

# פרויקט גמר להנדסאי תוכנה



**MY-KAV**

*משתלם בדרך שלי*

**המגמה לתכנה**

מוגש ע"י:

פנחסי רחל

וקוליץ טובה

בהנחיית: הגב' מרים שמעונוביץ

כ"א שבט תשפ"א / פברואר 2021

## תודות

ראשית לכל הננו רוצות להודות לריבונו של עולם שהביאנו עד הלום בידו הרחומה והאוהבת. אנו אסירות תודה לכמות כה רבה של אנשים שלוונו בסבלנות, מאור פנים ואוזן קשבת:

לרכזת המגמה גב' לאה יעקובוביץ שמנהלת את המגמה ביד רמה, במסירות ובאחריות מרובה שנים כה רבות. לכל צוות המורות במגמה על הכשרתנו המקצועית ועל סבלנותן הרבה, וכמובן למנחה היקרה גב' מרים שמעונוביץ שלימדה אותנו את עקרונות השפה והטכנולוגיה, על שליוותה אותנו בסבלנות מרובה, ונתנה לנו הרגשה נעימה, עודדה, תמכה והייתה תמיד לצדינו. תודה.

וכן תודה ענקית שלוחה להורינו ולמשפחותינו שליוו אותנו במסירות ונתנו לנו עידוד רב. וכן לכל החברות במגמה ששאלו, התעניינו ועודדו... שה' ישלם שכרכם!



**MY-KAV**

משתלם בדרך שלי

## תוכן עניינים

II .....	תודות
IV .....	הצעה לפרויקט גמר
XIII .....	אישור הצעת הפרויקט ממה"ט
1 .....	הצהרה
2 .....	מבוא
3 .....	מדריך למתכנת:
3 .....	אסטרטגיות טכנולוגיות:
5 .....	תיאור מבנה הפרויקט:
6 .....	עקרונות התכנון
9 .....	תרשימים:
12 .....	מבנה נתונים מאוכסנים:
22 .....	מדריך למשתמש:
22 .....	הוראות כלליות לשימוש באתר:
23 .....	צילומי מסכים:
31 .....	סיכום ומסקנות:
32 .....	נספחים:
33 .....	ביבליוגרפיה:



**MY-KAV**

משתלם בדרך שלי

# הצעה לפרויקט גמר



תאריך: 12/06/2019

לכבוד יחידת הפרויקטים מה"ט

## הצעה לפרויקט גמר

### א. פרטי הסטודנטים

שם הסטודנט	ת.ז.	כתובת	טלפון נייד	שנת סיום הלימודים
טובה קולין	207088535	יהודה הנשיא 47 בני ברק	0548590889	
רחל פנחסי	207088162	שבטי ישראל 22 בני ברק	0583284840	

שם המכללה: בית המורה

סמל המכללה: 76076

מסלול ההכשרה: הנדסאים

מגמת לימוד: תכנות מחשבים

מקום ביצוע הפרויקט: בסמינר ובבית

### ב. פרטי המנחה האישי

שם המנחה *	כתובת	טלפון נייד	תואר	מקום עבודה/תפקיד
מרים שימונוביץ	חזון איש 11 ב"ב	052-7171295	B.ED	סמינרים
			הנדסאי מחשבים	המכללה למנהל

חתימת הסטודנט

חתימת המנחה האישי

חתימת הגורם המקצועי מטעם מה"ט

ר. פנחסי

ט. קולין

## שם הפרויקט: My Kav

### רקע:

#### 2.1 תיאור ורקע כללי:

האתר My Kav מאפשרת תשלום חכם וחסכוני על הנסיעות בתחבורה הציבורית בישראל. כאשר חישוב המחיר הסופי לתשלום יחושב לאחר תקופה קבועה מראש (לרוב חודש). וכך ימליץ האתר על החוזה/החוזים החסכוניים ביותר לנסיעה.

#### 2.2 מטרות המערכת

- לחשב למשתמש איזה חוזה/חוזים הכי חסכוניים לו על פי הנסיעות שנסע.
- לאפשר למשתמש חישוב מדויק של כל נסיעותיו.
- לאפשר למשתמש לצפות בנסיעות שנסע בעבר, במדויק.

