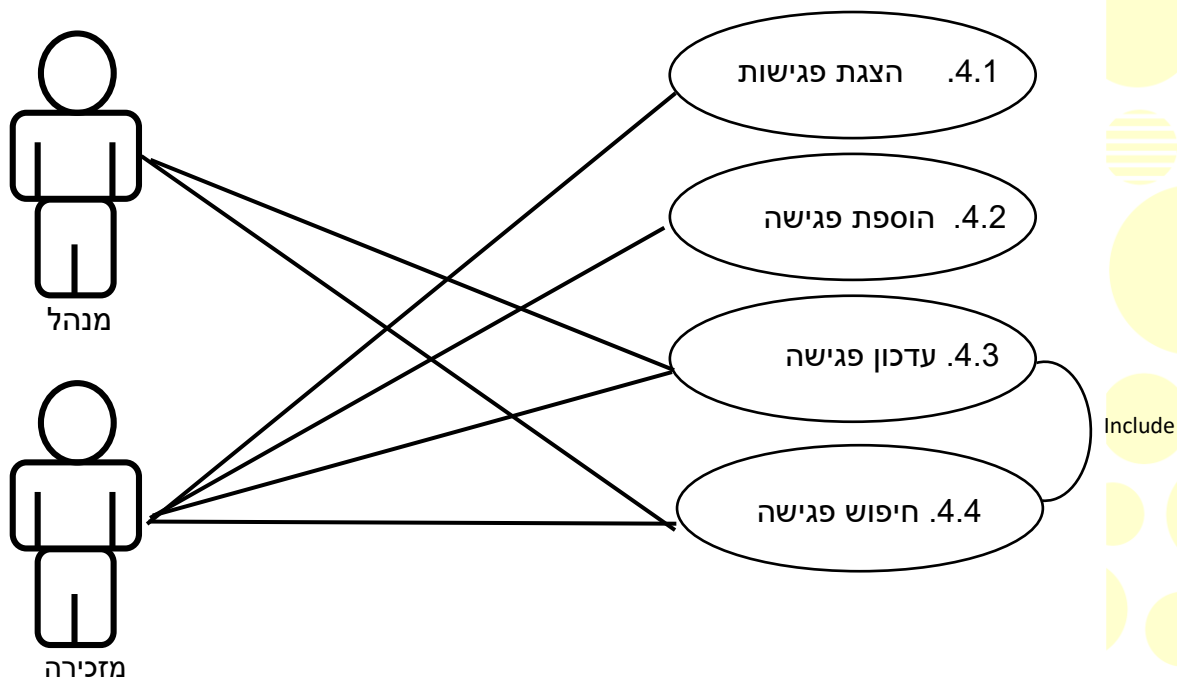
שם ה- UseCase	UseCase 3.2. הוספת מוצר לקטלוג
תיאור קצר	אפשרות להוסיף מוצר חדש למאגר
השחקנים	מזכירה
תנאי קדם	המוצר אינו קיים במאגר
היזנק	מוצר מתווסף למאגר הקטלוג
תיאור המהלך	השחקן מזין את פרטי המוצר החדש המערכת שומרת את פרטי המוצר
חריגים	אין
תנאי סופי	מוצר חדש התווסף למאגר

שם ה- UseCase	עדכון מוצר בקטלוג .3.3 UseCase
תיאור קצר	אפשרות לעדכן את פרטי מוצר בקטלוג
השחקנים	מזכירה
תנאי קדם	המוצר קיים במאגר
היזנק	פרטים מסוימים השתנו
תיאור המהלך	1. השחקן מבצע חיפוש מוצר בקטלוג 2. המערכת מציגה את פרטי המוצר 3. השחקן משנה את פרטי המוצר 4. המערכת שומרת את שינוי הפרטים
חריגים	אין
תנאי סופי	פרטי המוצר בקטלוג השתנו במאגר



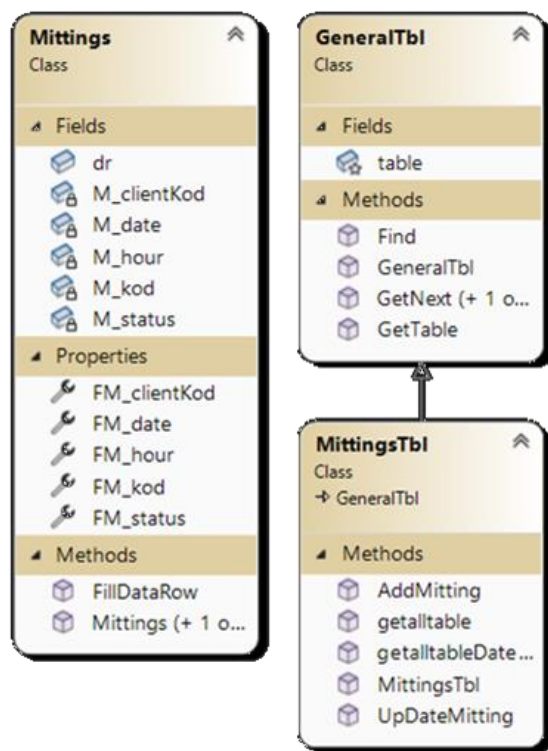
תושים Use Case 4:

ניהול פגישות:

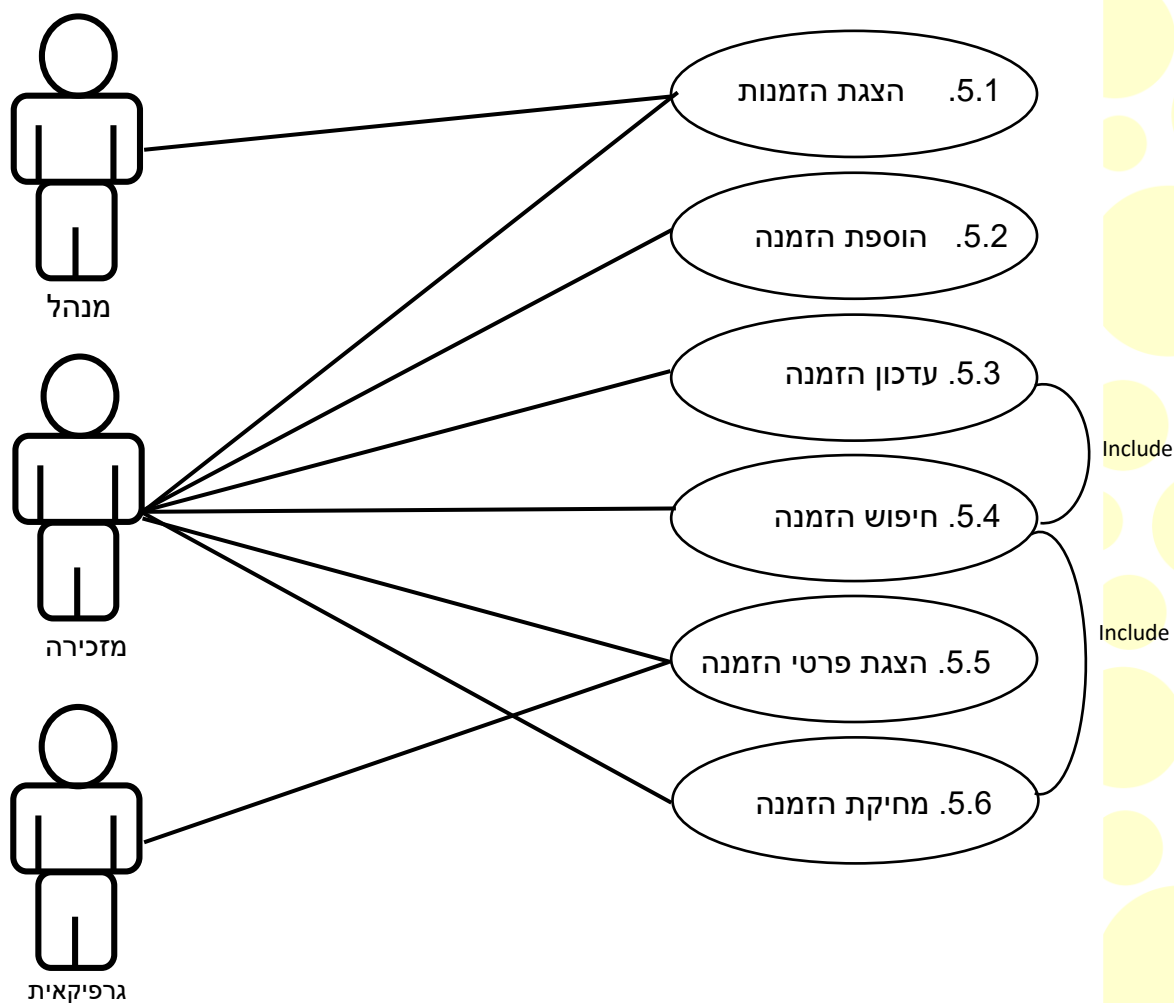


שם ה- UseCase	UseCase 4.2 הוספת פגישה
תיאור קצר	אפשרות להוסיף פגישה חדשה למערכת.
השחקנים	המזכירה
תנאי קדם	יש זימון פנוי
היזנק	המטופל רוצה לקבוע פגישה לטיפול
תיאור המהלך	1. השחקן מזין את פרטי פגישה החדש 2. המערכת שומרת את פרטי הפגישה
חריגים	חריגים: אין
תנאי סופי	פגישה נקבע במערכת

שם ה- UseCase	UseCase 4.3 עדכון פגישה
תיאור קצר	אפשרות לעדכן פרטי פגישה במערכת.
השחקנים	המזכירה, המנהל
תנאי קדם	הזימון קיים במערכת
היזנק	פרטי פגישה השתנו
תיאור המהלך	1. השחקן מזין את הנתונים המעודכנים 2. המערכת מעדכנת את הנתונים הנדרשים
חריגים	אין
תנאי סופי	פרטי פגישה מתעדכנים

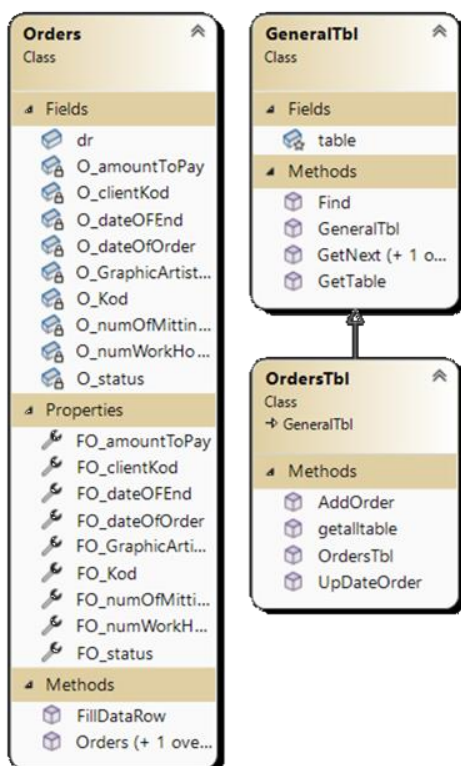


תרחיש Use Case 5: ניהול הזמנות:



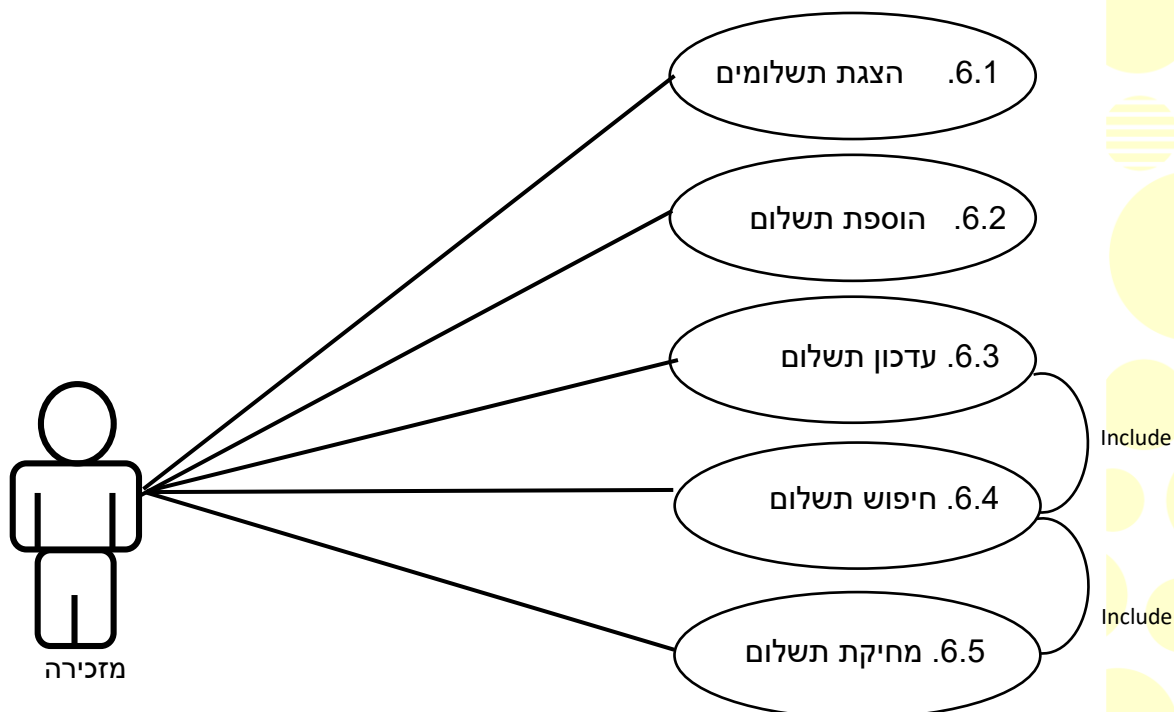
Use Case שם ה-	הוספת הזמנה 5.2 Use Case
תיאור קצר	אפשרות להוספת פרטי הזמנה למערכת.
השחקנים	מזכירה
תנאי קדם	הזמנה לא קימת במאגר.
הנזק	צורך בעדכון הזמנה.
תיאור המהלך	המערכת מציגה את פרטי הזמנה.
חריגים	אין.
תנאי סופי	הוספת פרטי ההזמנה למערכת.

UseCase-ה שם	עדכון הזמנה 5.3 UseCase
תיאור קצר	אפשרות לעדכון הזמנה במערכת
השחקנים	מזכירה
תנאי קדם	הזמנה נמצאת במאגר
היזנק	צורך בעדכון הזמנה
תיאור מהלך	המערכת מעדכנת הזמנה
חריגים	אין
תנאי סופי	עדכון הזמנה



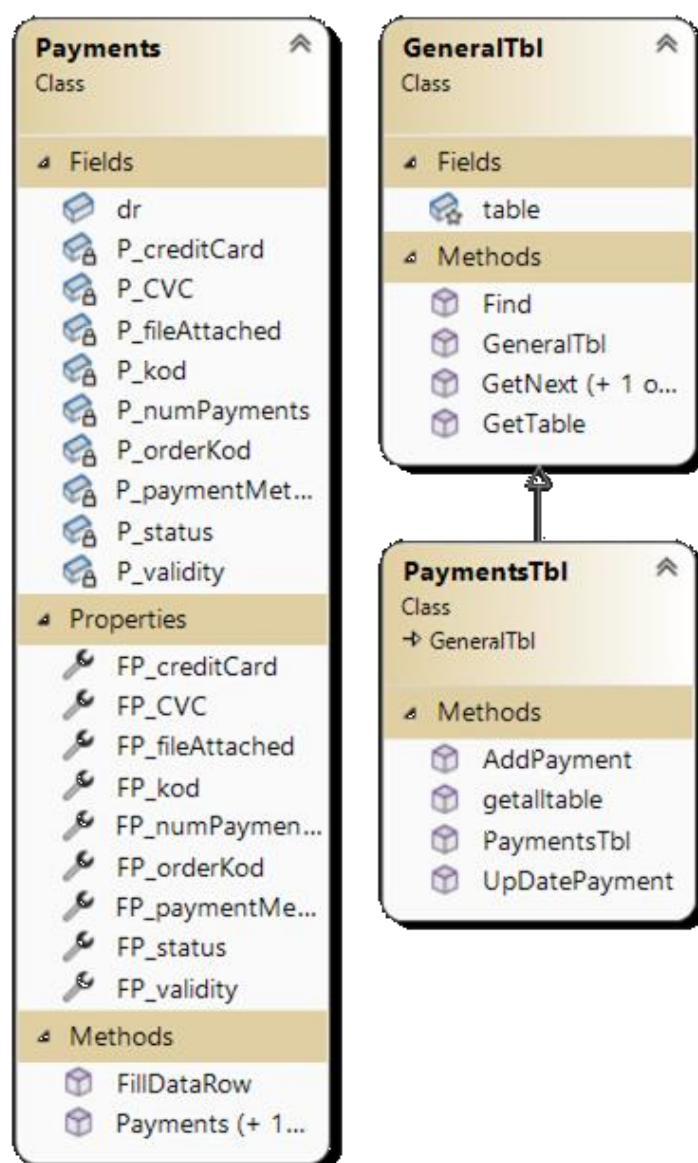
תורשים Use Case 6:

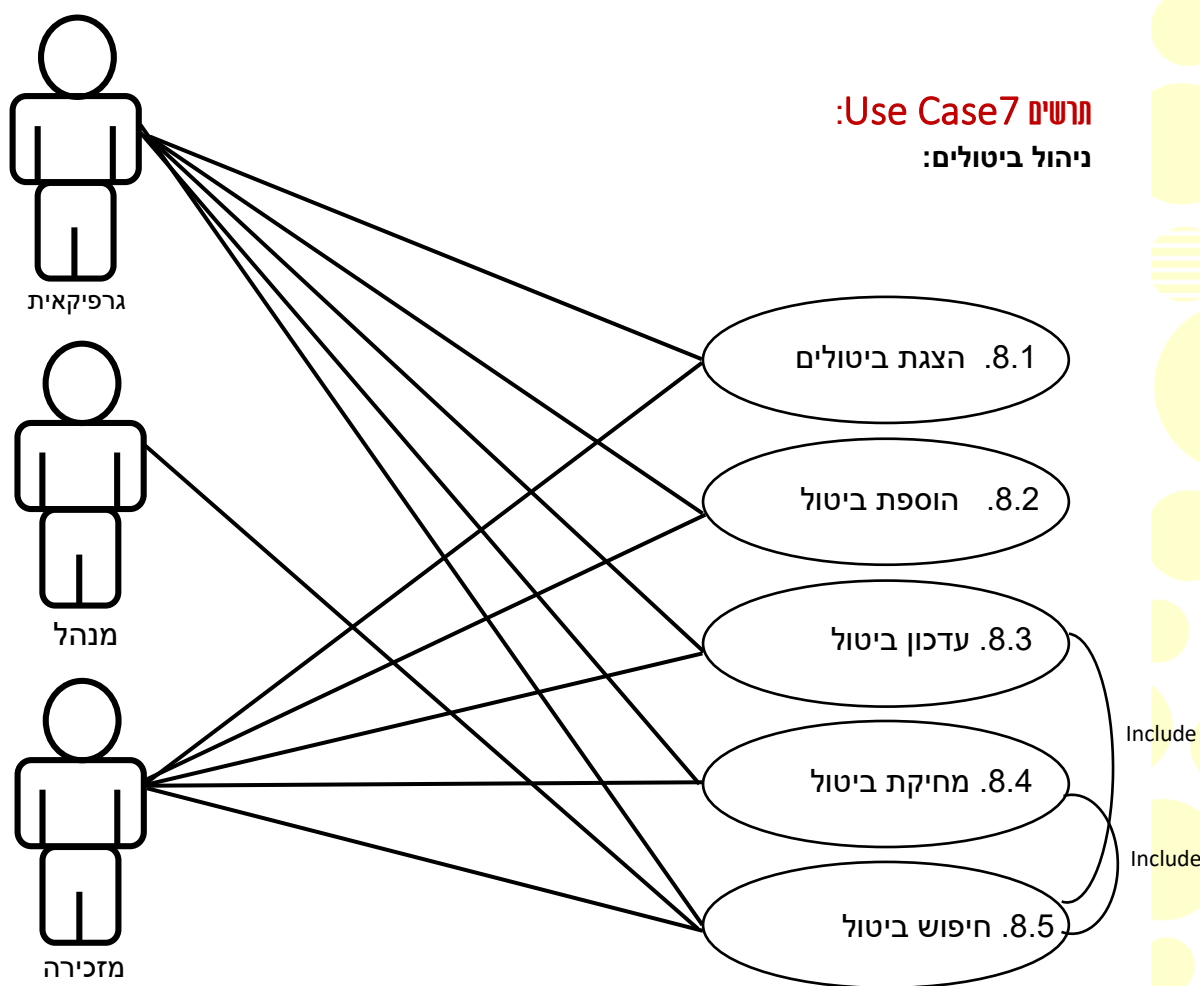
ניהול תשלומים:



שם ה-UseCase	UseCase 6.3 עדכון תשלום
תיאור קצר	אפשרות לעדכון תשלום במערכת
השחקנים	מזכירה
תנאי קדם	לקוח נמצא במאגר
היזנק	צורך בהוספת תשלום
תיאור מהלך	המערכת מעדכנת תשלום
חריגים	אין
תנאי סופי	עדכון תשלום

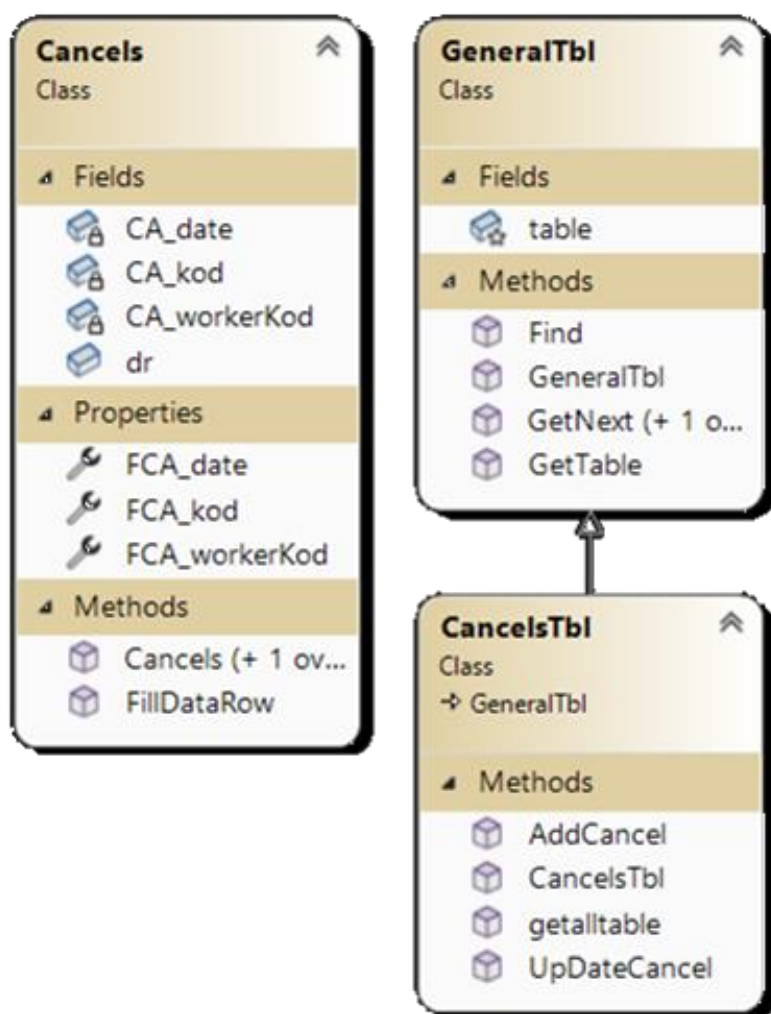
שם ה-UseCase	UseCase 6.1 הצגת תשלומים
תיאור קצר	אפשרות להצגת תשלומים במערכת
השחקנים	מזכירה
תנאי קדם	תשלום נמצא במאגר
היזנק	צורך בהצגת תשלומים
תיאור מהלך	המערכת מציגה את תשלומים
חריגים	אין
תנאי סופי	הצגת תשלומים





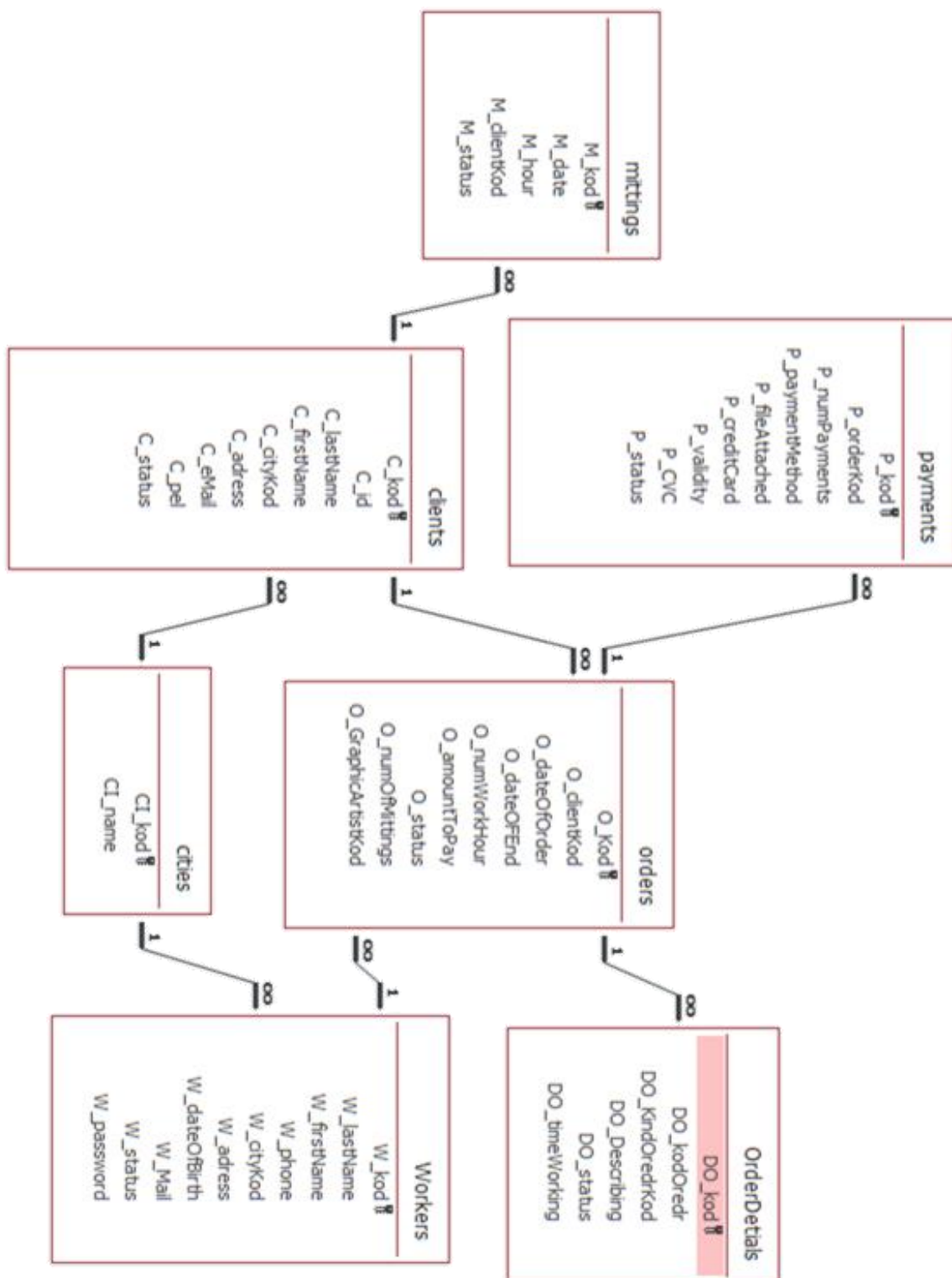
שם ה-UseCase	UseCase 8.2. הוספת ביטול
תיאור קצר	אפשרות להוסיף ביטולים למערכת
השחקנים	מזכירה, גרפיקאית
תנאי קדם	ביטולים נמצא במאגר
היזנק	צורך בהוספת ביטולים
תיאור מהלך	המערכת מוסיפה ביטולים
חריגים	אין
תנאי סופי	נוסף ביטול

שם ה-UseCase	UseCase 8.3. עדכון ביטול
תיאור קצר	אפשרות לעדכן ביטול במערכת
השחקנים	מזכירה, גרפיקאית
תנאי קדם	ביטול נמצא במאגר
היזנק	צורך בהוספת ביטול
תיאור מהלך	המערכת מוסיפה ביטול
חריגים	אין
תנאי סופי	עדכון ביטול



בסיס הנתונים:

תצלום קשרי גומלין:



פירוט הטבלאות:

טבלת ערים:

cities		
שם שדה	סוג נתונים	קוד עיר
CI_kod	מספר	קוד עיר
CI_name	טקסט קצר	שם

טבלת ביטולים:

cancels		
שם שדה	סוג נתונים	קוד ביטול
CA_kod	מספר	קוד ביטול
CA_workerKod	מספר	קוד עובד
CA_date	תאריך/שעה	תאריך
CA_Status	טקסט קצר	סטטוס

טבלת לקוחות:

clients		
שם שדה	סוג נתונים	קוד לקוח
C_kod	מספר	קוד לקוח
C_id	טקסט קצר	מספר זהות
C_lastName	טקסט קצר	שם משפחה
C_firstName	טקסט קצר	שם פרטי
C_cityKod	מספר	קוד עיר
C_adress	טקסט קצר	כתובת מגורים
C_eMail	טקסט קצר	כתובת מייל
C_pel	מספר	מספר פלאפון
C_status	כן/לא	סטטוס

טבלת קטלוג:

kindOrder		
שם שדה	סוג נתונים	קוד מוצר בקטלוג
K_kod	מספר	קוד מוצר בקטלוג
K_describing	טקסט קצר	תאור
K_price	מספר	מחיר
k_timeWorking	מספר	משך זמן עבודה
K_picture	היפר-קישור	תמונה

טבלת פגישות:

mittings		
שם שדה	סוג נתונים	קוד פגישה
M_kod	מספר	קוד פגישה
M_date	תאריך/שעה	תאריך
M_hour	תאריך/שעה	שעה
M_clientKod	מספר	קוד לקוח
M_status	כן/לא	סטטוס

טבלת מוצרים בהזמנה:

OrderDetails		
שם שדה	סוג נתונים	קוד מוצר בהזמנה
DO_kod	מספר	קוד מוצר בהזמנה
DO_kodOredr	מספר	קוד הזמנה
DO_KindOredrKod	מספר	קוד מוצר בקטלוג
DO_Describing	טקסט קצר	תאור
DO_status	כן/לא	סטטוס
DO_timeWorking	מספר	משך זמן עבודה בפועל

טבלת הזמנות:

orders		
שם שדה	סוג נתונים	קוד הזמנה
O_Kod	מספר	קוד הזמנה
O_clientKod	מספר	קוד לקוח
O_dateOfOrder	תאריך/שעה	תאריך ביצוע הזמנה
O_dateOfEnd	תאריך/שעה	תאריך סיום הזמנה
O_numWorkHour	מספר	מספר שעות עבודה נוספות
O_numOfMittings	מספר	מספר פגישות שבועיות
O_amountToPay	מספר	סכום לתשלום
O_status	כן/לא	סטטוס
O_GraphicArtistKod	מספר	קוד עובד

טבלת עובדים:

Workers		
שם שדה	סוג נתונים	קוד עובד
W_kod	מספר	קוד עובד
W_lastName	טקסט קצר	שם משפחה
W_firstName	טקסט קצר	שם פרטי
W_phone	מספר	מספר פלאפון
W_cityKod	מספר	קוד עיר
W_adress	טקסט קצר	כתובת מגורים
W_dateOfBirth	תאריך/שעה	תאריך לידה
W_Mail	טקסט קצר	כתובת מייל
W_status	כן/לא	סטטוס

טבלת תשלומים:

payments		
שם שדה	סוג נתונים	קוד תשלום
P_kod	מספר	קוד תשלום
P_orderKod	מספר	קוד הזמנה
P_numPayments	מספר	מספר תשלומים
P_paymentMethod	טקסט קצר	אמצעי תשלום
P_fileAttached	היפר-קישור	קובץ מצורף
P_creditCard	מספר	מספר כרטיס אשראי
P_validity	תאריך/שעה	תוקף
P_CVC	מספר	3 ספרות בגב הכרטיס
P_status	כן/לא	סטטוס

מבנה וארכיטקטורה:

התוכנה לניהול משרד הגרפיקה המתייחסת ל-2 תהליכים מרכזיים:

1. הזמנת עבודות גרפיקה.

2. ניהול יומן.

התוכנה נכתבה בשפת C# בסביבת Visual Studio 2019.

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

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 הכול 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:

מדריך הנתונים:

שכבה זו מטפלת בקישור התוכנה עם בסיס הנתונים, בה מחלקה לתקשורת עם Dal.cs.DataBasen

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 DataBas .

כל שאר המחלקות פונות ל DataBase דרך מחלקה זו.

במחלקה זו יש מספר פונקציות ששולפות, מעדכנות, מוחקות, ומוסיפות ל DataBase .

המחלקה Dal - בנוסף.

שכבת העסק BLL

שכבה המטפלת בכל הפונקציות והלוגיקה הקשורות לתוכנה עצמה .

המחלקות המוכלות בשכבה:

- מחלקת בסיס לטיפול בטבלה אחת מתוך ה DataBase – GeneralTbl .
- מחלקה עבור כל טבלה מה DataBase היורשת מהמחלקה GeneralTbl .

מחלקות nameTableRow - תפקידן הוא לקשר בין שכבת הנתונים לבין שכבת הממשק החיצוני.

המחלקה GeneralTbl :

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 .

המחלקה GeneralTbl - בנספח.

המחלקות NameTableTable:

כל אחת מן המחלקות הבאות יורשת מהמחלקה GeneralTbl ומנהלת את הטיפול בטבלה בודדת מתוך ה DataBase .

המחלקה CancelsTbl - בנספח.

המחלקה CitiesTbl - בנספח.

המחלקה ClientsTbl - בנספח.

המחלקה KindOrderTbl - בנספח.

המחלקה MitingsTbl - בנספח.

המחלקה OrderDeitalsTbl - בנספח.

המחלקה OrdersTbl - בנספח.

המחלקה PaymentsTbl - בנספח.

המחלקה WorkersTbl - בנספח.

מחלקת: nameTableRow

כל מחלקה מהן, מטופלת ברשימה אחת מתוך טבלה מסוימת .

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

המחלקה Cancels - בנספח.

המחלקה Cities - בנספח.

המחלקה Clients - בנספח.

המחלקה KindOrder - בנספח.

המחלקה Mittings - בנספח.

המחלקה OrderDeitals - בנספח.

המחלקה Orders - בנספח.

המחלקה Payments - בנספח.

המחלקה Workers - בנספח.

שכבת היישום:

שכבה המטפלת בממשק החיצוני של התוכנה.

טופס של כניסת משתמש: - בנספח.

טופס ראשי: - בנספח.

טופס הוספה ועדכון ביטולים+ הצגה: - בנספח.

טופס הוספה ועדכון ערים+ הצגה: - בנספח.

טופס הוספה ועדכון לקוחות+ הצגה: - בנספח.

טופס הוספה ועדכון קטלוג+ הצגה: - בנספח.

טופס הוספה ועדכון פגישות+ הצגה: - בנספח.

טופס הוספה ועדכון פרטי הזמנה+ הצגה: - בנספח.

טופס הוספה ועדכון הזמנה+ הצגה: - בנספח.

טופס הוספה ועדכון תשלומים+ הצגה: - בנספח.

טופס יומן: - בנספח.

שימוש ב- user control

User Katalog - נמצא בהוספת / עדכון הזמנה. מציג את קטלוג המוצרים שניתן להזמין בצורה מסודרת. ישנה אפשרות לראות את תמונת המוצר ע"י לחיצה על הכפתור המתאים.

הקוד- **בנספח**

האם להזמין	קוד	שם	משך זמן עבודה (בדקות)	מחיר	תמונה	האם ברצע
<input type="checkbox"/>	111	לוגר	180	400	לחץ להצגת תמונה	<input type="checkbox"/>
<input type="checkbox"/>	112	כרטיס ביקור צד אחד	60	200	לחץ להצגת תמונה	<input type="checkbox"/>
<input type="checkbox"/>	113	כרטיס ביקור דו צדדי	90	250	לחץ להצגת תמונה	<input type="checkbox"/>
<input type="checkbox"/>	114	פלייר	180	250	לחץ להצגת תמונה	<input type="checkbox"/>
<input type="checkbox"/>	115	פלייר דו צדדי	240	300	לחץ להצגת תמונה	<input type="checkbox"/>
<input type="checkbox"/>	116	רול אפ	180	400	לחץ להצגת תמונה	<input type="checkbox"/>

User Meeting - נמצא ביומן פגישות עם המנהל, כאשר יש פגישה מוצגים פרטיה, כאשר אין פגישה- ניתן לקבוע בתאריך ושעה נוכחיים ע"י לחיצה על כפתור 'לקביעת פגישה'.

הקוד- **בנספח**

שם לקוח וויספיש רות

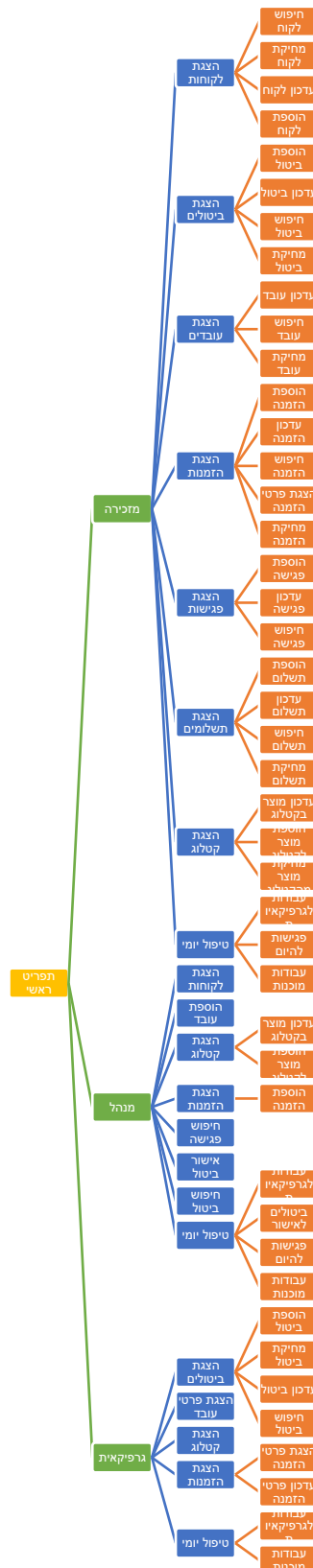
הצגת פרטי פגישה

← כאשר ישנה פגישה בזמן הנוכחי ביומן

לקביעת פגישה

← כאשר לא קיימת פגישה בזמן הנוכחי ביומן

עץ תפריטים:



צילומי מסך הרצה:

ברוך הבא לתוכנת ניהול משרד גרפיקה.

תוכנה זו נועדה לנהל את משרד הגרפיקה בצורה הטובה ביותר עם יומן פגישות מסודר.

מטרת התוכנה לעזור לניהול מסודר של הזמנות, פגישות, לקוחות, תשלומים ועוד, וכן ולהתנהל בצורה מסודרת בכל ענייני המשרד.

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

מאחלים בהצלחה ובהנאה לך וללקוחותיך:

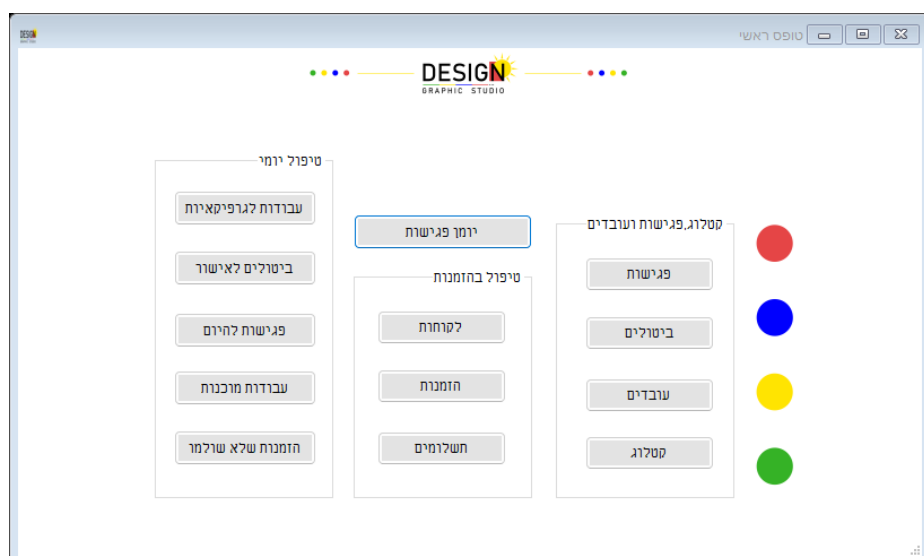
מסך הפתיחה:

מורשי הכניסה לתוכנה הם: מנהלת, מזכירה וגרפיקאיות.