### סקירת מצב קיים

התשלום מתבצע לפני הנסיעות בפועל ואין לנוסע אפשרות לדעת מה החוזה הכי משתלם בעבורו. ולכן יכול להיווצר מצב שנוסע ירכוש חוזה במחשבה שזה הכי חסכוני בשבילו ולאחר מעשה יתברר כי ישנם חוזים משתלמים יותר לנסיעות שנסע בפועל.

לדוגמא נוסע שרכש חוזה חופשי יומי במטרה לנסוע נסיעות רבות באותו יום ולבסוף נסע רק מספר מועט של נסיעות שאם היה משלם בערך צבור על אותם נסיעות היה יוצא לו יותר זול מחופשי יומי.

### מה הפרויקט אמור לחדש או לשפר?

האתר מחשב לו לאחר תקופה קבועה מראש את המחיר המשתלם ביותר בשבילו לאחר שכבר נסע את הנסיעות ואין צורך לשער מראש את החוזה המשתלם

### דרישות מערכת ופונקציונאליות

#### 1.1. דרישות מערכת, סביבת הטמעה ושימוש.

- המערכת תעבור קומפילציה והפצה בסביבת visual studio עם התקנת .net framework גרסה 4.0 ומעלה והיא אמורה לרוץ בסביבת שרת אשר מריץ IIS express לקבלת בקשות לתצוגת דפי אינטרנט.
- המשתמש יוכל להריץ את האתר בכל מכשיר אשר מותקן עליו דפדפן אינטרנט.
- הפרויקט יכתב בסביבת VisualStudio בטכנולוגיית web API והנתונים ישמרו ב-SQL.
- קוד הקליינט יהיה כתוב על פי תקן הכתיבה שמקובל ב-Angular7 בסביבת VSCode.
- המידע המוצג למשתמש יהיה מעודכן על פי הנתונים בשרת הנתונים. שרת הנתונים יתעדכן מיד בסיום עדכון המידע על ידי המשתמש, ובדיקת נכונותו על ידי המערכת.
- זמן התגובה של המערכת יהיה קצר מאוד עבור כל פעולת משתמש, גם עבור פעולות חישוביות מורכבות יותר.

#### 1.2. שרידות, ביצועים / התמודדות עם עומסים:

צד השרת מריץ IIS express המסוגל להתמודד עם מספר רב של קריאות בו זמנית.

גם עומס על שרת ה-SQL אינו צפוי בסדר גודל כזה של אתר מכיוון שהוא בנוי להתמודדות בהצלחה עם עומסים כבדים בהרבה.

### 1.3. דרישות פונקציונליות

רשימת דרישות המשתמש מהמערכת :

- 5.3.1 מאפשר למשתמש חדש להירשם למערכת, הוא נדרש להכניס את פרטיו האישיים ולבחור שם משתמש וסיסמא.
- 5.3.2 האפליקציה מאפשרת למשתמש קיים לצפות בכל הנסיעות שנסע בחודש האחרון.
- 5.3.3 האפליקציה מאפשרת למשתמש קיים לצפות במחיר האפשרי הנמוך ביותר על הנסיעות שנסע בחודש האחרון.

### בעיות צפויות במהלך הפיתוח ופתרונות

#### 1.4. תיאור הבעיות הללו כפועל יוצא של דרישות המשתמש מהאפליקציה

- בעיה 1 : כניסה של לקוח חדש באמצע החודש
- בעיה 2 : לקוח שרוצה לרכוש חוזה חופשי חודשי גמיש
- בעיה 3 : לקוח שרוצה לרכוש חוזה חופשי שנתי

#### 1.5. פתרונות אפשריים :

- לבעיה 1 : פתרון 1 : החודש הראשון של התשלום יהיה חודש קצר והתשלום יתבצע כרגיל בסוף החודש פתרון 2 : תהיה אפשרות להירשם למערכת רק בתחילת החודש פתרון 3 : הגביה תתבצע חודש לאחר הכניסה למערכת וכך לכל לקוח יהיה תאריך גביה שונה
- לבעיה 2 : פתרון 1 : הלקוח יצטרך לבקש הארכה על זמן גבית התשלום וישלם לאחר חודשיים פתרון 2 : לא תהיה אפשרות לרכוש את החוזה הנ"ל.
- לבעיה 3 : פתרון 1 : במידה והמערכת תראה שבמידה והלקוח ימשיך לנסוע כמספר הנסיעות שנסע בחודש האחרון ישתלם לו לרכוש חוזה חופשי שנתי המערכת תציע לו לרכוש חוזה זה עם פריסת התשלום לשנים עשר חודשים (החל מהחודש הנוכחי) בהתחייבות הלקוח לסיום החוזה. פתרון 2 : לא תהיה אפשרות לרכוש את החוזה הנ"ל.

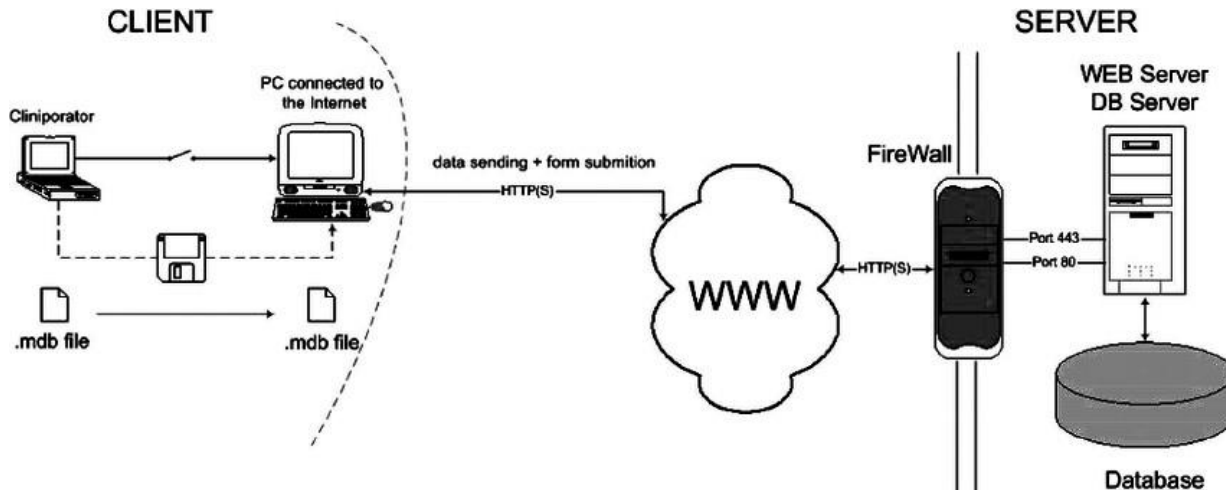
#### הפתרונות הנבחרים :

- לבעיה 1 : נשתמש בפתרון 1 כדי שתהיה אפשרות להירשם למערכת בכל תאריך ולעמוד בקריטריונים של משרד התחבורה של גביה בתאריך קבוע בסוף החודש
- לבעיה 2 : נשתמש בפתרון 1 כדי שתהיה אפשרות לרכוש את כל החוזים הקיימים.
- בעיה 3 : נשתמש בפתרון 1 כדי שתהיה אפשרות לרכוש את כל החוזים הקיימים.

## פתרון טכנולוגי נבחר :

### טופולוגית הפתרון

המערכת מורכבת משרת IIS המריץ את האפליקציה בסביבת ה- server, מסד נתונים DB's של sql-server



### טכנולוגיות בשימוש.

Angular7 – הפרויקט ירוץ בדפדפן וטכנולוגית אנגולר תאפשר לנו שילוב של scripts וכן תצוגת HTML.

### שפות הפיתוח

- שפות הפיתוח בצד השרת :

- (1) **C#.NET** - שפה זו היא שפת תכנות פופולארית שמיועדת לפיתוח כללי של מגוון אפליקציות בכל התחומים. התחביר והעקרונות שלה פשוטים מצד אחד, אך עשירים ביכולות מצד שני.
- (2) **SQL** - היא שפת מחשב הצהרתית לטיפול ועיבוד מידע בבסיסי נתונים יחסיים, השפה מאפשרת שליפת נתונים ועדכונם ויצירת טבלה ושינויה. היא השפה הנפוצה ביותר לתשאול בסיסי נתונים יחסיים.