עליך להקיש את הסיסמא המתאימה לתפקידך (בהתאם לסיסמת הכניסה יתבצעו גם הרשאות הגישה לטפסים השונים).

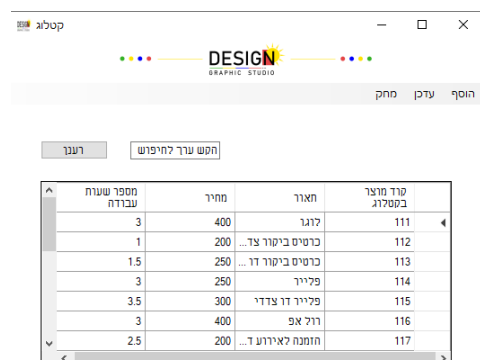


מכאן ניתן לגשת לכל הטפסים העיקריים בפרויקט: פגישות, קטלוג, עובדים, מחירון, ביטולים, תשלומים, לקוחות, הזמנות, ערים, עבודות לגרפיקאיות, יומן פגישות.



טופס הצגת הקטלוג:

כאן ניתן לראות את כל סוגי הגרפיקה שניתן להזמין:



סדר מוצר בסטלוג	תאור	מחיר	מספר שעות עבודה
111	לוגו	400	3
112	כרטיס ביקור צד	200	1
113	כרטיס ביקור דו ...	250	1.5
114	פלייר	250	3
115	פלייר דו צדדי	300	3.5
116	רול אפ	400	3
117	חומנה לאירוע ד...	200	2.5

בטבלה מוצגים כל סוגי הגרפיקה הנמצאים בטבלת קטלוג.

הוספת מוצר לקטלוג:

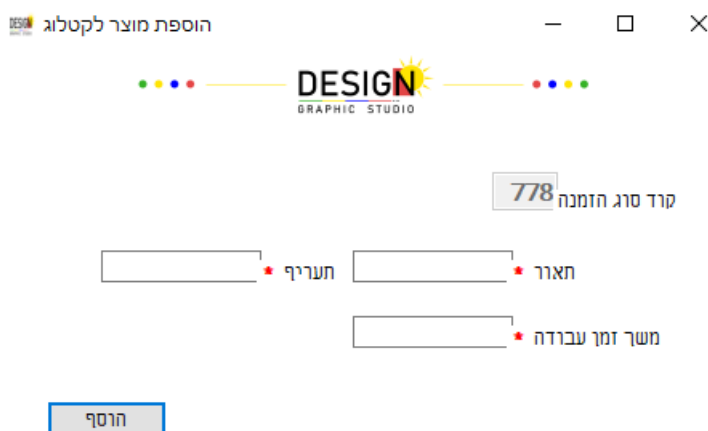
- ללחוץ על הוסף.
- למלא פרטים.
- ללחוץ על הוסף.

עדכון מוצר בקטלוג:

- ללחוץ על שורה של מוצר מהקטלוג
- ללחוץ על עדכון
- לעדכן פרטים
- ללחוץ על עדכן.

מחיקת מוצר מהקטלוג:

- ללחוץ על שורה של מוצר מהקטלוג
- ללחוץ על מחק.



הוספת מוצר לקטלוג

קוד סוג הזמנה 778

תאור *

תעריף *

משך זמן עבודה *

הוסף

טופס הצגת עובדים:

כאן ניתן לראות את כל העובדים:

טור עובד	שם	פלאפון	קוד עיר	כתובת	תאריך לידה	מייל
1	ברנדיז חיה	053-4115968	1	שעורה 4	25/06/2000	ch411@gmail.com
2	בירפוס לאה	054-8461179	4	שמואל הנביא 14	01/07/1995	lee@gmail.com
3	שמעוני ברוך	054-8461187	3	חבצלת חשורו 15	17/02/1997	nt411@gmail.com
4	איתור ברכה	053-4118972	6	חרוב 2	23/08/2002	bb23@gmail.com
5	שטינרס ריח	050-4168773	1	נרטיס 29	05/06/1986	yafa55@gmail.com
6	פרס מרים מירל	050-4277654	5	הרצל 65	18/10/1999	mirifux@gmail.com

בטבלה מוצגים כל העובדים הנמצאים בטבלת עובדים.

הוספת עובד:

- ללחוץ על הוסף
- למלא פרטים
- ללחוץ על הוסף.

עדכון עובד:

- ללחוץ על שורה של עובד
- ללחוץ על עדכון
- לעדכן פרטים
- ללחוץ על עדכון.

מחיקת עובד:

- ללחוץ על שורה של עובד
- ללחוץ על מחק.

הוספת לקוח

קוד עובד 7

פרטים אישיים

שם פרטי ומשפחה *

תאריך לידה

כתובת

עיר רחוב

פרטים ליצירת קשר

פלאפון * מייל *

סיסמת עובד

הוסף

טופס הצגת ביטולים- חופשות:

בטבלה ניתן לראות את כל החופשות שיש לעובדים:

ביטולים

DESIGN GRAPHIC STUDIO

הוסף עדכן אישור מחק

רענון חקש ערך לחיפוש

תאריך	תאריך החתמה	קוד עובד	קוד ביטול
2023	14/05/2023	4	96
2023	19/04/2023	1	97
2023	02/05/2023	6	98
2023	06/06/2023	4	99
2023	17/05/2023	5	100
2023	25/05/2023	5	95
2023	25/04/2023	1	93

בטבלה מוצגות כל חופשות הנמצאות בטבלת חופשות.

הוספת חופשה:

- ללחוץ על הוסף
- למלא פרטים
- ללחוץ על הוסף.

עדכון חופשה:

- ללחוץ על שורה של חופשה
- ללחוץ על עדכון
- לעדכן פרטים
- ללחוץ על עדכון.

מחיקת חופשה:

- ללחוץ על שורה של חופשה
- ללחוץ על מחק.

הוספת ביטול

DESIGN GRAPHIC STUDIO

קוד חופשה 101

פרטי עובד

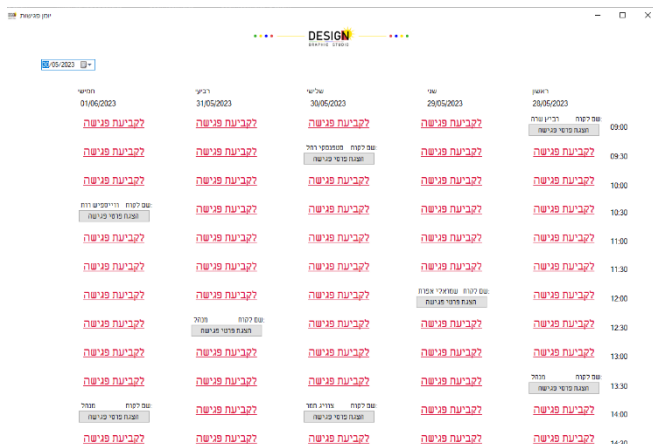
שם משפחה עובד * פרקס מרים מירקן טלפון 050-4277654

פרטי חופשה

תאריך החתמה 24/05/2023 תאריך סיום 24/05/2023

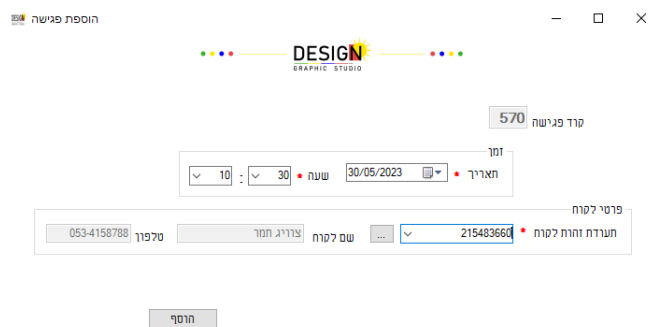
הוסף

כאשר לוחצים על **יומן פגישות** נפתח יומן הפגישות עם מנהלת המשרד.



יום	יום ראשון 29/05/2023	יום שני 30/05/2023	יום שלישי 31/05/2023	יום רביעי 01/06/2023	יום חמישי 02/06/2023
09:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
09:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
10:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
10:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
11:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
11:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
12:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
12:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
13:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
13:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
14:00	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת
14:30	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת	לוקט פגישת

בלחיצה על 'לקביעת פגישה' תפתח פגישה חדשה.



הוספת פגישה

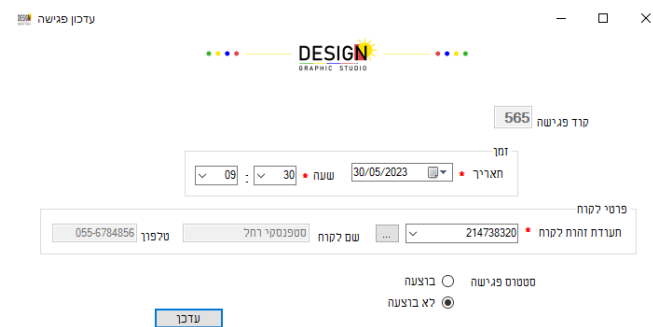
קוד פגישה: 570

זמן: 10 : 30 שעה 30/05/2023 תאריך

פרטי לקוח: חשודת זחור לקוח 215483660 שם לקוח: צוריק חמור טלפון: 053-4158788

הוסף

במידה ובתאריך ובשעה הרצויים ישנה פגישה- יוצג שם הלקוח, ובלחיצה על 'הצגת פרטי פגישה' יפתח טופס עדכון פגישה.



עדכון פגישה

קוד פגישה: 565

זמן: 09 : 30 שעה 30/05/2023 תאריך

פרטי לקוח: חשודת זחור לקוח 214738320 שם לקוח: טסנסקי רחל טלפון: 055-6784856

☐ ברצונך
☒ לא ברצונך

עדכן

ממלאים פרטים ולוחצים על הוסף / עדכן.

טופס הצגת לקוחות:

לקוחות

DESIGN GRAPHIC STUDIO

הוסף עדכן מחק

רענן הקש עזר לחיפוש

קוד לקוח	תעודת זהות	שם	עיר	כתובת	מייל	פלאפון
11	327706123	שמואלי אפרת	רכסים	שעורח 2	...EFRAT1@GMAIL	053-4115914
12	022852107	רביץ שרה	טריית אחא	ר עקיבא	...SS3245@GMAIL	054-8541132
13	228004133	ווייספיש רות	יקנעם	רש"י 28	...R4155@GMAIL	052-8544155
14	214738320	סטפנסקי רחל	טבעון	שמואל הנביא 15	...RR123@GMAIL	055-6784856
15	215483660	צוויג חמר	חיפה	מיכאל 25	...TZ0534158788	053-4158788
1	325629848	מנהל	רכסים	משרד	...OFFICE@GMAIL	077-8566514

בטבלה מוצגים כל הלקוחות הנמצאים בטבלת לקוחות.

הוספת לקוח:

- ללחוץ על הוסף
- למלא פרטים
- ללחוץ על הוסף.

עדכון לקוח:

- ללחוץ על שורה של לקוח
- ללחוץ על עדכון
- לעדכן פרטים
- ללחוץ על עדכן.

מחיקת לקוח:

- ללחוץ על שורה של לקוח
- ללחוץ על מחק.

הוספת לקוח

DESIGN GRAPHIC STUDIO

קוד לקוח 16

פרטים אישיים

מספר זהות * שם פרטי ומשפחה *

כתובת

עיר רכסים רחוב

פרטים ליצירת קשר

פלאפון * 050 מייל *

הוסף

הוספת הזמנה- בלחיצה על 'שמור הזמנה' תופיע השאלה:

שים לב!

האם התקיימה פגישת הכרות לפני ביצוע ההזמנה?

כן
לא

אם לא- ההזמנה תישמר במערכת,
אם כן- יש לעדכן את זמן הפגישה-

זמן פגישת הכרות

תאריך: 24/05/2023
שעה: 00 : 00
עדכן זמן פגישה

שמור הזמנה

תוצג הודעה למשתמש האם זמן הפגישה התעדכן- במידה והנתונים שהוזנו נכונים, או לא.

במערכת לא קיימת פגישה בזמן זה.
בדוק שהנתונים שהזנת נכונים.

אישור

עדכון זמן הפגישה בוצע בהצלחה.
כעת יש לשמור את ההזמנה.

אישור

כעת יש ללחוץ על 'שמור הזמנה'.

טופס הצגת תשלומים:

בטבלה ניתן לראות את כל התשלומים שיש במערכת:

תשלומים

הוסף עדכן מחק

הקש עזר לחיפוש

קוד תשלום	קוד חומנה	חשבונית זיהוי	שם חשבונית	שם פרטי	מספר תשלומים	אמצעי תשלום
6541	81	327706123	שמואלי אפרת		2	כרטיס אשר
6542	83	214738320	מספנסקי רחל		3	כרטיס אשר
6543	82	228004133	ורייספיש רות		1	כרטיס אשר
6544	84	327706123	שמואלי אפרת		1	העברה בנק
6545	85	215483660	צוויג חמר		1	העברה בנק

בטבלה מוצגות כל התשלומים הנמצאות בטבלת תשלומים.

הוספת תשלום:

- ללחוץ על הוסף
- למלא פרטים
- ללחוץ על הוסף.

עדכון תשלום:

- ללחוץ על שורה של תשלום
- ללחוץ על עדכון
- לעדכן פרטים
- ללחוץ על עדכון.

מחיקת תשלום:

- ללחוץ על שורה של תשלום
- ללחוץ על מחק.

הוספת תשלום

DESIGN GRAPHIC STUDIO

קוד חומנה 85

קוד תשלום 6546

סוג תשלום: ☒ תשלום באשראי ☐ העברה בנקאית

פרטי כרטיס אשראי

מספר כרטיס אשראי * חוקף * 2023 / 01 ספרות בגב הכרטיס *

מספר תשלומים 1

הוסף

טופס הצגת פגישות:

בטבלה ניתן לראות את כל הפגישות שיש במערכת:

DESIGN GRAPHIC STUDIO						
הוסף עדכן מחק						
הקש עזר לחיפוש						
רענו						
קוד פגישה	תאריך	שעה	חודרת זמון	שם	סטטוס פגישה	
555	29/05/2023	12:00	327706123	שמואלי אפרת	לא בוצעה	
556	25/05/2023	12:30	325629848	מחל	בוצעה	
557	10/05/2023	11:30	214738320	סטפנסקי רחל	לא בוצעה	
558	12/06/2023	11:00	214738320	סטפנסקי רחל	לא בוצעה	
559	25/05/2023	12:00	022852107	רביץ שרה	בוצעה	
560	17/05/2023	10:00	327706123	שמואלי אפרת	בוצעה	
561	22/05/2023	14:00	215483660	צוריק חמר	לא בוצעה	

בטבלה מוצגות כל הפגישות הנמצאות בטבלת פגישות.

הוספת פגישה:

- ללחוץ על הוסף
- למלא פרטים
- ללחוץ על הוסף.

עדכון פגישה:

- ללחוץ על שורה של פגישה
- ללחוץ על עדכון
- לעדכן פרטים
-
- ללחוץ על עדכון.

מחיקת פגישה:

- ללחוץ על שורה של פגישה
- ללחוץ על מחק.

הוספת פגישה

קוד פגישה 570

זמן

תאריך 24/05/2023 שעה 09:30

פרטי לקוח

טלפון 054-8541132 שם לקוח רביץ שרה חודרת זמון לקוח 022852107

הוסף

כאשר לוחצים על **עבודות לגרפיקאות**- נפתח טופס עבודות לגרפיקאות.
ניתן לבחור עובדת ולראות את העבודות שלה.
כפתור 'הצג עבודות להזמנה':

- ללחוץ על שורה של הזמנה
- ללחוץ על הצג עבודות להזמנה.

קוד הזמנה	שם העובדת	תאריך תחילת העבודה	תאריך סיום העבודה	מספר עבודה	סטטוס
81	שמאלי אפרת	25/02/2023	01/04/2023	0	730
82	רויטספיש רות	20/03/2023	29/03/2023	1	330
85	צוריק חמר	02/03/2023	20/04/2023	4	200

כאשר לוחצים על **ביטולים לאישור**- נפתח טופס ביטולים לאישור.

קוד ביטול	קוד עובד	תאריך תחילת הביטול	תאריך סיום הביטול	סטטוס
96	4	14/05/2023	18/05/2023	בקשה
97	1	19/04/2023	19/04/2023	בקשה
93	1	25/04/2023	25/03/2023	בקשה

כאשר לוחצים על **פגישות להיום**- נפתח טופס ובו רשימת הפגישות להיום.
לתשלום על הפגישה:

- ללחוץ על שורה של פגישה.
- ללחוץ על לתשלום על הפגישה.

כפתור זה פותח טופס תשלום חדש על פגישה זו.
לפתיחת הזמנה:

- ללחוץ על לפתיחת הזמנה.
- כפתור זה יפתח טופס הוספת הזמנה.

MeetingsForToday

DESIGN GRAPHIC STUDIO

לתשלום על הפגישה לפתיחת הזמנה

שם	תעודת זהות לוקח	שעה	תאריך	קוד פגישה
מנחל	325629848	12:30	28/05/2023	556
רביץ שור	022852107	12:00	28/05/2023	559
מנחל	325629848	10:30	28/05/2023	562
צוריק חמו	215483660	11:00	28/05/2023	563

כאשר לוחצים על **עבודות מוכנות** - נפתח טופס ובו רשימת העבודות שבוצעו.
בלחיצה על העבר ללקוח- יפתח טופס הוספת תשלום על ההזמנה והסטטוס שלה יהיה 'סגור'.
העבר ללקוח:

- ללחוץ על שורה של הזמנה.
- ללחוץ על העבר ללקוח.

ReadyOrders

DESIGN GRAPHIC STUDIO

העבר ללקוח

קוד הזמנה	תעודת זהות לוקח	שם	תאריך ביצוע ההזמנה	תאריך הגשת ההזמנה	מספר שעות עבודה נוספות	מכוס לחשור
82	228004133	ווייספיש רוח	20/03/2023	29/03/2023	1	330
84	327706123	שמואלי אפרת	15/01/2023	20/02/2023	0	1000
85	215483660	צוריק חמו	02/03/2023	20/04/2023	4	200

כאשר לוחצים על **הזמנות שלא שולמו**- נפתח טופס ובו רשימת ההזמנות שלא שולמו.
כשאר לוחצים על 'לתשלום' יפתח טופס הוספת תשלום על ההזמנה שנבחרה.
לתשלום:

- ללחוץ על שורה של הזמנה.
- ללחוץ על 'לתשלום'.

OrdersNotPaid

DESIGN GRAPHIC STUDIO

לתשלום

קוד הזמנה	תעודת זהות לוקח	שם	תאריך ביצוע ההזמנה	תאריך הגשת ההזמנה	מספר שעות עבודה נוספות	מכוס לחשור
82	228004133	ווייספיש רוח	20/03/2023	29/03/2023	1	330
84	327706123	שמואלי אפרת	15/01/2023	20/02/2023	0	1000
85	215483660	צוריק חמו	02/03/2023	20/04/2023	4	200

רפלקציה:

הפרויקט אתגר אותי לחשוב בצורה יעילה ויצירתית, לנתח לעומק את הפרטים ולהיות עקבית בעבודתי.

תהליך הפיתוח דרש ממני הרבה חשיבה ומאמץ, הייתי צריכה לפתח קודים, לבדוק את מקור הבעיות בתוכנה.

לעיתים העבודה זרמה בצורה חלקה ובקלות, אך לפעמים העבודה הייתה יותר מאתגרת וקצת מתסכלת ומייאשת- היו נפילות של התוכנה, טעויות בכתיבת הקודים.

אך עם זאת, גיליתי שדווקא הקושי נותן לי המון. שכן הוא לימד אותי שזה בסדר לעשות טעויות וללמוד מהן, וכשאצליח בעזרת ה' להתגבר עליהם- הסיפוק יהיה גדול יותר.

אבל לא היו רק קשיים, היו גם הצלחות רבות שהוסיפו למוטיבציה ולכוח הרצון להמשיך להתמיד בעבודה שנתרה.

עם הזמן העבודה הפכה לי ליותר קלה ונחמדה מאחר והצלחתי לפתור בעיות לבד- כלקח מטעויות שכבר עשיתי לפני כן.

בנוסף, הלמידה בפרויקט המלווה שעשינו יחד עם המורה לפני הפרויקט האישי עזרה לי להבנת התהליך ולניסיון בעבודה בפרויקט.

העבודה בשיתוף פעולה עם חברותי תרמה רבות להעלאת רעיונות ודרכי פתרון לבעיות שונות שנוצרו במהלך העבודה.

אחד הדברים שנהניתי ממנו בפרויקט היה ההזדמנות ליישם את הידע התיאורטי שיש לי לעבודה מעשית, להתנסות, וכך להגיע לתוצאה הרצויה.

במבט לאחור אני חושבת שיכולתי להפיק תועלת לו הייתי כותבת על כל הקודים מה הם עושים. דבר זה היה חוסך לי זמן כאשר לפעמים הייתי צריכה להבין מהי מטרת כתיבת הקוד המסוים...

אני מודה על ההזדמנות ליצירת פרויקט שכזה- שהביא את היכולות שלי למימוש, ונתן לי להתמודד עם אתגרים שונים שהפרויקט זימן לי לאורך כתיבתו.

אני מאמינה שהכישורים והידע שצברתי מפרויקט זה ישמשו אותי בעתיד, שכן אני רוצה בעזרת ה' להמשיך ללמוד במגמת תכנות בסמינר.

ולסיום, אני רוצה להביע את תודתי למורה היקרה שתמכה ועודדה לאורך כל הדרך. היא עשתה הכול בסבלנות, באדיבות ובשמחה- הכול כדי שיצא הכי טוב שיכול להיות. תודה!

תודה רבה:

אפרת.

ביבליוגרפיה:

תכנות מערכות מנהליות בשפת C# .

הכנה לפרויקט הגמר 5 יח"ל.

ב. לנג.

נספחים



שכבת הנתונים:

המחלקה Dal:

```
public class Dal
{
    // מופע המכיל את הנתונים למסד נתונים
    private static OleDbConnection con;
    // מופע המכיל עותק של הטבלאות באקסס
    public static DataSet ds;

    public static Hashtable adapters = new Hashtable();
    // פעולה בונה
    static Dal()
    {
        // להתאים את הנתונים לכל מחשב
        string path = @"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" +
            Application.StartupPath + @"\DataBase\efratSmueliProject.mdb;Persist
            Security Info=True";
        // כאחראי על הנתונים הגדרת con
        con = new OleDbConnection(path);
        // הקצאת מקום לדטא-סט
        ds = new DataSet();
    }
    public static void AddTable(string tableName)
    {
        if (!ds.Tables.Contains(tableName))
        {
            // נוסע להביא את הטבלה
            OleDbDataAdapter adapter = new OleDbDataAdapter("select * from " +
                tableName, con);
            // יוצר משפטי הוספה, עדכון ומחיקה
            OleDbCommandBuilder builder = new OleDbCommandBuilder(adapter);

            // הוספה
            adapter.InsertCommand = builder.GetInsertCommand();
            // עדכון
            adapter.UpdateCommand = builder.GetUpdateCommand();
            // מחיקה
            adapter.DeleteCommand = builder.GetDeleteCommand();
            adapters.Add(tableName, adapter);

            // מדביק את הטבלה ומשפטי הסלקט על הדטא-סט
            adapter.Fill(ds, tableName);
        }
    }
    public static DataTable GetTable(string tableName)
    {
        return ds.Tables[tableName];
    }
    public static DataTable GetDisplayTable(string sqlSelect)
    {
    }
```

```

DataTable dt = new DataTable();
OleDbDataAdapter adapter = new OleDbDataAdapter(sqlSelect, con);
adapter.Fill(dt);
return dt;
}
public static void Save(string tableName)
{
OleDbDataAdapter adapter = (OleDbDataAdapter)adapters[tableName];
adapter.Update(ds, tableName);
}

}

```

שכבת העסק BLL

```

class GeneralTbl
{
protected DataTable table;
public GeneralTbl(string tableName)
{
Classes.Dal.AddTable(tableName);
this.table = Classes.Dal.GetTable(tableName);
}
public DataTable GetTable()
{
return this.table;
}

public int GetNext(string cellName, string tableName)
{
int mis = 0;
DataTable dt = Dal.GetDisplayTable("select max(" + cellName + ") from "
+ tableName);
if (dt.Rows.Count > 0)
{
mis = int.Parse(dt.Rows[0][0].ToString());
mis = mis + 1;
}
else
mis = 1;
return mis;
}

public DataRow Find(string cellName, object value)
{
foreach (DataRow dr in table.Rows)
{
if (dr[cellName].ToString() == value.ToString())
return dr;
}
return null;
}
//פעולה להחזרת ערך המפתח הראשי
public int GetNext(string cellName)
{
DataTable dt = Dal.GetDisplayTable("select max(" + cellName + ") from "
+ table.TableName);

```



```
if (dt.Rows[0][0] is DBNull)
return 1;
int mis = Convert.ToInt32(dt.Rows[0][0]);
return ++mis;
}
}
```

המחלקות: NameTableTable

המחלקה CanceledTbl

```
internal class CanceledTbl : GeneralTbl
{
public CanceledTbl() : base("Canceled")
{ }
public DataTable getalltable()
{
return Dal.GetDisplayTable("SELECT canceled.CA_kod AS [קוד ביטול],
canceled.CA_workerKod AS [קוד עובד], canceled.CA_date AS [תאריך] FROM
canceled;");
}

public void UpDateCancel(Canceled cc)
{
cc.FillDataRow();
Dal.Save(table.TableName);
}

public void AddCancel(Canceled cc)
{
// יצירת שורה בתבנית של הטבלה
cc.dr = table.NewRow();
// מכניסה את התכונות של העצם לשורה
cc.FillDataRow();
// מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
table.Rows.Add(cc.dr);
// עדכון אקסס
Dal.Save(table.TableName);
}
}
```

המחלקה CitiesTbl

```
internal class CitiesTbl : GeneralTbl
{
public CitiesTbl() : base("Cities")
{ }
public DataTable getalltable()
{
return Dal.GetDisplayTable("SELECT cities.CI_kod AS [קוד עיר],
cities.CI_name AS [שם עיר] FROM cities");
}

public void UpDateCity(Cities c)
{
c.FillDataRow();
}
```

```
Dal.Save(table.TableName);
}

public void AddCity(Cities c)
{
    //יצירת שורה בתבנית של הטבלה
    c.dr = table.NewRow();
    //מכניסה את התכונות של העצם לשורה
    c.FillDataRow();
    //מדביק את השורה שמכילה את התכונות של העצם החדש לטבלת קטגוריות
    table.Rows.Add(c.dr);
    //עדכון אקסס
    Dal.Save(table.TableName);
}
}
```

המחלקה ClientsTbl

```
internal class ClientsTbl: GeneralTbl
{
    public ClientsTbl() : base("Clients")
    { }
    public DataTable getalltable()
    {

        return Dal.GetDisplayTable("SELECT clients.C_kod AS [קוד לקוח],
        clients.C_id AS [תעודת זהות], clients.C_lastName AS [שם משפחה],
        clients.C_firstName AS [שם פרטי], cities.CI_name AS [עיר],
        clients.C_adress AS [כתובת], clients.C_eMail AS [מייל], clients.C_peł AS
        [פלאפון], clients.C_status AS [סטטוס] FROM cities INNER JOIN clients ON
        cities.CI_kod = clients.C_cityKod;");

    }
    public void UpDateClients(Clients cl)
    {
        cl.FillDataRow();
        Dal.Save(table.TableName);
    }

    public void AddClients(Clients cl)
    {
        //יצירת שורה בתבנית של הטבלה
        cl.dr = table.NewRow();
        //מכניסה את התכונות של העצם לשורה
        cl.FillDataRow();
        //מדביק את השורה שמכילה את התכונות של העצם החדש לטבלת קטגוריות
        table.Rows.Add(cl.dr);
        //עדכון אקסס
        Dal.Save(table.TableName);
    }
}
```

המחלקה KindOrderTbl

```
internal class KindOrderTbl : GeneralTbl
{
    public KindOrderTbl() : base("KindOrder")
    {
```

```
{ }
public DataTable getalltable()
{
return Dal.GetDisplayTable("SELECT kindOrder.K_kod AS [קוד סוג הזמנה],
kindOrder.K_describing AS [תאור סוג הזמנה], kindOrder.K_price AS [מחיר סוג
הזמנה], kindOrder.k_timeWorking AS [מספר שעות עבודה]\r\nFROM
kindOrder;\r\n");
}
public void UpDateKindOrder(KindOrder k)
{
k.FillDataRow();
Dal.Save(table.TableName);
}

public void AddKindOrder(KindOrder k)
{
//יצירת שורה בתבנית של הטבלה
k.dr = table.NewRow();
//מכניסה את התכונות של העצם לשורה
k.FillDataRow();
//מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
table.Rows.Add(k.dr);
//עדכון אקסס
Dal.Save(table.TableName);
}
}
```

MittingsTbl המחלקה

```
internal class MittingsTbl:GeneralTbl
{
public MittingsTbl() : base("Mittings")
{ }
public DataTable getalltable()
{
//האם לעשות בשאלתה רק מתי שסטטוס לא בוצע?
//where mittings.M_status='לא בוצע'
return Dal.GetDisplayTable("SELECT mittings.M_kod AS [קוד פגישה],
mittings.M_date AS תאריך, mittings.M_hour AS שעה, clients.C_id AS [תעודת
לקוח], clients.C_lastName AS [שם משפחה], clients.C_firstName AS [שם
פרטי], mittings.M_status AS [סטטוס פגישה] FROM clients INNER JOIN mittings
ON clients.C_kod = mittings.M_clientKod ;");
}
//שאלתה שמביאה את הפגישה לפי זמן ותאריך
public DataTable getalltableDateHour(string hour, string date)
{
return Dal.GetDisplayTable("SELECT mittings.M_kod AS [קוד פגישה],
mittings.M_date AS תאריך, mittings.M_hour AS שעה, clients.C_id AS [תעודת
לקוח], clients.C_lastName AS [שם משפחה], clients.C_firstName AS [שם
פרטי], mittings.M_status AS [סטטוס פגישה] FROM clients INNER JOIN mittings
ON clients.C_kod = mittings.M_clientKod WHERE mittings.M_hour = '" +
hour + "' and mittings.M_date = '" + date + "' ");
}
public void UpDateMitting(Mittings m)
{
m.FillDataRow();
Dal.Save(table.TableName);
}
```

```

}

public void AddMitting(Mittings m)
{
    //יצירת שורה בתבנית של הטבלה
    m.dr = table.NewRow();
    //מכניסה את התכונות של העצם לשורה
    m.FillDataRow();
    //מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
    table.Rows.Add(m.dr);
    //עדכון אקסס
    Dal.Save(table.TableName);
}
}

```

OrderDeitalsTbl המחלקה

```

internal class OrderDeitalsTbl :GeneralTbl
{
    public OrderDeitalsTbl() : base("OrderDetials")
    { }
    public DataTable getalltable()
    {
        return Dal.GetDisplayTable("SELECT OrderDetials.DO_kod AS[קוד],
        OrderDetials.DO_kodOredr AS[קוד הזמנה], OrderDetials.DO_KindOredrKod
        AS[קוד סוג הזמנה], OrderDetials.DO_Describing AS[תאור],
        OrderDetials.DO_status AS[סטטוס], OrderDetials.DO_timeWorking AS[משך זמן
        עבודה] FROM OrderDetials\r\n");
    }
    public void UpDateOrderDeitals(OrderDeitals od)
    {
        od.FillDataRow();
        Dal.Save(table.TableName);
    }
    public DataTable GetProductOfOrder(int code)
    {
        return Dal.GetDisplayTable("SELECT OrderDeitals.*\r\nFROM OrderDetails
        where DO_kod=" + code);
    }
    public void AddOrderDeitals(OrderDeitals cl)
    {
        //יצירת שורה בתבנית של הטבלה
        cl.dr = table.NewRow();
        //מכניסה את התכונות של העצם לשורה
        cl.FillDataRow();
        //מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
        table.Rows.Add(cl.dr);
        //עדכון אקסס
        Dal.Save(table.TableName);
    }
}

```

OrdersTbl המחלקה

```

internal class OrdersTbl:GeneralTbl
{

```

```
public OrdersTbl() : base("orders")
{ }
public DataTable getalltable()
{
return Dal.GetDisplayTable("SELECT orders.O_Kod AS [קוד הזמנה],
clients.C_id AS [תעודת זהות לקוח], clients.C_lastName AS [שם משפחה],
clients.C_firstName AS [שם פרטי], orders.O_dateOfOrder AS [תאריך ביצוע הזמנה],
orders.O_dateOfEnd AS [תאריך הגשת ההזמנה], orders.O_numWorkHour AS [מספר שעות
עבודה נוספות], orders.O_amountToPay AS [סכום לתשלום], orders.O_status AS [סטטוס
הזמנה], orders.O_numOfMittings AS [מספר פגישות], orders.O_GraphicArtistKod
AS[קוד גרפיקאי]\r\nFROM (cities INNER JOIN clients ON cities.CI_kod =
clients.C_cityKod) INNER JOIN orders ON clients.C_kod =
orders.O_clientKod;");
}
public void UpDateOrder(Orders o)
{
o.FillDataRow();
Dal.Save(table.TableName);
}

public void AddOrder(Orders o)
{
//יצירת שורה בתבנית של הטבלה
o.dr = table.NewRow();
//מכניסה את התכונות של העצם לשורה
o.FillDataRow();
//מדביק את השורה שמכילה את התכונות שמכילה העצם לטבלת קטגוריות
table.Rows.Add(o.dr);
//עדכון אקסס
Dal.Save(table.TableName);
}
}
```

המחלקה PaymentsTbl

```
internal class PaymentsTbl:GeneralTbl
{
public PaymentsTbl() : base("Payments")
{ }
public DataTable getalltable()
{
return Dal.GetDisplayTable("SELECT payments.P_kod AS [קוד תשלום],
payments.P_orderKod AS [קוד הזמנה], clients.C_id AS [תעודת זהות לקוח מזמין],
clients.C_lastName AS [שם משפחה], clients.C_firstName AS [שם פרטי],
payments.P_numPayments AS [מספר תשלומים], payments.P_paymentMethod AS
[אמצעי תשלום], payments.P_fileAttached AS [קובץ מצורף], payments.P_creditCard
AS [תוקף כרטיס אשראי], payments.P_validity AS [מספר כרטיס אשראי], payments.P_CVC
AS [CVC כרטיס אשראי], payments.P_status AS [סטטוס תשלום] FROM clients INNER
JOIN (orders INNER JOIN payments ON orders.O_Kod = payments.P_orderKod)
ON clients.C_kod = orders.O_clientKod;");
}
public void UpDatePayment(Payments p)
{
p.FillDataRow();
Dal.Save(table.TableName);
}
}
```

```
public void AddPayment(Payments p)
{
    //יצירת שורה בתבנית של הטבלה
    p.dr = table.NewRow();
    //מכניסה את התכונות של העצם לשורה
    p.FillDataRow();
    //מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
    table.Rows.Add(p.dr);
    //עדכון אקסס
    Dal.Save(table.TableName);
}
}
```

המחלקה WorkersTbl

```
internal class WorkersTbl:GeneralTbl
{
    public WorkersTbl() : base("Workers")
    { }
    public DataTable getalltable()
    {
        return Dal.GetDisplayTable("SELECT Workers.W_kod AS[קוד עובד],
Workers.W_lastName AS[שם משפחה], Workers.W_firstName AS[שם פרטי],
Workers.W_phone AS[פלאפון], Workers.W_cityKod AS[קוד עיר], Workers.W_adress
AS[כתובת], Workers.W_dateOfBirth AS[תאריך לידה], Workers.W_Mail AS[מייל],
Workers.W_status AS[סטטוס]\r\nFROM Workers;\r\n");
    }
    public void UpDateWorkers(Workers cl)
    {
        cl.FillDataRow();
        Dal.Save(table.TableName);
    }
}
```

```
public void AddWorkwrs(Workers cl)
{
    //יצירת שורה בתבנית של הטבלה
    cl.dr = table.NewRow();
    //מכניסה את התכונות של העצם לשורה
    cl.FillDataRow();
    //מדביק את השורה שמכילה את התכונות שמכילה העצם החדש לטבלת קטגוריות
    table.Rows.Add(cl.dr);
    //עדכון אקסס
    Dal.Save(table.TableName);
}
}
```

מחלקת: nameTableRow

המחלקה Cancels

```
internal class Cancels
{
    public DataRow dr;
    private int CA_kod;
```

```
private string CA_workerKod;
private string CA_date;

public Cancels() { }
public Cancels(DataRow drc)
{
    this.CA_kod = int.Parse(drc["CA_kod"].ToString());
    this.CA_workerKod = drc["CA_workerKod"].ToString();
    this.CA_date = drc["CA_date"].ToString();
    this.dr = drc;
}
public int FCA_kod
{
    set
    {
        if (Legal.IsNumber(value.ToString()))
            this.CA_kod = value;
        else
            throw new Exception("קוד לא תקין");
    }
    get
    {
        return this.CA_kod;
    }
}
public string FCA_workerKod
{
    set
    {
        if (Legal.IsNumber(value))
            this.CA_workerKod = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.CA_workerKod;
    }
}
public string FCA_date
{
    set
    {
        if (Legal.IsNumber(value))
            this.CA_date = value;
        else
            throw new Exception("הזן מספר בלבד");
    }
    get
    {
        return this.CA_date;
    }
}
public void FillDataRow()
{
}
```

```
this.dr["CA_kod"] = this.CA_kod;
this.dr["CA_workerKod"] = this.CA_workerKod;
this.dr["CA_date"] = this.CA_date;

}
}
```

המחלקה Cities

```
internal class Cities
{
    public DataRow dr;
    private int CI_kod;
    private string CI_name;
    public Cities() { }
    public Cities(DataRow drc)
    {
        this.CI_kod = int.Parse(drc["CI_kod"].ToString());
        this.CI_name = drc["CI_name"].ToString();
        this.dr = drc;
    }
    public int FCI_code
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
                this.CI_kod = value;
            else
                throw new Exception("הקוד שגוי");
        }
        get
        {
            return this.CI_kod;
        }
    }

    public string FCI_name
    {
        set
        {
            if (Legal.IsHebrew(value))
                this.CI_name = value;
            else
                throw new Exception("הזן אותיות עבריות בלבד");
        }
        get
        {
            return this.CI_name;
        }
    }
    public void FillDataRow()
    {
        this.dr["CI_kod"] = this.CI_kod;
        this.dr["CI_name"] = this.CI_name;
    }
}
```


המחלקה Clients

```
internal class Clients
{
    public DataRow dr;
    private int C_kod;
    private string C_id;
    private string C_lastName;
    private string C_firstName;
    private int C_cityKod;
    private string C_address;
    private string C_eMail;
    private string C_pel;
    private string C_status;

    public Clients() { }
    public Clients(DataRow drc)
    {
        this.C_kod = int.Parse(drc["C_kod"].ToString());
        this.C_id = drc["C_id"].ToString();
        this.C_lastName = drc["C_lastName"].ToString();
        this.C_firstName = drc["C_firstName"].ToString();
        this.C_cityKod = int.Parse(drc["C_cityKod"].ToString());
        this.C_address = drc["C_address"].ToString();
        this.C_eMail = drc["C_eMail"].ToString();
        this.C_pel = drc["C_pel"].ToString();
        this.C_status = drc["C_status"].ToString();

        this.dr = drc;
    }
    public int FC_kod
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
                this.C_kod = value;
            else
                throw new Exception("קוד לא תקין");
        }
    }
    public string FC_eMail
    {
        set
        {
            if (Legal.CheackMail(value))
                this.C_eMail = value;
            else
                throw new Exception("הזן כתובת מייל תקינה");
        }
        get
        {
            return this.C_eMail;
        }
    }
}
```

```

    }
    public string FC_id
    {
        set
        {
            if (Legal.LegalId(value))
                this.C_id = value;
            else
                throw new Exception("מספר זהות שגוי");
        }
        get
        {
            return this.C_id;
        }
    }
    public string FC_lastName
    {
        set
        {
            if (Legal.IsHebrew(value))
                this.C_lastName = value;
            else
                throw new Exception("הכנס אותיות עבריות בלבד");
        }
        get
        {
            return this.C_lastName;
        }
    }
    public string FC_firstName
    {
        set
        {
            if (Legal.IsHebrew(value))
                this.C_firstName = value;
            else
                throw new Exception("הכנס אותיות עבריות בלבד");
        }
        get
        {
            return this.C_firstName;
        }
    }
    public string FC_address
    {
        set
        {
            this.C_address = value;
        }
        get
        {
            return this.C_address;
        }
    }

    public string FC_pe1
    {

```

```

set
{
if (Legal.IsCellPhone(value))
this.C_pel = value;
else
throw new Exception("מספר לא תקין");
}
get
{
return this.C_pel;
}
}

public int FC_cityKod
{
set
{
if (Legal.IsNumber(value.ToString()))
this.C_cityKod = value;
else
throw new Exception("קוד לא תקין");
}
get
{
return this.C_cityKod;
}
}
public string FC_status
{
set
{
this.C_status = value;
}
get
{
return this.C_status;
}
}

public void FillDataRow()
{
this.dr["C_kod"] = this.C_kod;
this.dr["C_id"] = this.C_id;
this.dr["C_lastName"] = this.C_lastName;

this.dr["C_firstName"] = this.C_firstName;
this.dr["C_cityKod"] = this.C_cityKod;
this.dr["C_adress"] = this.C_adress;
this.dr["C_eMail"] = this.C_eMail;
this.dr["C_pel"] = this.C_pel;
this.dr["C_status"] = this.C_status;
}
}

internal class KindOrder
{