- שפות הפיתוח בצד הלקוח :

- (1) **Typescript** - היא שפת תכנות חנימית ומבוססת קוד פתוח המפותחת ומתוחזקת על ידי מיקרוסופט. היא מכילה את קבוצת כל פקודות ותחביר JavaScript הפופולרית ומוסיפה עליה טיפוסים סטטיים ותכנות מונחה עצמים מבוסס מחלקות.
- (2) **HTML** - שפת תגיות לתצוגה ועיצוב דפי אינטרנט ותוכן לתצוגה בדפדפן. זו שפת התגיות המרכזית בעולם האינטרנט, המהווה שלד למרבית עמודי התוכן באינטרנט. השפה מאפשרת עיצוב תוכן בצורה מהירה, HTML תוכננה לעבוד על כל מחשב, מכל סוג והיא סלחנית מאד לגבי פרטים קטנים.
- (3) **CSS** - גיליונות סגנון מדורגים הם פורמט לעיצוב דפי אינטרנט. הגיליונות קובעים את

עיצובם של תגים ב-HTML, XHTML וכל שפה דומה ל-XML לבניית אתרי אינטרנט.  
CSS נוצר במטרה להפריד בין תוכן ומבנה דפי האינטרנט לבין עיצובם.

## ארכיטקטורה נבחרת

הארכיטקטורה הנבחרת היא חלוקה לשלוש שכבות. Tier Architecture-3. בפרויקטים שמבוססים על DB מומלץ להשתמש בגישה של מודל השכבות, בארכיטקטורה זו קיימת הפרדה בין השכבות השונות בפרויקט. היתרון העיקרי בחלוקה לשכבות הוא תחזוקה קלה, במיוחד במערכות גדולות כך שניתן יחסית בקלות להחליף של שכבה בלי לגעת בשכבות האחרות כאשר נרצה לעשות שינויים בתכנית חלוקה לתכניות ומודולים.

1. שכבת תצוגה (presentation layer) - שכבת ממשק המשתמש (UI-User Interface).  
שכבת התצוגה מדפי HTML באתר אינטרנט.  
השכבה מתקשרת בין המשתמש לבין שאר השכבות של המערכת.
2. שכבת הלוגיקה העסקית (BL-Business logic) - השכבה שאמונה על הלוגיקה של המערכת, עוסקת בעיבוד המידע חישובים שונים ושליחתו לשכבת התצוגה.
3. שכבת הנתונים (DAL- data access layer) - שכבה זו מורכבת ממקור נתונים – מסד הנתונים שלנו, וממערכת תוכנה entity framework אשר תפקידה לקרוא את המידע הנדרש למערכת לשמור את העדכונים ולהוסיף מידע חדש או למחוק פרטי מידע קיימים.

## סביבת השרת

לצורך הפרויקט נשתמש בשרת מקומי, express IIS המסופק עם סביבת העבודה של visual studio. אם האפליקציה תרכש על ידי לקוח נעלה אותה לשרת ארוח כל שהוא או Microsoft azure ממשק המשתמש/לקוח – GUI  
שכבת הGUI מורכבת מדפי ה html שמוצגים למשתמש דרך הדפדפן.

ממשקים למערכות אחרות / API :  
לא רלוונטי.

שימוש בחבילות תוכנה.

Bootstrap, CSS, Entity Framework

## 2. שימוש במבני נתונים וארגון קבצים

### מבני הנתונים

משתמש- תעודת זהות, האם מנהל, מספר רב קו, סיסמא, שם פרטי, שם משפחה, קוד פרופיל.  
נסיעות- קוד נסיעה, תאריך ושעה, תעודת זהות משתמש, קוד כלי תחבורה.  
כרטיס- קוד כרטיס, תאור כרטיס, חופשי חודשי, חופשי שבועי, חופשי יומי, נסיעה בודדת.  
כלי רכב- קוד כלי רכב, מספר רכב, קוד כרטיס.  
פרופילים- קוד פרופיל, שם פרופיל, אחוז הנחה.