```

המחלקה KindOrder

```

public DataRow dr;
private int K_kod;
private string K_describing;
private string K_price;
private string k_timeWorking;
private string K_picture;
public KindOrder() { }
public KindOrder(DataRow drc)
{
    this.K_kod = int.Parse(drc["K_kod"].ToString());
    this.K_describing = drc["K_describing"].ToString();
    this.K_price = drc["K_price"].ToString();
    this.k_timeWorking = drc["k_timeWorking"].ToString();
    this.dr = drc;
}
public int FK_kod
{
    set
    {
        if (Legal.IsNumber(value.ToString()))
            this.K_kod = value;
        else
            throw new Exception("קוד לא תקין");
    }
    get
    {
        return this.K_kod;
    }
}
public string FK_describing
{
    set
    {
        if (Legal.IsHebrew(value))
            this.K_describing = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.K_describing;
    }
}
public string FK_price
{
    set
    {
        if (Legal.IsNumber(value))
            this.K_price = value;
        else
            throw new Exception("הזן מספר בלבד");
    }
    get
    {
        return this.K_price;
    }
}

```

```
public string Fk_timeWorking
{
    set
    {
        if (Legal.IsNumber(value))
            this.k_timeWorking = value;
        else
            throw new Exception("הזן מספר בלבד");
    }
    get
    {
        return this.k_timeWorking;
    }
}
public void FillDataRow()
{
    this.dr["K_kod"] = this.K_kod;
    this.dr["K_describing"] = this.K_describing;
    this.dr["K_price"] = this.K_price;
    this.dr["k_timeWorking"] = this.k_timeWorking;
}
}
```

המחלקה Mittings

```
internal class Mittings
{
    public DataRow dr;
    private int M_kod;
    private string M_date;
    private string M_hour;
    private int M_clientKod;
    private string M_status;
    public Mittings() { }
    public Mittings(DataRow drc)
    {
        this.M_kod = int.Parse(drc["M_kod"].ToString());
        this.M_date = drc["M_date"].ToString();
        this.M_hour = drc["M_hour"].ToString();
        this.M_clientKod = int.Parse(drc["M_clientKod"].ToString());
        this.M_status = drc["M_status"].ToString();

        this.dr = drc;
    }
    public int FM_kod
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))

                this.M_kod = value;
            else
                throw new Exception("קוד לא תקין");
        }
        get
        {
            return this.M_kod;
        }
    }
}
```

```

    }
    }
    public string FM_hour
    {
        set
        {
            this.M_hour = value;
        }
        get
        {
            return this.M_hour;
        }
    }
    public int FM_clientKod
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
            this.M_clientKod = value;
            else
            throw new Exception("קוד לא תקין");
        }
        get
        {
            return this.M_clientKod;
        }
    }
    public string FM_date
    {
        set
        {
            this.M_date = DateTime.Parse(value).ToShortDateString().ToString();
        }
        get
        {
            return this.M_date;
        }
    }
    public string FM_status
    {
        set
        {
            this.M_status = value;
        }
        get
        {
            return this.M_status;
        }
    }
    public void FillDataRow()
    {
        this.dr["M_kod"] = this.M_kod;
        this.dr["M_date"] = this.M_date;

        this.dr["M_hour"] = this.M_hour;
        this.dr["M_date"] = this.M_date;
        this.dr["M_clientKod"] = this.M_clientKod;
        this.dr["M_status"] = this.M_status;
    }

```

```
}
}
```

OrderDeitals

```
internal class OrderDeitals
{
    public DataRow dr;
    private int DO_kod;
    private int DO_kodOredr;
    private int DO_KindOredrKod;
    private string DO_Describing;
    private string OD_status;
    public OrderDeitals() { }
    public OrderDeitals(DataRow drc)
    {
        this.DO_kod = int.Parse(drc["DO_kod"].ToString());
        this.DO_kodOredr = int.Parse(drc["DO_kodOredr"].ToString());
        this.DO_KindOredrKod = int.Parse(drc["DO_KindOredrKod"].ToString());
        this.DO_Describing = drc["DO_Describing"].ToString();
        this.OD_status = drc["OD_status"].ToString();
        this.dr = drc;
    }
    public int FDO_kod
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
                this.DO_kod = value;
            else
                throw new Exception("הקוד שגוי");
        }
        get
        {
            return this.DO_kod;
        }
    }
    public int FDO_kodOredr
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
                this.DO_kodOredr = value;
            else
                throw new Exception("הקוד שגוי");
        }
        get
        {
            return this.DO_kodOredr;
        }
    }
    public int FDO_KindOredrKod
    {
        set
        {
            if (Legal.IsNumber(value.ToString()))
                this.DO_KindOredrKod = value;
        }
    }
}
```

```

else
throw new Exception("הקוד שגוי");
}
get
{
return this.DO_KindOredrKod;
}
}
public string FDO_Describing
{
set
{
if (Legal.IsHebrew(value))
this.DO_Describing = value;
else
throw new Exception("הזן אותיות עבריות בלבד");
}
get
{
return this.DO_Describing;
}
}
public string FOD_status
{
set
{
if (Legal.IsHebrew(value))
this.OD_status = value;
else
throw new Exception("הזן אותיות עבריות בלבד");
}
get
{
return this.OD_status;
}
}
public void FillDataRow()
{
this.dr["DO_kod"] = this.DO_kod;
this.dr["DO_KindOredrKod"] = this.DO_KindOredrKod;
this.dr["DO_Describing"] = this.DO_Describing;
this.dr["DO_kodOredr"] = this.DO_kodOredr;
this.dr["OD_status"] = this.OD_status;
}
}

```

```

internal class Orders
{
public DataRow dr;
private int O_Kod;
private int O_clientKod;
private string O_dateOfOrder;
private string O_dateOFEnd;
private string O_numWorkHour;
private string O_amountToPay;
private string O_status;

```

המחלקה Orders


```
private string O_GraphicArtistKod;
private string O_numOfMittings;
public Orders() { }
public Orders(DataRow drc)
{
    this.O_Kod = int.Parse(drc["O_Kod"].ToString());
    this.O_clientKod = int.Parse(drc["O_clientKod"].ToString());
    this.O_dateOfOrder = drc["O_dateOfOrder"].ToString();
    this.O_dateOfEnd = drc["O_dateOfEnd"].ToString();
    this.O_numWorkHour = drc["O_numWorkHour"].ToString();
    this.O_amountToPay = drc["O_amountToPay"].ToString();
    this.O_status = drc["O_status"].ToString();
    this.O_GraphicArtistKod = drc["O_GraphicArtistKod"].ToString();
    this.O_numOfMittings = drc["O_numOfMittings"].ToString();

    this.dr = drc;
}

public string FO_dateOfEnd
{
    set
    {
        this.O_dateOfEnd =
        DateTime.Parse(value).ToShortDateString().ToString();
    }
    get
    {
        return this.O_dateOfEnd;
    }
}

public string FO_dateOfOrder
{
    set
    {
        this.O_dateOfOrder =
        DateTime.Parse(value).ToShortDateString().ToString();
    }
    get
    {
        return this.O_dateOfOrder;
    }
}

public string FO_numWorkHour
{
    set
    {
        if (Legal.IsNumber(value))
            this.O_numWorkHour = value;
        else
            throw new Exception("הזן מספר בלבד");
    }
    get
    {
        return this.O_numWorkHour;
    }
}

public string FO_numOfMittings
{

```

```

set
{

if (Legal.IsNumber(value))
this.O_numOfMittings = value;
else
throw new Exception("הזן מספר בלבד");
}
get
{
return this.O_numOfMittings;
}
}
public string FO_GraphicArtistKod
{
set
{
if (Legal.IsNumber(value))
this.O_GraphicArtistKod = value;
else
throw new Exception("הזן מספר בלבד");
}
get
{
return this.O_GraphicArtistKod;
}
}
public string FO_amountToPay
{
set
{
if (Legal.IsNumber(value))
this.O_amountToPay = value;
else
throw new Exception("הזן מספר בלבד");
}
get
{
return this.O_amountToPay;
}
}
public int FO_Kod
{
set
{
if (Legal.IsNumber(value.ToString()))
this.O_Kod = value;
else
throw new Exception("קוד לא תקין");
}
get
{
return this.O_Kod;
}
}
public int FO_clientKod
{
set

```

```
{
if (Legal.IsNumber(value.ToString()))
this.O_clientKod = value;
else
throw new Exception("קוד לא תקין");
}
get
{
return this.O_clientKod;
}
}
public string FO_status
{
set
{
this.O_status = value;
}
get
{
return this.O_status;
}
}
public void FillDataRow()
{
this.dr["O_Kod"] = this.O_Kod;
this.dr["O_clientKod"] = this.O_clientKod;
this.dr["O_dateOfOrder"] = this.O_dateOfOrder;
this.dr["O_dateOFEnd"] = this.O_dateOFEnd;
this.dr["O_numWorkHour"] = this.O_numWorkHour;
this.dr["O_amountToPay"] = this.O_amountToPay;
this.dr["O_status"] = this.O_status;
this.dr["O_GraphicArtistKod"] = this.O_GraphicArtistKod;
this.dr["O_numOfMittings"] = this.O_numOfMittings;
}
}
```

המחלקה Payments

```
internal class Payments
{
public DataRow dr;
private int P_kod;
private int P_orderKod;
private string P_numPayments;
private string P_paymentMethod;
private string P_fileAttached;
private string P_creditCard;
private string P_validity;
private string P_CVC;
private string P_status;
public Payments() { }
public Payments(DataRow drc)
{
this.P_kod = int.Parse(drc["P_kod"].ToString());
this.P_orderKod = int.Parse(drc["P_orderKod"].ToString());
this.P_numPayments = drc["P_numPayments"].ToString();
this.P_paymentMethod = drc["P_paymentMethod"].ToString();
}
```

```

this.P_fileAttached = drc["P_fileAttached"].ToString();
this.P_creditCard = drc["P_creditCard"].ToString();
this.P_validity = drc["P_validity"].ToString();
this.P_CVC = drc["P_CVC"].ToString();
this.P_status = drc["P_status"].ToString();
this.dr = drc;
}
public string FP_status

{
    set
    {
        if (Legal.IsHebrew(value))
            this.P_status = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.P_status;
    }
}
public string FP_fileAttached
{
    set
    {
        if (Legal.IsHebrew(value))
            this.P_fileAttached = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.P_fileAttached;
    }
}
public string FP_paymentMethod
{
    set
    {
        if (Legal.IsHebrew(value))
            this.P_paymentMethod = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.P_paymentMethod;
    }
}
public int FP_kod
{
    set
    {
        if (Legal.IsNumber(value.ToString()))
            this.P_kod = value;
        else
            throw new Exception("קוד לא תקין");
    }
}

```

```

get
{
return this.P_kod;
}
}
public int FP_orderKod
{
set
{
if (Legal.IsNumber(value.ToString()))
this.P_orderKod = value;

else
throw new Exception("קוד לא תקין");
}
}
get
{
return this.P_orderKod;
}
}
public string FP_numPayments
{
set
{
if (Legal.IsNumber(value))
this.P_numPayments = value;
else
throw new Exception("הזן מספר בלבד");
}
}
get
{
return this.P_numPayments;
}
}
public string FP_creditCard
{
set
{
if (Legal.IsNumber(value))
this.P_creditCard = value;
else
throw new Exception("הזן מספר בלבד");
}
}
get
{
return this.P_creditCard;
}
}
public string FP_CVC
{
set
{
if (Legal.IsNumber(value))
this.P_CVC = value;
else
throw new Exception("הזן מספר בלבד");
}
}
get
{

```

```

return this.P_CVC;
}
}

public string FP_validity
{
set
{
this.P_validity = value;
}
get
{
return this.P_validity;
}
}

}
public void FillDataRow()
{
this.dr["P_kod"] = this.P_kod;
this.dr["P_orderKod"] = this.P_orderKod;
this.dr["P_numPayments"] = this.P_numPayments;
this.dr["P_fileAttached"] = this.P_fileAttached;
this.dr["P_creditCard"] = this.P_creditCard;
this.dr["P_validity"] = this.P_validity;
this.dr["P_CVC"] = this.P_CVC;
this.dr["P_status"] = this.P_status;
this.dr["P_paymentMethod"] = this.P_paymentMethod;
}
}
}

```

המחלקה Workers

```

internal class Workers
{
public DataRow dr;
private int W_kod;
private string W_FirstName;
private string W_lastName;
private string W_phone;
private int W_cityKod;
private string W_address;
private string W_dateOfBirth;
private string W_Mail;
private string W_status;

public Workers() { }
public Workers(DataRow drc)
{
this.W_kod = int.Parse(drc["W_kod"].ToString());
this.W_FirstName = drc["W_FirstName"].ToString();
this.W_lastName = drc["W_lastName"].ToString();
this.W_phone = drc["W_phone"].ToString();
this.W_cityKod = int.Parse(drc["W_cityKod"].ToString());
this.W_address = drc["W_address"].ToString();
this.W_dateOfBirth = drc["W_dateOfBirth"].ToString();
this.W_Mail = drc["W_Mail"].ToString();
this.W_status = drc["W_status"].ToString();
}
}

```

```

this.dr = drc;
}
public int FW_kod
{
    set
    {
        if (Legal.IsNumber(value.ToString()))
            this.W_kod = value;
        else
            throw new Exception("קוד לא תקין");
    }
    get
    {
        return this.W_kod;
    }
}

public int FW_cityKod
{
    set
    {
        if (Legal.IsNumber(value.ToString()))
            this.W_cityKod = value;
        else
            throw new Exception("קוד לא תקין");
    }
    get
    {
        return this.W_cityKod;
    }
}
public string FW_FirstName
{
    set
    {
        if (Legal.IsHebrew(value))
            this.W_FirstName = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.W_FirstName;
    }
}
public string FW_lastName
{
    set
    {
        if (Legal.IsHebrew(value))
            this.W_lastName = value;
        else
            throw new Exception("הכנס אותיות עבריות בלבד");
    }
    get
    {
        return this.W_lastName;
    }
}

```

```

    }
    public string FW_phone
    {
        set
        {
            if (Legal.IsNumber(value))
                this.W_phone = value;
            else
                throw new Exception("הכנס מספרים בלבד");
        }
        get
        {
            return this.W_phone;
        }
    }
    public string FW_dateOfBirth
    {
        set
        {
            this.W_dateOfBirth = value;
        }
        get
        {
            return this.W_dateOfBirth;
        }
    }
    public string FW_Mail
    {
        set
        {
            if (Legal.CheackMail(value))
                this.W_Mail = value;
            else
                throw new Exception("הזן כתובת מייל תקינה");
        }
        get
        {
            return this.W_Mail;
        }
    }
    public string FW_status
    {
        set
        {
            this.W_status = value;
        }
        get
        {
            return this.W_status;
        }
    }
    public string FW_adress
    {
        set
        {
            this.W_adress = value;
        }
        get
    }

```



```
{  
return this.W_adress;  
}  
}  
public void FillDataRow()  
{  
this.dr["W_kod"] = this.W_kod;  
this.dr["W_FirstName"] = this.W_FirstName;  
this.dr["W_lastName"] = this.W_lastName;  
this.dr["W_phone"] = this.W_phone;  
this.dr["W_cityKod"] = this.W_cityKod;  
this.dr["W_adress"] = this.W_adress;  
this.dr["W_dateOfBirth"] = this.W_dateOfBirth;  
this.dr["W_Mail"] = this.W_Mail;  
this.dr["W_status"] = this.W_status;  
}  
}
```

שכבת היישום:

טופס של כניסת משתמש:

```
public partial class Welcome : Designer
{
    Form1 f1;
    WorkersTbl wtbl = new WorkersTbl();
    public Welcome()
    {
        InitializeComponent();
    }

    private void Welcome_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        this.Text = "ברוכים הבאים :)";
        comboBox1.Visible = false;
        FillCmbGrafics();
    }

    private void rbGrafics_CheckedChanged(object sender, EventArgs e)
    {
        if (rbGrafics.Checked)
            comboBox1.Visible = true;
        else
            comboBox1.Visible = false;
    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
    }

    public void FillCmbGrafics()
    {
        comboBox1.SelectedIndexChanged -= comboBox1_SelectedIndexChanged;
        comboBox1.DataSource = wtbl.getalltable();
        comboBox1.DisplayMember = "שם";
        comboBox1.ValueMember = "קוד עובד";
        comboBox1.SelectedIndexChanged += comboBox1_SelectedIndexChanged;
        comboBox1.SelectedIndex = 0;
    }

    private void btnEnter_Click(object sender, EventArgs e)
    {
        if (rbPricipal.Checked)
        {
            if (textBox1.Text == Properties.Settings.Default.s1)
            {
                f1 = new Form1("Pricipal");
                f1.ShowDialog();
                textBox1.Text = "";
            }
            else
            {
                MessageBox.Show("סיסמת מנהל שגויה, נסה שנית.");
                textBox1.Text = "";
            }
        }
        else
    }
}
```

```
{
if(rbsecretary.Checked)
{
if (textBox1.Text == Properties.Settings.Default.s2)
{
f1 = new Form1("secretary");
f1.ShowDialog();
textBox1.Text = "";
}
else
{
MessageBox.Show("סיסמת מזכירה שגויה, נסה שנית.");
textBox1.Text = "";
}
}
else
{
DataRow dr = wtbl.Find("W_lastName", comboBox1.Text);
if (textBox1.Text == dr[9].ToString())
{
f1 = new Form1(dr[1].ToString());
f1.ShowDialog();
textBox1.Text = "";
}
else
{
MessageBox.Show("סיסמת גרפיקאית שגויה, נסה שנית.");
textBox1.Text = "";
}
}
}
}
}
```

טופס ראשי:

```
public partial class Form1 : Designer
{
string mishtamesh;
public Form1(string mishtamesh)
{
InitializeComponent();
this.mishtamesh = mishtamesh;
}

private void btncities_Click(object sender, EventArgs e)
{
CitiesShow c1 = new CitiesShow();
c1.ShowDialog();
}

private void btnclients_Click(object sender, EventArgs e)
{
ClientsShow c2 = new ClientsShow();
c2.ShowDialog();
}

private void btnkindorder_Click(object sender, EventArgs e)
{
KindOrderShow c3 = new KindOrderShow();
c3.ShowDialog();
}
```

```

    }

    private void btnmittings_Click(object sender, EventArgs e)
    {
        MittingsShow c4 = new MittingsShow();
        c4.ShowDialog();
    }

    private void btnorders_Click(object sender, EventArgs e)
    {
        OrdersShow c5 = new OrdersShow();
        c5.ShowDialog();
    }

    private void btnpayments_Click(object sender, EventArgs e)
    {
        PaymentsShow c6 = new PaymentsShow();
        c6.ShowDialog();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        CenterToScreen();
    }

    private void btnSchedule_Click(object sender, EventArgs e)
    {
        ViewSchedule ss = new ViewSchedule();
        ss.ShowDialog();
    }

    private void btnWorkers_Click(object sender, EventArgs e)
    {
        WorkersShow c7 = new WorkersShow();
        c7.ShowDialog();
    }

    private void btnOrdetDeitals_Click(object sender, EventArgs e)
    {
        OrderDeitalsShow c8 = new OrderDeitalsShow();
        c8.ShowDialog();
    }

    private void btnCancel_Click(object sender, EventArgs e)
    {
        CancelsShow c9 = new CancelsShow();
        c9.ShowDialog();
    }

    private void btnWorkingForGrafics_Click(object sender, EventArgs e)
    {
        WorkingForGrafics w1 = new WorkingForGrafics();
        w1.ShowDialog();
    }

    public partial class CancelsAddUpDate : Form
    {

```

טופס הוספה ועדכון ביטולים- הצגה

```

string status;
WorkersTbl wtbl = new WorkersTbl();
CancelsTbl caTbl = new CancelsTbl();
Cancels ca;
public CancelsAddUpDate()
{
InitializeComponent();
status = "Add";
ca = new Cancels();
}
public CancelsAddUpDate(int CA_kod)
{
InitializeComponent();
status = "UpDate";
DataRow dr = caTbl.Find("CA_kod", CA_kod);
ca = new Cancels(dr);
FillForm();
}
public void FillForm()
{
txtKodCancel.Text = ca.FCA_kod.ToString();
cmbName.SelectedValue = ca.FCA_workerKod.ToString();
dateTimePicker1.Text = ca.FCA_dateOfStart.ToString();
AddKindOrder.Text = "עדכון";
}
private void CancelsAddUpDate_Load(object sender, EventArgs e)
{
CenterToScreen();
fillCmbName();
if (status == "Add")
{
dateTimePicker1.Value = DateTime.Now;
dateTimePicker2.Value = DateTime.Now;
this.Text = "הוספת ביטול";
txtKodCancel.Text = caTbl.GetNext("CA_kod").ToString();
cmbName.SelectedIndex = 0;
}
else
{
this.Text = "עדכון ביטול";
DataRow drC = wtbl.Find("W_kod", ca.FCA_workerKod);
Workers.Workers w = new Workers.Workers(drC);
cmbName.Text = w.FW_lastName;
dateTimePicker1.Text = ca.FCA_dateOfStart.ToString();
dateTimePicker2.Text = ca.FCA_dateOfEnd.ToString();
}
}
public bool CheckLegal()
{
errorProviderWorkerKod.Clear();
bool degel = true;
ca.FCA_kod = int.Parse(txtKodCancel.Text);
ca.FCA_workerKod = int.Parse(cmbName.SelectedValue.ToString());

try
{
ca.FCA_dateOfStart = dateTimePicker1.Text;
}
}

```

```

catch (Exception ex1)
{
    errorProviderDATE.SetError(dateTimePicker1, ex1.Message);
    degel = false;
}
try
{
    ca.FCA_dateOfEnd = dateTimePicker2.Text;
}
catch (Exception ex1)
{
    errorProviderDATE.SetError(dateTimePicker2, ex1.Message);
    degel = false;
}

return degel;
}

private void AddKindOrder_Click(object sender, EventArgs e)
{
    if (CheckLegal())
    {
        if (status == "Add")
        {
            caTbl.AddCancel(ca);
        }
        else
        {
            caTbl.UpDateCancel(ca);
        }
        this.Close();
    }
}

public void fillCmbName()
{
    cmbName.SelectedIndexChanged -= cmbID_SelectedIndexChanged;
    cmbName.DataSource = wtbl.getalltable();
    cmbName.DisplayMember = "שם משפחה";
    cmbName.ValueMember = "קוד עובד";
    cmbName.SelectedIndexChanged += cmbID_SelectedIndexChanged;
    cmbName.SelectedIndex = 1;
}

private void cmbID_SelectedIndexChanged(object sender, EventArgs e)
{
    //נתונים נוספים של עובד, משתנה כאשר יש שינוי בחירת ת.ז.
    DataRow drc = wtbl.Find("W_kod", cmbName.SelectedValue.ToString());
    Workers.Workers w = new Workers.Workers(drc);
    txtbNameWorker.Text = w.FW_FirstName;
    txtbWorkerPhone.Text = w.FW_phone;
}

public partial class CancelsShow : Form
{
    CancelsTbl catbl;
    public CancelsShow()
    {
        InitializeComponent();
    }
}

```

```
private void CancelsShow_Load(object sender, EventArgs e)
{
    CenterToScreen();
    this.Text = "ביטולים";
    catbl = new CancelsTbl();
    DGCancels.DataSource = catbl.getalltable();
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    CancelsAddUpDate cc = new CancelsAddUpDate();
    cc.ShowDialog();
    DGCancels.DataSource = catbl.getalltable();
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (DGCancels.SelectedRows.Count > 0)
    {
        CancelsAddUpDate a = new
        CancelsAddUpDate(int.Parse(DGCancels.SelectedRows[0].Cells[0].Value.ToString()));
        a.ShowDialog();
        DGCancels.DataSource = catbl.getalltable();
    }
    else
    {
        MessageBox.Show("עליך לבחור שורה");
    }
}
}
```

טופס הוספה ועדכון ערים+ הצגה:

```
public partial class CitiesAddUpdate : Form
{
    //ClientsTbl cTbl = new ClientsTbl();
    CitiesTbl cbb = new CitiesTbl();
    Cities c;
    string status;
    public CitiesAddUpdate()
    {
        InitializeComponent();
        status = "Add";
        c = new Cities();
    }
    public CitiesAddUpdate(int CI_name)
    {
        InitializeComponent();
        status = "Update";
        DataRow dr = cbb.Find("CI_kod", CI_name);
        c = new Cities(dr);
        FillForm();
    }
    public void FillForm()
    {
        txtbkodCity.Text = c.FCi_code.ToString();
        txtbNameCity.Text = c.FCi_name.ToString();
        AddCity.Text = "עדכון";
    }
}
```

```

    }
    private void CitiesAddUpdate_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        if (status == "Add")
        {
            this.Text = "הוספת עיר";
            txtbkodCity.Text = cbb.GetNext("CI_kod").ToString();
        }
        else
        {
            this.Text = "עדכון עיר";
        }
    }
    private void txtkodCity_TextChanged(object sender, EventArgs e)
    {
    }
    public bool CheckLegal()
    {
        ErpNameCity.Clear();
        bool degel = true;
        c.FCi_code = int.Parse(txtbkodCity.Text);
        if (txtbNameCity.Text != "")
        {
            try
            {
                c.FCi_name = txtbNameCity.Text;
            }
            catch (Exception ex1)
            {
                ErpNameCity.SetError(txtbNameCity, ex1.Message);
                degel = false;
            }
        }
        else
        {
            txtbNameCity.BackColor = Color.LightSalmon;
            degel = false;
        }
        return degel;
    }
    private void AddCitiy_Click(object sender, EventArgs e)
    {
        if (CheckLegal())
        {
            if (status == "Add")
            {
                cbb.AddCity(c);
            }
            else
            {
                cbb.UpDateCity(c);
            }
            this.Close();
        }
    }

    private void txtbNameCity_TextChanged(object sender, EventArgs e)
    {
        if (txtbNameCity.BackColor == Color.LightSalmon)
    }

```



```

txtbNameCity.BackColor = Color.White;
}
}

public partial class CitiesShow : Form
{
    CitiesTbl ctbl;
    public CitiesShow()
    {
        InitializeComponent();
    }

    private void CitiesShow_Load(object sender, EventArgs e)
    {
        CenterToScreen();

        this.Text = "ערים";
        ctbl = new CitiesTbl();
        DGCities.DataSource = ctbl.getalltable();
    }

    private void DGCities_CellContentClick(object sender,
        DataGridViewCellEventArgs e)
    {
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        CitiesAddUpdate cc = new CitiesAddUpdate();
        cc.ShowDialog();
        DGCities.DataSource = ctbl.getalltable();
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (DGCities.SelectedRows.Count > 0)
        {
            CitiesAddUpdate a = new
            CitiesAddUpdate(int.Parse(DGCities.SelectedRows[0].Cells[0].Value.ToStr
            ing()));
            a.ShowDialog();
            DGCities.DataSource = ctbl.getalltable();
        }
        else
        {
            MessageBox.Show("עליך לבחור שורה");
        }
    }
}

public partial class ClientsAddUpdate : Form
{
    ClientsTbl cctbl=new ClientsTbl();
    string status;
    Cities.Cities city;
    Clients c = new Clients();
    CitiesTbl citiestbl = new CitiesTbl();
    public ClientsAddUpdate()

```

הוספה ועדכון לקוחות: הצגה:

```
{
InitializeComponent();
status = "Add";
//סטטוס לקוח חדש- פעיל/
rbStatusp.Checked = true;
c = new Clients();
}
public ClientsAddUpdate(string C_id)
{
InitializeComponent();
status = "Update";
DataRow dr = cctbl.Find("C_id", C_id);
c = new Clients(dr);
FillForm();
}
public void FillForm()
{
txtbKodClient.Text = c.FC_kod.ToString();
txtbAdress.Text = c.FC_address.ToString();
txtbID.Text = c.FC_id.ToString();
txtbLastName.Text = c.FC_lastName.ToString();
txtbFirstName.Text = c.FC_firstName.ToString();
txtbMail.Text = c.FC_eMail.ToString();

if (c.FC_pel != "")
{
string[] str1 = c.FC_pel.Split('-');
cmbPhone2.Text = str1[0];
txtbPhone2.Text = str1[1];
}
txtbAdress.Text = c.FC_address.ToString();
btnAddClient.Text = "עדכן";
this.Text = "עדכון לקוח";
if (c.FC_status == "פעיל")
{
rbStatusp.Checked = true;
}
else
{
rbstatusnp.Checked = true;
}

} public void FillCBCity()
{
cmbCity.SelectedIndexChanged -= cmbCity_SelectedIndexChanged;
cmbCity.DataSource = citiestbl.getalltable();
cmbCity.DisplayMember = "שם עיר";
cmbCity.ValueMember = "קוד עיר";
cmbCity.SelectedIndexChanged += cmbCity_SelectedIndexChanged;
cmbCity.SelectedIndex = 0;
}
public bool CheckLegal()
{
errorProviderLastName.Clear();
errorProviderDATEOFBIRTH.Clear();
errorProviderMail.Clear();
errorProviderID.Clear();
errorProviderFirstName.Clear();
errorProviderKODCITY.Clear();
errorProviderPHONE1.Clear();
}
```

```

errorProviderPHONE2.Clear();
errorProviderDATEOFBIRTH.Clear();
errorProviderSTATUS.Clear();
errorProviderADDRESS.Clear();
bool degel = true;
c.FC_cityKod = int.Parse(cmbCity.SelectedValue.ToString());
c.FC_kod = int.Parse(txtbKodClient.Text);
//סטטוס
if (rbStatusp.Checked)
c.FC_status = "פעיל";
else
c.FC_status = "לא פעיל";
//ת.ז.
if (txtbID.Text != "")
{

if (status == "Add" && cctlbl.Find("C_id", txtbID.Text) != null)
{
labellid.Text = "מספר זהות קיים במערכת";
degel = false;
}
else
try
{
c.FC_id = txtbID.Text;
}
catch (Exception ex1)
{
errorProviderID.SetError(txtbID, ex1.Message);
degel = false;
}
else
{
txtbID.BackColor = Color.LightSalmon;
degel = false;
}

//שם פרטי
if (txtbFirstName.Text != "")
{
try
{
c.FC_firstName = txtbFirstName.Text;
}
catch (Exception ex1)
{
errorProviderFirstName.SetError(txtbFirstName, ex1.Message);
degel = false;
}
}
else
{
txtbFirstName.BackColor = Color.LightSalmon;
degel = false;
}
//שם משפחה
if (txtbLastName.Text != "")

try

```

```
{
c.FC_lastname = txtbLastName.Text;
}
catch (Exception ex1)
{
errorProviderLastName.SetError(txtbLastName, ex1.Message);
degel = false;
}
else
{
txtbLastName.BackColor = Color.LightSalmon;
degel = false;
}
//כתובת
if (txtbAddress.Text != "")
try
{
c.FC_address = txtbAddress.Text;
}
catch (Exception ex1)
{
errorProviderADDRESS.SetError(txtbAddress, ex1.Message);
degel = false;
}
else
{
txtbAddress.BackColor = Color.LightSalmon;
degel = false;
}
//מיקל
if (txtbMail.Text != "")

try
{
c.FC_eMail = txtbMail.Text;
}
catch (Exception ex1)
{
errorProviderMail.SetError(txtbMail, ex1.Message);
degel = false;
}
else
{
txtbMail.BackColor = Color.LightSalmon;
degel = false;
}
//פלאפון
if (txtbPhone2.Text != "")
try
{
c.FC_pel = cmbPhone2.Text + "-" + txtbPhone2.Text;
}
catch (Exception ex1)
{
errorProviderPHONE2.SetError(txtbPhone2, ex1.Message);
degel = false;
}
else
{
txtbPhone2.BackColor = Color.LightSalmon;
}
```

```

degel = false;
}

return degel;
}
private void ClientsAddUpdate_Load(object sender, EventArgs e)
{
    CenterToScreen();
    cctbl = new ClientsTbl();
    FillCBCity();

    if (status != "Add")
    {
        this.Text = "עדכון לקוח";
        DataRow drcity = citiestbl.Find("CI_kod", c.FC_cityKod);
        city = new Cities.Cities(drcity);
        cmbCity.Text = city.FCi_name;
    }
    else
    {
        txtbKodClient.Text = cctbl.GetNext("C_kod").ToString();
        this.Text = "הוספת לקוח";
        rbstatusnp.Visible = false;
        rbStatusp.Visible = false;
        lblStatus.Visible = false;
        cmbPhone2.SelectedIndex = 0;
    }
}

private void CITYADD_Click(object sender, EventArgs e)
{
    {
        CitiesAddUpdate C = new CitiesAddUpdate();
        C.ShowDialog();
        cmbCity.SelectedIndexChanged -= cmbCity_SelectedIndexChanged;
        cmbCity.DataSource = citiestbl.getalltable();
        cmbCity.DisplayMember = "שם עיר";
        cmbCity.ValueMember = "קוד עיר";
        cmbCity.SelectedIndexChanged += cmbCity_SelectedIndexChanged;
        cmbCity.SelectedIndex = 0;
        DataRow DR = citiestbl.Find("CI_kod", citiestbl.GetNext("CI_kod") - 1);
        cmbCity.Text = DR["CI_name"].ToString();
    }
}

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

private void btnAddClient_Click(object sender, EventArgs e)
{
    {
        if (CheckLegal())
        {
            if (status == "Add")
            {
                cctbl.AddClients(c);
            }
            else

```

```
{
cctbl.UpDateClients(c);
}
this.Close();
}
}

private void btnAddCitiy_Click(object sender, EventArgs e)
{
CitiesAddUpdate cc = new CitiesAddUpdate();
cc.ShowDialog();
CitiesAddUpdate C = new CitiesAddUpdate();
cmbCity.SelectedIndexChanged -= cmbCity_SelectedIndexChanged;
cmbCity.DataSource = citiestbl.getalltable();
cmbCity.DisplayMember = "שם עיר";
cmbCity.ValueMember = "קוד עיר";
cmbCity.SelectedIndexChanged += cmbCity_SelectedIndexChanged;
cmbCity.SelectedIndex = 0;
DataRow DR = citiestbl.Find("CI_kod", citiestbl.GetNext("CI_kod") - 1);
cmbCity.Text = DR["CI_name"].ToString();
}

private void txtbID_TextChanged(object sender, EventArgs e)
{
if (txtbID.BackColor == Color.LightSalmon)
txtbID.BackColor = Color.White;
}

private void txtbLastName_TextChanged(object sender, EventArgs e)
{
if (txtbLastName.BackColor == Color.LightSalmon)
txtbLastName.BackColor = Color.White;
}

private void txtbFirstName_TextChanged(object sender, EventArgs e)
{
if (txtbFirstName.BackColor == Color.LightSalmon)
txtbFirstName.BackColor = Color.White;
}

private void txtbAdress_TextChanged(object sender, EventArgs e)
{
if (txtbAdress.BackColor == Color.LightSalmon)
txtbAdress.BackColor = Color.White;
}

private void txtbPhone2_TextChanged(object sender, EventArgs e)
{
if (txtbPhone2.BackColor == Color.LightSalmon)
txtbPhone2.BackColor = Color.White;
}

private void txtbMail_TextChanged(object sender, EventArgs e)
{
if (txtbMail.BackColor == Color.LightSalmon)
txtbMail.BackColor = Color.White;
}
}
```

```

public partial class ClientsShow : Form
{
    ClientsTbl cltbl;
    DataTable dtClient;
    public ClientsShow()
    {
        InitializeComponent();
    }

    private void ClientsShow_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        this.Text = "לקוחות";

        cltbl = new ClientsTbl();
        DGClients.DataSource = cltbl.getalltable();
        dtClient = cltbl.getalltable();
    }

    private void menuStrip1_ItemClicked(object sender,
    ToolStripItemClickedEventArgs e)
    {
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        ClientsAddUpdate cc = new ClientsAddUpdate();
        cc.ShowDialog();
        DGClients.DataSource = cltbl.getalltable();
    }

    private void menuStrip1_ItemClicked_1(object sender,
    ToolStripItemClickedEventArgs e)
    {
    }

    private void ToolStripMenuItem_Click_1(object sender, EventArgs e)
    {
        if (DGClients.SelectedRows.Count > 0)
        {
            ClientsAddUpdate a = new
            ClientsAddUpdate(DGClients.SelectedRows[0].Cells[1].Value.ToString());
            a.ShowDialog();
            DGClients.DataSource = cltbl.getalltable();
        }
        else
        {
            MessageBox.Show("עליך לבחור שורה");
        }
    }

    private void btnFresh_Click(object sender, EventArgs e)
    {
        DGClients.DataSource = dtClient;
        txtSearch.Text = "הקש ערך לחיפוש";
    }

    private void txtSearch_TextChanged(object sender, EventArgs e)

```

```
{
if (txtSearch.Text == "")
DGClients.DataSource = dtClient;
else
DGClients.DataSource = Classes.SmartSreach.Search(dtClient,
txtSearch.Text);
}

private void txtSearch_MouseLeave(object sender, EventArgs e)
{
//כשאר העכבר יוצא מתיבת טקסט
if (txtSearch.Text == "")
{
txtSearch.Text = "הקש ערך לחיפוש";
DGClients.DataSource = dtClient;
}
else
if (txtSearch.Text != "הקש ערך לחיפוש")
{
string s = txtSearch.Text;
txtSearch.Text = "הקש ערך לחיפוש";
DGClients.DataSource = Classes.SmartSreach.Search(dtClient, s);
}
}

private void txtSearch_MouseClick(object sender, MouseEventArgs e)
{
txtSearch.Text = "";
}
}
```

טופס הוספה ועדכון קטלוג+ הצגה:

```
public partial class KindOrderAddUpdate : Form
{
string status;
KindOrderTbl kTbl = new KindOrderTbl();
KindOrder k;
public KindOrderAddUpdate()
{
InitializeComponent();
status = "Add";
k = new KindOrder();
}
public KindOrderAddUpdate(int K_kod)
{
InitializeComponent();
status = "UpDate";
DataRow dr = kTbl.Find("K_kod", K_kod);
k = new KindOrder(dr);
FillForm();
}
public void FillForm()
{
txtbKodKindOrder.Text = k.FK_kod.ToString();
txtbNameKindOrder.Text = k.FK_describing.ToString();
txtbPriceKindOrder.Text = k.FK_price.ToString();
txttimeWorking.Text = k.Fk_timeWorking.ToString();

AddKindOrder.Text = "עדכון";
}
```


}

```
private void KindOrderAddUpdate_Load(object sender, EventArgs e)
{
    CenterToScreen();
    if (status == "Add")
    {
        this.Text = "הוספת סוג הזמנה";
        txtbKodKindOrder.Text = kTbl.GetNext("K_kod").ToString();
    }
    else
    {
        this.Text = "עדכון סוג הזמנה";
    }
    public bool CheckLegal()
    {
        ErpDescribKindOrder.Clear();
        erpPriceKindOrder.Clear();
        bool degel = true;
        k.FK_kod = int.Parse(txtbKodKindOrder.Text);
        if (txtbNameKindOrder.Text != "")
        {
            try
            {
                k.FK_describing = txtbNameKindOrder.Text;
            }
            catch (Exception ex1)
            {
                ErpDescribKindOrder.SetError(txtbNameKindOrder, ex1.Message);
                degel = false;
            }
        }
        else
        {
            txtbNameKindOrder.BackColor = Color.LightSalmon;
            degel = false;
        }
        if (txtbPriceKindOrder.Text != "")
        {
            try
            {
                k.FK_price = txtbPriceKindOrder.Text;
            }
            catch (Exception ex1)
            {
                erpPriceKindOrder.SetError(txtbPriceKindOrder, ex1.Message);
                degel = false;
            }
        }
        else
        {
            txtbPriceKindOrder.BackColor = Color.LightSalmon;
            degel = false;
        }
        if (txttimeWorking.Text != "")
        {
            try
            {
                k.Fk_timeWorking = txttimeWorking.Text;
            }
        }
    }
}
```

```

    }
    catch (Exception ex1)
    {
        erpPriceKindOrder.SetError(txttimeWorking, ex1.Message);
        degel = false;
    }
    }
    else
    {
        txttimeWorking.BackColor = Color.LightSalmon;
        degel = false;
    }
    return degel;
}

private void AddKindOrder_Click(object sender, EventArgs e)
{
    if (CheckLegal())
    {
        if (status == "Add")
        {
            kTbl.AddKindOrder(k);
        }
        else
        {
            kTbl.UpDateKindOrder(k);
        }
        this.Close();
    }
}

private void label1_Click(object sender, EventArgs e)
{
}

private void txtbNameKindOrder_TextChanged(object sender, EventArgs e)
{
    if (txtbNameKindOrder.BackColor == Color.LightSalmon)
        txtbNameKindOrder.BackColor = Color.White;
}

private void txtbPriceKindOrder_TextChanged(object sender, EventArgs e)
{
    if (txtbPriceKindOrder.BackColor == Color.LightSalmon)
        txtbPriceKindOrder.BackColor = Color.White;
}

private void txttimeWorking_TextChanged(object sender, EventArgs e)
{
    if (txttimeWorking.BackColor == Color.LightSalmon)
        txttimeWorking.BackColor = Color.White;
}
} public partial class KindOrderShow : Form
{
    KindOrderTbl ktbl;
    public KindOrderShow()
    {
        InitializeComponent();
    }
}