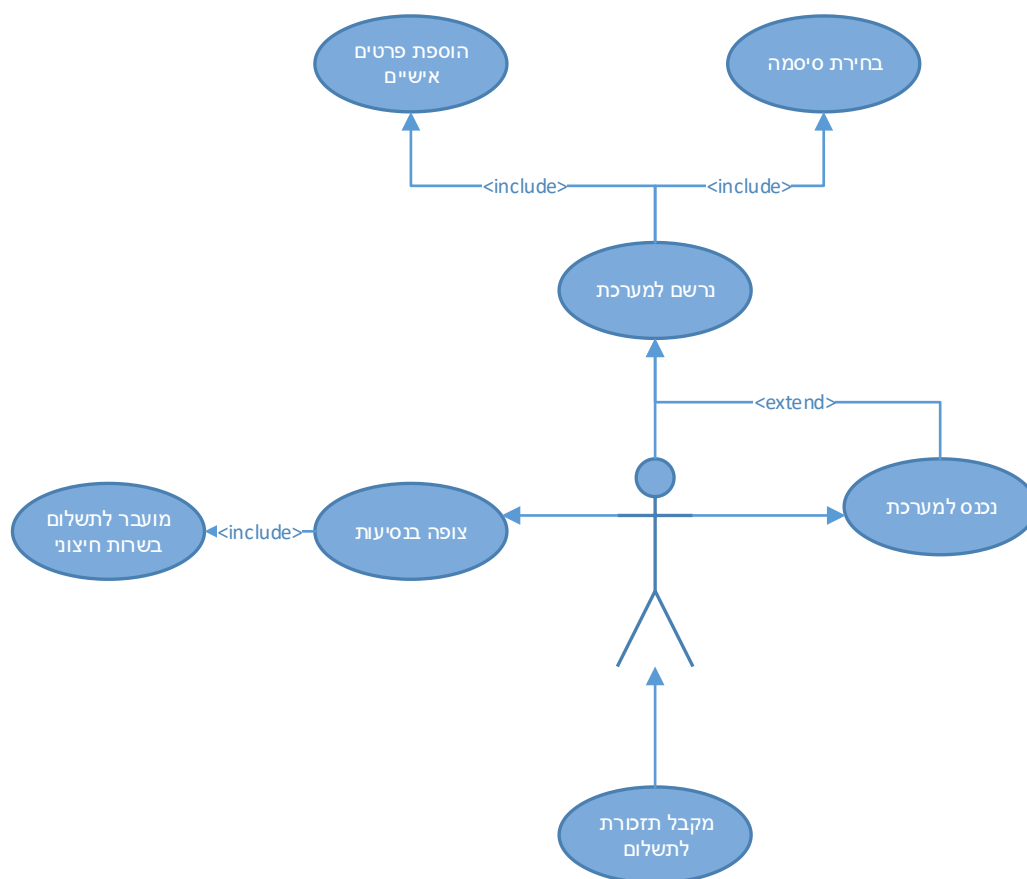


## שיטת האחסון

שיטת האחסון היא מסד נתונים. הגישה למסד הנתונים היא ע"י Entities. sql server בנוי לתמיכה במס' גדול של קריאות בו זמניות ואין חשש לקריסה ברמה של הפרויקט.

3. תרשימי מערכת מרכזיים

## Use Case

תיאור המרכיב האלגוריתמי – חישובי.

איזה בעיה בא לפתור, איך יפתור?

בעיה: כשמשמש נכנס למערכת צריך לחשב עבורו את הסכום לתשלום הזול ביותר.

פתרון: בעזרת חישובים מתאימים יחושב הסכום לתשלום.

איסוף מידע וניתוחים סטטיסטיים (אנליטיקות)

ניתן לבצע סטטיסטיקה על היחס בין גיל המשתמש לכמות הנסיעות שלו, על אזור הנסיעות

הפופולארי ביותר, הגיל הפופולארי ביותר בקרב משתמשי המערכת.

תיאור/התייחסות לנושאי אבטחת מידע

יש לדאוג לאבטחת השרת מעומסים מופרזים הנגרמים יל ידי גורמים זדוניים, ואת שרת ה-sql-

server מ-sql injection



**MY-KAV**

משתלם בדרך שלי

משתמש הנכנס לאתר קוד המשתמש ישלח לשרת והוא יצטרף לכל רשומה הנשמרת עבורו לא יתכן מצב בו משתמש נכנס לאתר בתור משתמש אחר.  
נא ציין מס' מקרים ותגובות להם ניתן מענה אבטחתי.

- במקרה שמשתמש חדש מנסה להכנס כמשתמש רשום המערכת תפנה אותו לדף ההרשמה לאתר.
- במקרה שבשעת כניסת משתמש הסיסמה אינה תואמת למספר הרב קו שהקיש המערכת תציג לו הודעת שגיאה ולא תאפשר שגיאה.
- הסיסמה תהיה מוסתרת.