```

```
private void KindOrderShow_Load(object sender, EventArgs e)
{
    CenterToScreen();
    this.Text = "סוגי הזמנות";
    ktbl = new KindOrderTbl();
    DGKindOrders.DataSource = ktbl.getalltable();
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    KindOrderAddUpdate cc = new KindOrderAddUpdate();
    cc.ShowDialog();
    DGKindOrders.DataSource = ktbl.getalltable();
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (DGKindOrders.SelectedRows.Count > 0)
    {
        KindOrderAddUpdate a = new
        KindOrderAddUpdate(int.Parse(DGKindOrders.SelectedRows[0].Cells[0].Value.ToString()));
        a.ShowDialog();
        DGKindOrders.DataSource = ktbl.getalltable();
    }
    else
    {
        MessageBox.Show("עליך לבחור שורה");
    }
}
}
```

טופס הוספה ועדכון פגישות+ הצגה:

```
public partial class MittingsAddUpdate : Form
{
    string status;
    ClientsTbl ctbl = new ClientsTbl();
    Mittings m = new Mittings ();
    MittingsTbl mbb = new MittingsTbl();
    DataRow dr;
    public MittingsAddUpdate()
    {
        InitializeComponent();
        status = "Add";
    }
    public MittingsAddUpdate(int Mittingkod)
    {
        InitializeComponent();
        status = "Update";
        DataRow dr = mbb.Find("M_kod", Mittingkod);
        m = new Mittings(dr);
        FillForm();
    }
    public MittingsAddUpdate(string date, string hour)
    {
        InitializeComponent();
        status = "Add";
        dateTimePickerMitting.Text = date;
        cmbHour.Text = hour.Substring(0, 1);
        cmbMoment.Text = hour.Substring(2,3);
    }
}
```

```

}
public bool CheckLegal()
{
    errorProviderDate.Clear();
    errorProviderHour.Clear();
    errorProviderId.Clear();
    errorProviderName.Clear();
    errorProviderPhone.Clear();
    errorProviderStatus.Clear();
    bool degel = true;
    m.FM_kod = int.Parse(txtbkodMitting.Text.ToString());
    m.FM_clientKod = int.Parse(cmbID.Selected.Value.ToString());
    //סטטוס
    if (rbStatusMittingYes.Checked)
        m.FM_status = "בוצע";
    else
        m.FM_status = "לא בוצע";

    //תאריך
    try
    {
        m.FM_date = dateTimePickerMitting.Text;
    }
    catch (Exception ex1)
    {
        errorProviderDate.SetError(dateTimePickerMitting, ex1.Message);
        degel = false;
    }

    //שעה
    if (cmbMoment.Text.ToString() == "00" || cmbMoment.Text.ToString() ==
        "30")
    {
        if (cmbHour.Text.ToString() == "08" || cmbHour.Text.ToString() == "09"
            || cmbHour.Text.ToString() == "10" || cmbHour.Text.ToString() == "11"
            || cmbHour.Text.ToString() == "12" || cmbHour.Text.ToString() == "13"
            || cmbHour.Text.ToString() == "14" || cmbHour.Text.ToString() == "15")
        {
            try
            {
                m.FM_hour = cmbHour.Text.ToString() + ":" + cmbMoment.Text.ToString();
            }
            catch (Exception ex1)
            {
                errorProviderHour.SetError(cmbHour, ex1.Message);
                degel = false;
            }
        }
        else
        {
            cmbHour.BackColor = Color.LightSalmon;
        }
        else
        {
            cmbMoment.BackColor = Color.LightSalmon;
            degel = false;
        }
    }

    return degel;
}
public void FillForm()

```

```
{
Clients.Clients cl;
dateTimePickerMitting.Value = DateTime.Parse(m.FM_date);
txtbkodMitting.Text = m.FM_kod.ToString();
cmbHour.Text = m.FM_hour.Substring(0,2);
cmbMoment.Text = m.FM_hour.Substring(3, 2);

//שמירת שם לקוח (ולא ת.ז), וגם טלפון

//שמירת שורה של פרטי הלקוח לפי הקוד שלו בפגישה
drr = ctbl.Find("C_kod", m.FM_clientKod);
//שמירת פרטי לקוח בתוך עצם מסוג לקוח
cl = new Clients.Clients(drr);

//// לתקן- תעודת זהות בחורה לפי נתוני לקוח
//cmbID.SelectedIndex = cl.FC_id;
//txtbClientName.Text = cl.FC_firstName + " " + cl.FC_lastName;
//txtbClientPhone.Text = cl.FC_pel.ToString();
//if (cl.FC_pel != "")
//{
//    txtbClientPhone= cl.FC_pel;
//}

//סטטוס פגישה
if (m.FM_status == "בוצעה")
{
rbStatusMittingYes.Checked = true;
}
else
{
rbStatusMittingNo.Checked = true;
}
btnAddMitting.Text = "עדכן";
this.Text = "עדכון פגישה";
}
private void MitingsAddUpdate_Load(object sender, EventArgs e)
{
CenterToScreen();
fillCmbID();

if (status == "Add")
{
dateTimePickerMitting.Value = DateTime.Now;
txtbkodMitting.Text = mbb.GetNext("M_kod").ToString();
this.Text = "הוספת פגישה";
rbStatusMittingNo.Visible = false;
rbStatusMittingYes.Visible = false;
lblStatusMitting.Visible = false;
cmbHour.SelectedIndex = 1;
cmbMoment.SelectedIndex = 1;
}
if (status == "Update")
{
this.Text = "עדכון פגישה";
DataRow drC = ctbl.Find("C_kod", m.FM_clientKod);
Clients.Clients c = new Clients.Clients(drC);
cmbID.Text = c.FC_id;
}
txtbClientName.Enabled = false;
txtbClientPhone.Enabled = false;
```

```

}

private void AddMitting_Click(object sender, EventArgs e)
{
    if (CheckLegal())
    {
        if (status == "Add")
        {
            mbb.AddMitting(m);
        }
        else
        {
            mbb.UpDateMitting(m);
        }
        this.Close();
    }
}

private void cmbID_SelectedIndexChanged(object sender, EventArgs e)
{
    //נתונים נוספים של הלקוח, משתנה כאשר יש שינוי בתיבת בחירת ת.ז.
    DataRow drc = ctbl.Find("C_kod", cmbID.SelectedValue.ToString());
    Clients.Clients c = new Clients.Clients(drc);
    txtbClientName.Text = c.FC_firstName + " " + c.FC_lastName;
    txtbClientPhone.Text = c.FC_pel;
}

public void fillCmbID()
{
    cmbID.SelectedIndexChanged -= cmbID_SelectedIndexChanged;
    cmbID.DataSource = ctbl.GetTable();
    cmbID.DisplayMember = "C_id";
    cmbID.ValueMember = "C_kod";
    cmbID.SelectedIndexChanged += cmbID_SelectedIndexChanged;
    cmbID.SelectedIndex = 1;
}

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

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

}

public partial class MittingsShow : Form
{
    MittingsTbl mtbl;
    public MittingsShow()
    {
        InitializeComponent();
    }

    private void MittingsShow_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        this.Text = "פגישות";
        mtbl = new MittingsTbl();
        DGMitings.DataSource = mtbl.getalltable();
    }
}

```

```

}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    MittingsAddUpdate cc = new MittingsAddUpdate();
    cc.ShowDialog();
    DGMittings.DataSource = mtbl.getalltable();
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (DGMittings.SelectedRows.Count > 0)
    {
        MittingsAddUpdate m = new
        MittingsAddUpdate(int.Parse(DGMittings.SelectedRows[0].Cells[0].Value.T
        oString()));
        m.ShowDialog();
        DGMittings.DataSource = mtbl.getalltable();
    }
    else
    {
        MessageBox.Show("עליך לבחור שורה");
    }
}

private void ToolStripMenuItem_Click(object sender, EventArgs
e)
{
    if (DGMittings.SelectedRows.Count > 0)
    {
        MittingsTbl mittbl = new MittingsTbl();
        DataTable dt = mittbl.GetTable();
        DataRow dr = mtbl.Find("M_kod",
        DGMittings.SelectedRows[0].Cells[0].Value.ToString());
        Mittings m = new Mittings(dr);
        m.FM_status = "ברצעה";
        mtbl.UpDateMitting(m);
        DGMittings.DataSource = mtbl.getalltable();
    }
    else
    {
        MessageBox.Show("עליך לבחור שורה");
    }
}
}


```

טופס הוספה ועדכון פרטי הזמנה: הצגה:

```

public partial class OrderDeitalsAddUpDate : Form
{
    public OrderDeitalsAddUpDate()
    {
        InitializeComponent();
    }

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

```

```
private void txtbNameKindOrder_TextChanged(object sender, EventArgs e)
{
    if (txtbNameKindOrder.BackColor == Color.LightSalmon)
        txtbNameKindOrder.BackColor = Color.White;
}

private void txtTimeWorking_TextChanged(object sender, EventArgs e)
{
    if (txtTimeWorking.BackColor == Color.LightSalmon)
        txtTimeWorking.BackColor = Color.White;
}
}
```

טופס הוספה ועדכון הזמנה: הצגה:

```
public partial class OrdersAddUpdate : Form
{
    DataTable dt;
    MittingsTbl mtbl = new MittingsTbl();
    Orders o = new Orders();
    string status;
    OrdersTbl otbl = new OrdersTbl();
    ClientsTbl ctbl = new ClientsTbl();
    KindOrderTbl ktbl = new KindOrderTbl();
    OrderDeitalsTbl oDtbl = new OrderDeitalsTbl();
    WorkersTbl wtbl = new WorkersTbl();

    public OrdersAddUpdate()
    {
        InitializeComponent();
        status = "Add";
    }

    public OrdersAddUpdate(int code)
    {
        InitializeComponent();
        status = "UpDate";
        DataRow dro = otbl.Find("O_kod", code);
        o = new Orders(dro);
        dt = oDtbl.GetProductOfOrder(o.FO_Kod);
        FillFrom();
    }

    public void FillFrom()
    {
        //שינוי שמות לפקדים
        btnAddOrder.Text = "עדכון הזמנה";
        this.Text = "עדכון הזמנה";
        //מילוי פקדים של הזמנה
        txtbKodOrder.Text = o.FO_Kod.ToString();
        dateTimePickerMake.Value = DateTime.Parse(o.FO_dateOfOrder);
        dateTimePickerTarget.Value = DateTime.Parse(o.FO_dateOfEnd);
        txtbNumHoursWorking.Text = o.FO_numWorkHour;
        txtbNumMittings.Text = o.FO_numOfMittings;
        txtbSumToPay.Text = o.FO_amountToPay;
        if (o.FO_status == "בוצע")
            rbStatusOrderFinish.Checked = true;
        else
            rbStatusOrderIn.Checked = true;
        //הצגת מוצרים בהזמנה
        for (int i = 0; i < dt.Rows.Count; i++)
        {

```



```

OrderDeitals.OrderDeitals od = new
OrderDeitals.OrderDeitals(dt.Rows[i]);
DataRow dtp = ktbl.Find("K_kod", od.FD0_KindOredrKod);
KindOrder.KindOrder p = new KindOrder.KindOrder(dtp);
flowLayoutPanel1.Controls.Add(new UserKatalog(od.FD0_kod));
}
}
private void OrdersAddUpdate_Load(object sender, EventArgs e)
{
CenterToScreen();
if (status == "Add")
{
fillCmbName();

//תאריך שליחה בעוד 7 ימים
dateTimePickerMake.MinDate = DateTime.Now;
dateTimePickerTarget.MinDate = dateTimePickerMake.Value.AddDays(7);
txtbKodOrder.Text = otbl.GetNext("O_Kod").ToString();
this.Text = "הוספת הזמנה";
rbStatusOrderFinish.Visible = false;
rbStatusOrderIn.Visible = false;
lblStatusOrder.Visible = false;
}
else
{
this.Text = "עדכון הזמנה";
DataRow drC = ctbl.Find("C_kod", o.FO_clientKod);
Clients.Clients c = new Clients.Clients(drC);
cmbName.Text = c.FC_lastName;
}
txtbId.Enabled = false;
txtbPhone.Enabled = false;

DataTable dt = Dal.GetDisplayTable("SELECT katalog.K_kod,
katalog.K_describing, katalog.K_price, katalog.k_timeWorking,
katalog.K_picture FROM katalog;");
DataTable dtOrderDeitals = Dal.GetTable("OrderDetials");
bool d = true;
for (int i = 0; i < dt.Rows.Count && d; i++)
{
//-----
//if (dt.Rows[i]["K_kod"].ToString() == "555" ||
dt.Rows[i]["K_kod"].ToString() == "777")
//    d = false;
//else
//{
UserKatalog u = new UserKatalog(int.Parse(dt.Rows[i][0].ToString()),
this);
//במצב עדכון, לסמן מה הזמין
if (status != "Add")
{
for(int k = 0; k < dtOrderDeitals.Rows.Count; k++)
{
if (dtOrderDeitals.Rows[k][1].ToString() == txtbKodOrder.Text &&
dtOrderDeitals.Rows[k][2].ToString() == u.GetCode().ToString())
u.ChangeCheckedcheckBox1();
}
}
}
flowLayoutPanel1.Controls.Add(u);

```

```
//}
}
groupBox2.Visible = false;
btnTimeMitting.Visible = false;

}
public bool CheckLegal()
{
    errorProviderDateOfEnd.Clear();
    errorProviderKodOredr.Clear();
    errorProviderMittings.Clear();
    errorProviderNumWorking.Clear();
    errorProviderStatus.Clear();
    errorProviderSum.Clear();
    bool degel = true;
    o.FO_Kod = int.Parse(txtbKodOrder.Text);
    o.FO_clientKod = int.Parse(cmbName.SelectedValue.ToString());
    //סטטוס
    if (rbStatusOrderIn.Checked)
    o.FO_status = "בוצע";
    else
    o.FO_status = "בשלב עבודה";
    //מס שעות עבודה נוספות
    if (txtbNumHoursWorking.Text != "")
    {
        try
        {
            o.FO_numWorkHour = txtbNumHoursWorking.Text;
        }
        catch (Exception ex1)
        {
            errorProviderNumWorking.SetError(txtbNumHoursWorking, ex1.Message);
            degel = false;
        }
    }
    else
    {
        txtbNumHoursWorking.BackColor = Color.LightSalmon;
        degel = false;
    }
    //מס פגישות
    if (txtbNumMittings.Text != "")
    {
        try
        {
            o.FO_numOfMittings = txtbNumMittings.Text;
        }
        catch (Exception ex1)
        {
            errorProviderMittings.SetError(txtbNumMittings, ex1.Message);
            degel = false;
        }
    }
    else
    {
        txtbNumMittings.BackColor = Color.LightSalmon;
        degel = false;
    }
    return degel;
}
```

```

}
private void btnAddOrder_Click(object sender, EventArgs e)
{
    // int mone = 0;
    // DateTime d1 = dateTimePickerTarget.Value;
    // DateTime d2 = dateTimePickerMake.Value;
    //while (d1<=d2)
    // {
    //     mone++;
    //     d1 = d1.AddDays(1);
    // }

    if (groupBox2.Visible == false)
    if (MessageBox.Show("האם התקיימה פגישת הכרות לפני ביצוע ההזמנה?", "שים לב",
    MessageBoxButtons.YesNo,
    MessageBoxIcon.Question, MessageBoxDefaultButton.Button2,
    MessageBoxOptions.RtlReading) == DialogResult.Yes)
    {
        o.FO_numOfMittings = o.FO_numOfMittings + 1;
        groupBox2.Visible = true;
        btnTimeMitting.Visible = true;
        cmbHour.SelectedIndex = 0;
        cmbMoment.SelectedIndex = 0;
        btnAddOrder.Enabled = false;
    }
    else
    SaveOrder();
    else
    SaveOrder();
    }
    private void SaveOrder()
    {
        o.FO_Kod = int.Parse(txtbKodOrder.Text);
        o.FO_clientKod = int.Parse(cmbName.SelectedValue.ToString());
        //סטטוס
        if (rbStatusOrderIn.Checked)
        o.FO_status = "בוצע";
        else
        o.FO_status = "בשלבי עבודה";
        o.FO_dateOfOrder = dateTimePickerMake.Text;
        o.FO_dateOfEnd = dateTimePickerTarget.Text;
        o.FO_numWorkHour = txtbNumHoursWorking.Text;
        o.FO_amountToPay = txtbSumToPay.Text;
        o.FO_numOfMittings = txtbNumMittings.Text;
        //להוסיף פקד לגרפיקאית...
        o.FO_GraphicArtistKod = '1'.ToString();

        int[] arr;
        if (status == "Add")
        {
            otbl.AddOrder(o);
            arr = new int[1];
        }
        else
        {
            otbl.UpDateOrder(o);
            arr = new int[dt.Rows.Count];
        }
        bool d = true;
        //הוספת הזמנה
    }

```

```
foreach (UserKatalog u in flowLayoutPanel1.Controls)
{
    if (u.checkBox1.Checked)
    {
        OrderDeitals.OrderDeitals od = new OrderDeitals.OrderDeitals();
        //הוספת מוצר בהזמנה
        if (status == "Add")
        od.FD0_kod = oDtbl.GetNext("DO_kod");
        else
        {
            for (int j = 0; j < dt.Rows.Count; j++)
            {
                if (dt.Rows[j][2].ToString() == u.GetCode().ToString())
                {
                    od.FD0_kod = int.Parse(dt.Rows[j][0].ToString());
                    od.dr = dt.Rows[j];
                    arr[j] = 1;
                    d = false;
                    j = dt.Rows.Count;
                }
            }
            if (d)
            od.FD0_kod = oDtbl.GetNext("DO_kod");
        }
        od.FD0_kod0redr = o.FO_Kod;
        od.FD0_Kind0redrKod = u.GetCode();
        od.FOD_status = "בשלבי עבודה";
        if (status == "Add" || d)
        oDtbl.AddOrderDeitals(od);
        else
        oDtbl.UpDateOrderDeitals(od);
    }
}
//////זימון טופס תשלום
//PaymentsAddUpdate p = new PaymentsAddUpdate(o.FO_Kod, "Add");
//p.ShowDialog();

//DataTable dt1 = new DataTable("Workers");
//for(int i = 0; i < dt1.Rows.Count; i++)
//{
//}

if (status == "Add")
{
    otbl.UpDateOrder(o);
}
else
{
    otbl.UpDateOrder(o);
}
MessageBox.Show("ההזמנה נשמרה בהצלחה");
this.Close();
}
private void btnTimeMitting_Click(object sender, EventArgs e)
{
    DataTable dt = mtbl.getalltableDateHour((cmbHour.Text + ":" +
    cmbMoment.Text).ToString(), dateTimePickerMitting.Text);
    if (dt.Rows.Count > 0)
```

```
{
DataRow dr = dt.Rows[0];
Mittings.Mittings m = new Mittings.Mittings(dr, "Shalom");
m.FM_clientKod = int.Parse(cmbName.SelectedValue.ToString());
mtbl.UpdateMitting(m);
btnAddOrder.Enabled = true;
MessageBox.Show("כעת יש לשמור את ההזמנה. \r\n עדכון זמן הפגישה בוצע בהצלחה.");
}
else
MessageBox.Show("בדוק שהנתונים שהזנת נכונים. \r\n במערכת לא קיימת פגישה בזמן זה");
}
public void fillCmbName()
{
cmbName.SelectedIndexChanged -= cmbID_SelectedIndexChanged;
cmbName.DataSource = ctbl.getalltable();
cmbName.DisplayMember = "שם";
cmbName.ValueMember = "קוד לקוח";
cmbName.SelectedIndexChanged += cmbID_SelectedIndexChanged;
cmbName.SelectedIndex = 1;
}

private void cmbID_SelectedIndexChanged(object sender, EventArgs e)
{
//נתונים נוספים של הלקוח, משתנה כאשר יש שינוי בתיבת בחירה
DataRow drc = ctbl.Find("C_kod", cmbName.SelectedValue);
Clients.Clients c = new Clients.Clients(drc);
txtbId.Text = c.FC_id;
txtbPhone.Text = c.FC_pel;

}
//-----
public void SumAndTime()
{
int sumToPay = 0;
double timeWorking = 0;
KindOrder.KindOrder k = new KindOrder.KindOrder();
DataRow drM = ktbl.Find("K_kod", 555);
DataRow drT = ktbl.Find("K_kod", 777);
KindOrder.KindOrder KNumHoursWorking = new KindOrder.KindOrder(drT);
KindOrder.KindOrder KNumMittings = new KindOrder.KindOrder(drM);
foreach (UserKatalog u in flowLayoutPanel1.Controls)
{
if (u.checkBox1.Checked == true)
{
sumToPay += int.Parse(u.lblPrice.Text.ToString());
timeWorking += double.Parse(u.txtbTimeWorking.Text.ToString());
}
}
txtbSumToPay.Text = sumToPay.ToString();
//חישוב סכום לתשלום
txtbSumToPay.Text = ((int.Parse(KNumHoursWorking.FK_price) *
int.Parse(txtbNumHoursWorking.Text)) +
(int.Parse(KNumMittings.FK_price) * int.Parse(txtbNumMittings.Text)) +
int.Parse(txtbSumToPay.Text)).ToString();
lblTextSumTimeWorking.Text = timeWorking.ToString();
}

private void btnAddClient_Click(object sender, EventArgs e)
{
ClientsAddUpdate c1 = new ClientsAddUpdate();
}
```

```

c1.ShowDialog();
cmbName.SelectedIndexChanged -= cmbID_SelectedIndexChanged;
cmbName.DataSource = ctbl.getalltable();
cmbName.DisplayMember = "שם משפחה";
cmbName.ValueMember = "קוד לקוח";
cmbName.SelectedIndexChanged += cmbID_SelectedIndexChanged;
cmbName.SelectedIndex = 1;
DataRow DR = ctbl.Find("C_kod", ctbl.GetNext("C_kod") - 1);
cmbName.Text = DR["C_lastName"].ToString();
}

private void dateTimePickerMake_ValueChanged(object sender, EventArgs e)
{
    dateTimePickerTarget.Value = dateTimePickerMake.Value.AddDays(7);
    dateTimePickerTarget.MinDate = dateTimePickerMake.Value.AddDays(7);
}

public partial class OrdersShow : Designer
{
    OrdersTbl otbl;
    DataTable odt;
    public OrdersShow()
    {
        InitializeComponent();
    }

    private void OrdersShow_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        this.Text = "הזמנות";
        otbl = new OrdersTbl();
        DGOrders.DataSource = otbl.getalltable();
        odt = otbl.getalltable();
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        OrdersAddUpdate cc = new OrdersAddUpdate();
        cc.ShowDialog();
        DGOrders.DataSource = otbl.getalltable();
    }

    private void btnFresh_Click(object sender, EventArgs e)
    {
        DGOrders.DataSource = odt;
        txtSearch.Text = "הקש ערך לחיפוש";
    }

    private void txtSearch_TextChanged(object sender, EventArgs e)
    {
        if (txtSearch.Text == "")
            DGOrders.DataSource = odt;
        else
            DGOrders.DataSource = Classes.SmartSreach.Search(odt, txtSearch.Text);
    }

    private void txtSearch_MouseLeave(object sender, EventArgs e)

```

```
{
//שאר העכבר יוצא מתיבת טקסט
if (txtSearch.Text == "")
{
txtSearch.Text = "הקש ערך לחיפוש";
DGOrders.DataSource = odt;
}
else
if (txtSearch.Text != "הקש ערך לחיפוש")
{
string s = txtSearch.Text;
txtSearch.Text = "הקש ערך לחיפוש";
DGOrders.DataSource = Classes.SmartSreach.Search(odt, s);
}
}

private void txtSearch_MouseClick(object sender, MouseEventArgs e)
{
txtSearch.Text = "";
}

private void ToolStripMenuItem_Click(object sender, EventArgs e)
{
if (DGOrders.SelectedRows.Count > 0)
{
OrdersAddUpdate m = new
OrdersAddUpdate(int.Parse(DGOrders.SelectedRows[0].Cells[0].Value.ToString()));
m.ShowDialog();
DGOrders.DataSource = otbl.getalltable();
}
else
{
MessageBox.Show("עליך לבחור שורה");
}
}
}
```

טופס הוספה ועדכון תשלומים+ הצגה:

```
public partial class PaymentsAddUpdate : Designer
{
string status;
int orderKodd;
Payments p;
PaymentsTbl ptbl = new PaymentsTbl();
public PaymentsAddUpdate(int orderKod, string orderStatus)
{
InitializeComponent();
status = orderStatus;
rbCreditCard.Checked = true;
orderKodd= orderKod;
//הוספה
if(status == "Add")
{
p = new Payments();
p.FP_orderKod= orderKod;
}
//עדכון
else
{
}
```

```

DataRow dr = ptbl.Find("P_orderKod", orderKod);
p = new Payments(dr);
FillForm();
}
}
//public PaymentsAddUpdate()
//{
//    InitializeComponent();
//    status = "Add";
//    rbCreditCard.Checked = true;
//    p = new Payments();
//}
public void FillForm()
{
txtbCVC.Text = p.FP_CVC.ToString();
txtbKodPayment.Text = p.FP_kod.ToString();
txtbNumCreditCard.Text = p.FP_creditCard.ToString();

if(p.FP_paymentMethod == "כרטיס אשראי")
{
rbCreditCard.Checked = true;
cmbMonth.Text = p.FP_validity.Substring(0, 2);
cmbYear.Text = p.FP_validity.Substring(3, 4);
groupBox2.Visible=false;
groupBox1.Visible = true;

}
else
{
rbBankTransfer.Checked = true;
groupBox1.Visible = false;
groupBox2.Visible = true;

}
txtKodOrder.Text = p.FP_orderKod.ToString();
//מה יהיה בחור במספר תשלומים
cmbNumPayments.Text = p.FP_numPayments.ToString();
txtKodOrder.Enabled = false;
txtbKodPayment.Enabled = false;

//סימון סטטוס- שולם או לא שולם
if (p.FP_status == "שולם")
{
rbStatusPaymentYes.Checked = true;
}
else
{
rbStatusPaymentNo.Checked = true;
}
AddPayment.Text = "עדכון";
}
private void PaymentsAddUpdate_Load(object sender, EventArgs e)
{
CenterToScreen();
////////////////////
txtKodOrder.Text = orderKod.ToString();
if (status == "Add")
{
txtbKodPayment.Text = ptbl.GetNext("P_kod").ToString();

```



```

this.Text = "הוספת תשלום";
rbStatusPaymentYes.Visible = false;
rbStatusPaymentNo.Visible = false;
lblStatusPayment.Visible = false;

cmbNumPayments.SelectedIndex = 0;
cmbMonth.SelectedIndex = 0;
cmbYear.SelectedIndex = 0;
}
else
this.Text = "עדכון תשלום";

//העברה בנקאית- הפקדים של אשראי מוסתרים
if (rbBankTransfer.Checked)
{
    lblNumPayments.Visible = false;
    cmbNumPayments.Visible = false;
    groupBox1.Visible = false;
    groupBox2.Visible = true;
}
//אשראי- הפקדים של העברה בנקאית מוסתרים
if (rbCreditCard.Checked)
{
    lblNumPayments.Visible = true;
    cmbNumPayments.Visible = true;
    groupBox1.Visible = true;
    groupBox2.Visible = false;
}
}

public bool CheckLegal()
{
    errorProviderCVC.Clear();
    errorProviderFileAttached.Clear();
    errorProviderMonth.Clear();
    errorProviderNumCredetCard.Clear();
    errorProviderYear.Clear();
    bool d = true;
    //קוד תשלום והזמנה
    p.FP_kod = int.Parse(txtbKodPayment.Text);
    p.FP_orderKod = int.Parse(txtKodOrder.Text);

    // בדיקת קוד הזמנה
    if (txtKodOrder.Text != "")
    {
        try
        {
            p.FP_orderKod = int.Parse(txtKodOrder.Text);
        }
        catch (Exception exp1)
        {
            errorProviderKodOrder.SetError(txtKodOrder, exp1.Message);
            d = false;
        }
    }
    else
    {
        txtKodOrder.BackColor = Color.LightCoral;
    }
}

```

```
d = false;
}

//אמצעי תשלום
if (rbBankTransfer.Checked)
{
p.FP_paymentMethod = "העברה בנקאית";
//איך לבדוק אם הוסיפו קובץ באסמכתא
}
else
{
p.FP_paymentMethod = "כרטיס אשראי";
// מספר כרטיס
if (txtbNumCreditCard.Text != "" && txtbNumCreditCard.Text.Length ==
16)
{
try
{
p.FP_creditCard = txtbNumCreditCard.Text;
}
catch (Exception exp1)
{
errorProviderNumCredetCard.SetError(txtbNumCreditCard, exp1.Message);
d = false;
}
}
else
{
txtbNumCreditCard.BackColor = Color.LightSalmon;
d = false;
}
// תוקף
if (cmbYear.Text != "")
try
{
p.FP_validity = cmbMonth.Text + "/" + cmbYear.Text;
}
catch (Exception ex1)
{
errorProviderValidity.SetError(cmbYear, ex1.Message);
d = false;
}
else
{
cmbYear.BackColor = Color.LightSalmon;
d = false;
}
// ספרות בגב הכרטיס 3
if (txtbCVC.Text != "" && txtbCVC.Text.Length == 3)
{
try
{
p.FP_CVC = txtbCVC.Text;
}
catch (Exception exp1)
{
errorProviderCVC.SetError(txtbCVC, exp1.Message);
d = false;
}
}
```

```

    }
    else
    {
        txtbCVC.BackColor = Color.LightSalmon;
        d = false;
    }
    // מספר תשלומים
    p.FP_numPayments = cmbNumPayments.Text;
}
// סטטוס
if (rbStatusPaymentYes.Checked)
    p.FP_status = "שולם";
else
    p.FP_status = "לא שולם";

return d;
}
private void AddPayment_Click(object sender, EventArgs e)
{
    if (CheckLegal())
    {
        if (status == "Add")
        {
            ptbl.AddPayment(p);
        }
        else
        {
            ptbl.UpDatePayment(p);
        }
        this.Close();
    }
}

private void rbBankTransfer_CheckedChanged(object sender, EventArgs e)
{
    if (rbBankTransfer.Checked)
    {
        lblNumPayments.Visible = false;
        cmbNumPayments.Visible = false;
        groupBox1.Visible = false;
        groupBox2.Visible = true;
    }
}

private void rbCreditCard_CheckedChanged(object sender, EventArgs e)
{
    if (rbCreditCard.Checked)
    {
        lblNumPayments.Visible = true;
        cmbNumPayments.Visible = true;
        groupBox1.Visible = true;
        groupBox2.Visible = false;
    }
}

private void txtbNumCreditCard_TextChanged(object sender, EventArgs e)
{
    if (txtbNumCreditCard.BackColor == Color.LightSalmon)

```

```

txtbNumCreditCard.BackColor = Color.White;
}

private void txtbCVC_TextChanged(object sender, EventArgs e)
{
    if (txtbCVC.BackColor == Color.LightSalmon)
        txtbCVC.BackColor = Color.White;
}

public partial class PaymentsShow : Designer
{
    PaymentsTbl ptbl;
    public PaymentsShow()
    {
        InitializeComponent();
    }

    private void PaymentsShow_Load(object sender, EventArgs e)
    {
        CenterToScreen();
        this.Text = "תשלומים";
        ptbl = new PaymentsTbl();
        DGPayments.DataSource = ptbl.getalltable();
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        PaymentsAddUpdate cc = new PaymentsAddUpdate(85, "Add");
        cc.ShowDialog();
        DGPayments.DataSource = ptbl.getalltable();
    }

    private void ToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (DGPayments.SelectedRows.Count > 0)
        {
            PaymentsAddUpdate pa = new
            PaymentsAddUpdate(int.Parse(DGPayments.SelectedRows[0].Cells[1].Value.ToString()), "Update");
            pa.ShowDialog();
            DGPayments.DataSource = ptbl.getalltable();
        }
        else
        {
            MessageBox.Show("עליך לבחור שורה");
        }
    }

    public partial class ViewSchedule : Designer
    {
        string[] dates = new string[5];
        public int kodOrder;
        DateTime date;
        int kodC;
        string status;
        string kindTor;
        int misT = 0;
    }
}

```

סופס יומן:

```

Orders.OrdersAddUpdate addS;
public ViewSchedule(string kod0, int kodC, string kindTor,
Orders.OrdersAddUpdate addS, int misT)
{
InitializeComponent();
this.kodOrder = int.Parse(kod0);
this.kodC = kodC;
this.kindTor = kindTor;
this.addS = addS;
this.misT = misT;

}
public ViewSchedule(string kod0, int kodC, string kindTor)
{
InitializeComponent();
this.kodOrder = int.Parse(kod0);
this.kodC = kodC;
this.kindTor = kindTor;
}
public ViewSchedule()
{
InitializeComponent();
tableLayoutPanel1.Enabled = true;
}

private void AddUpdateSchdule_Load(object sender, EventArgs e)
{
dateTimePicker1.Value = DateTime.Today;
}
public void FillSchedule()
{
int i = 0;
string time = "0";
string timeLmdida = "0";

for (int k = 0; k < tableLayoutPanel1.RowCount; i++, k++)
for (int j = 0; j < tableLayoutPanel1.ColumnCount; j++)
{
//שעה
if (int.Parse(label16.Tag.ToString()) == i)
{
time = "09:00";
}
else
if (int.Parse(label17.Tag.ToString()) == i)
{
time = "09:30";
}
else
if (int.Parse(label18.Tag.ToString()) == i)
{
time = "10:00";
}
else
if (int.Parse(label19.Tag.ToString()) == i)
{
time = "10:30";
}
else

```

```

if (int.Parse(label20.Tag.ToString()) == i)
    time = "11:00";

else
if (int.Parse(label11.Tag.ToString()) == i)
{
    time = "11:30";
}
else
if (int.Parse(label12.Tag.ToString()) == i)
{
    time = "12:00";
}
else
if (int.Parse(label13.Tag.ToString()) == i)
{

time = "12:30";
}
else
if (int.Parse(label21.Tag.ToString()) == i)
{
    time = "13:00";
}
else
if (int.Parse(label23.Tag.ToString()) == i)
{
    time = "13:30";
}
else
if (int.Parse(label22.Tag.ToString()) == i)
{
    time = "14:00";
}
else
if (int.Parse(label24.Tag.ToString()) == i)
{
    time = "14:30";
}

UserMeeting u = new UserMeeting(dates[j], time, this);
tableLayoutPanel1.Controls.Add(u, j, i);
}
}
private void ViewSchedule_Load(object sender, EventArgs e)
{
    CenterToScreen();
    DateChange();
    //DataTable dt = Dal.GetDisplayTable("SELECT mittings.M_kod,
mittings.M_date, mittings.M_hour, mittings.M_clientKod,
mittings.M_status FROM mittings;");
    //for(int i = 0; i < dt.Rows.Count; i++)
    //{
    //    UserMeeting u = new UserMeeting(dateTimePicker1.Text,
label16.Text, this);
    //    tableLayoutPanel1.Controls.Add(u);
    //}
}
public void DateChange()
{

```

```

int mone = 0;
string day = dateTimePicker1.Value.DayOfWeek.ToString();
if (label1.Tag.ToString() == day)
{
    label1.Text = dateTimePicker1.Value.ToShortDateString();
    mone++;
    label2.Text =
        (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
    mone++;
    label3.Text =
        (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
    mone++;
    label4.Text =
        (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
    mone++;
    label5.Text =
        (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
    mone++;
}

else
{
    mone--;
    if (label2.Tag.ToString() == day)
    {
        label2.Text = dateTimePicker1.Value.ToShortDateString();
        label1.Text =
            (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
        mone++;
        mone++;
        label3.Text =
            (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
        mone++;
        label4.Text =
            (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
        mone++;
        label5.Text =
            (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
        mone++;
    }
    else
    {
        mone--; //-2
        if (label3.Tag.ToString() == day) //שלישי? לא
        {
            label3.Text = dateTimePicker1.Value.ToShortDateString();
            label1.Text =
                (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
            mone++;
            label2.Text =
                (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
            mone++;
            mone++;
            label4.Text =
                (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
            mone++;
            label5.Text =
                (dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
        }
    }
}

```

```

mone++;

}
else
{
mone--; //-3
if (label4.Tag.ToString() == day) //יום רביעי?
{
label4.Text = dateTimePicker1.Value.ToShortDateString(); //יום רביעי שווה להיום
label1.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString(); //יום ראשון שווה
ליום ראשון
mone++; //-2
label2.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++; //-1
label3.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++; //0
mone++; //1
label5.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;

}
else
{
mone--;
if (label5.Tag.ToString() == day)
{
label5.Text = dateTimePicker1.Value.ToShortDateString();
label1.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label2.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label3.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label4.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
mone++;

}

else
{
mone--;

label1.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label2.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label3.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
}
}

```



```

label4.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;
label5.Text =
(dateTimePicker1.Value.AddDays(mone)).ToShortDateString();
mone++;

}

}
}
}
}
dates[0] = label1.Text;
dates[1] = label2.Text;
dates[2] = label3.Text;
dates[3] = label4.Text;
dates[4] = label5.Text;
tableLayoutPanel1.Controls.Clear();
FillSchedule();

}
private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    DateChange();
}

public void End()
{
    if (addS != null)
    {
        misT -= 1;
        //addS.MinusTorim();
    }

    MessageBox.Show("קביעת הפגישה בוצעה בהצלחה");
    this.Close();
}
}

```

המחלקה UserKatalog:

```
public partial class UserKatalog : UserControl
{
    KindOrderTbl ktbl = new KindOrderTbl();
    KindOrder.KindOrder k;
    OrdersAddUpdate orderAU;
    int codeOrder;
    string describing;
    string price;

    public UserKatalog(int codeKatalog, OrdersAddUpdate o)
    {
        InitializeComponent();
        this.orderAU = o;
        k = new KindOrder.KindOrder(ktbl.Find("K_kod", codeKatalog));
        FillUser();
    }
    public UserKatalog(int code)
    {
        InitializeComponent();
        ////////// כאן
    }
    public void FillUser()
    {
        lblKodCatalog.Text = k.FK_kod.ToString();
        textBox1.Text = k.FK_describing;
        lblPrice.Text = k.FK_price;
        txtbTimeWorking.Text = k.Fk_timeWorking;
    }
    private void btnPicture_Click(object sender, EventArgs e)
    {
        KatalogPicture kp = new KatalogPicture(k.FK_kod);
        kp.ShowDialog();
    }

    private void UserKatalog_Load(object sender, EventArgs e)
    {
    }
    public int GetCode()
    {
        return int.Parse(lblKodCatalog.Text);
    }
    public bool GetCheckedCheckBox2()
    {
        return checkBox2.Checked;
    }
    public void ChangeCheckedcheckBox1()
    {
        checkBox1.Checked = true;
    }
    public void ChangeCheckedcheckBox2()
    {
        checkBox2.Checked = true;
    }
    private void checkBox1_CheckedChanged(object sender, EventArgs e)
    {
        orderAU.SumAndTime();
    }
}
```

```
}  
}
```

המחלקה :UserMeeting

```
public partial class UserMeeting : UserControl
{
    string date;
    string hour;
    ViewSchedule sch;
    DataTable dt;
    MittingsTbl mtbl=new MittingsTbl ();

    int codeClient;
    Clients.Clients c;
    int kodOrder;
    Orders.OrdersTbl oTbl = new Orders.OrdersTbl();
    Clients.ClientsTbl cTbl = new Clients.ClientsTbl();
    Orders.Orders o= new Orders.Orders();
    public UserMeeting()
    {
        InitializeComponent();
    }
    public UserMeeting(string date, string hour, ViewSchedule sch)
    {
        InitializeComponent();
        this.date = date;
        this.hour = hour;
        this.sch = sch;
    }
    private void UserMeeting_Load(object sender, EventArgs e)
    {
        this.Enabled = true;
        //שאלתה שמביאה את הפגישה לפי זמן ותאריך
        dt = mtbl.getalltableDateHour(hour, date);

        if (dt.Rows.Count > 0)
        {
            DataRow dr1 = dt.Rows[0];
            Mittings.Mittings m = new Mittings.Mittings(dr1, "shalom:");
            //מילוי פקדים
            linkLabel2.Visible = false;

            //DataRow dr = oTbl.Find("O_Kod", kodOrder);
            //o = new Orders.Orders(dr);
            //kodOrder = int.Parse(dr["O_Kod"].ToString());
            DataRow drc = cTbl.Find("C_kod", m.FM_clientKod);
            c = new Clients.Clients(drc);
            labelName.Text = drc["C_lastName"].ToString();
            btnMittingDeitals.Visible = true;
            label1.Visible = true;
            labelName.Visible = true;
        }
    }

    private void linkLabel2_LinkClicked(object sender,
    LinkLabelLinkClickedEventArgs e)
    {
        MittingsAddUpdate c4 = new MittingsAddUpdate(date, hour);
```

```
c4.ShowDialog();  
sch.DateChange();  
}
```

```
private void btnMittingDeitals_Click(object sender, EventArgs e)  
{  
    MittingsAddUpdate m1 = new MittingsAddUpdate(date, hour);  
    m1.ShowDialog();  
  
}  
}
```