### משאבים הנדרשים לפרויקט:

מספר שעות המוקדש לפרויקט: 700  
חלוקת עבודה בין חברי הצוות הוא 350 שעות לכל אחת  
ציוד נדרש  
מחשב הכולל חיבור לאינטרנט, CPU I5 , RAM 8GB, HD SSD  
תוכנות נדרשות  
Visual studio, sql server, IIS express, דפדפנים  
ידע חדש שנדרש ללמוד לצורך ביצוע הפרויקט  
נרחיב את הידע שלנו ב- Angular7 A וב- C# ובשימוש ב- Web API

### ספרות ומקורות מידע

- [stackoverflow.com](https://stackoverflow.com)
- [codeproject.com](https://codeproject.com)
- [msdn.microsoft.com](https://msdn.microsoft.com)
- [getbootstrap.com](https://getbootstrap.com)
- [w3schools.com](https://w3schools.com)

### תכנית עבודה ושלבנים למימוש הפרויקט:

- ייזום הרעיון- מאי
- ניתוח מערכות –מאי
- ניתוח מבני נתונים – מאי
- אפיון UX/UI- עד סוף מאי
- כתיבת הלוגיקה העיסקית- עד תחילת פברואר
- כתיבת ממשק משתמש – עד תחילת פברואר
- עיצוב- עד סוף פברואר
- בדיקות התוכנה -עד סוף פברואר
- התקנה והטמעה -עד סוף פברואר

## 14 תכנון הבדיקות שיבוצעו

14.1 נא פרט בטבלה, בדיקות תהליכיות ברמת משתמש בהן נדרשת המערכת לעמוד (full Flow).

מספר בדיקה	מס' דרישה במסמך אפיון	מקרי הבדיקה	ידנית/ אוטומטית	חשיבות	הערות
1	5.3.1	בדיקה תקינות מס הרב קו	ידנית	גבוהה	בדיקה האם מס' הרב קו נמצא במערכת באם לא תוצג הודעת שגיאה מתאימה למשתמש או אפשרות להירשם כמשתמש חדש
2	5.3.1	בדיקת שדה תעודת זהות האם חוקי	ידנית	גבוהה	במקרה של תעודת זהות שגויה תוצג הודעה למשתמש
3	5.3.3	בדיקת מצב בקשת הארכת תשלום של משתמש	ידנית	בינונית	אם משתמש קיבל הארכת תשלום של יותר מחודשיים תשלח לו הודעה למייל שבמקרה ולא יסדיר את התשלום לא יוכל להמשיך להשתמש בשרותי האתר

## 15 בקרת גרסאות (version control)

מכיוון ואנו 2 מגישות ואנו עובדות במקביל. נחלק את העבודה. ונעבוד עם שרותיו של פקודות GIT נשתמש באתר GITHUB לשמירה וניהול הגרסאות עד לתוצר המוגמר.

חתימת המנחה

חתימת הסטודנט

חתימת הסטודנט

76

רחל סנחסי

הערות ראש המגמה במכללה:

\_\_\_\_\_

אישור ראש המגמה:

הערות הגורם המקצועי מטעם מה"ט:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# אישור הצעת הפרויקט ממה"ט

# הצהרה

**מחש** המכון הממשלתי להכשרה בטכנולוגיה ובמדע

## יחידת הפרויקטים

חוזר מנהל מה"ט 11-4-52 – נספח מס' 3

### הצהרת סטודנט

שם הסטודנט: טובה קוליץ ת.ז. 207088535

שם הסטודנט: רחל פנחסי ת.ז. 207088162

שם המכללה בה לומדים הסטודנטים: בית המורה

אני הח"מ, מצהיר בזאת כי פרויקט הגמר וספר הפרויקט המצ"ב נעשו על ידי בלבד.

פרויקט הגמר נעשה על סמך הנושאים שלמדתי במכללה ובאופן עצמאי.

פרויקט הגמר וספר הפרויקט נעשו על בסיס הנחייתו של המנחה האישי.

מקורות המידע בהם השתמשתי לביצוע פרויקט הגמר מצוינים ברשימת המקורות המצוינים בספר הפרויקט.

אני מודע לאחריות שהנני מקבל על עצמי על ידי חתימתי על הצהרה זו ישראל היאמיר רה אמת ורק אמת.

חתימת הסטודנט: 76 חתימת הסטודנט: רחל טובה

### אישור המנחה האישי

הריני מאשר שהפרויקט בוצע בהנחייתי, קראתי את ספר הפרויקט ומצאתי כי הוא מוכן לצורך הגשת הסטודנט להגנה על פרויקט גמר.

שם המנחה: \_\_\_\_\_ חתימה: \_\_\_\_\_ תאריך: \_\_\_\_\_

### אישור ראש המגמה

הריני מאשר שספר הפרויקט מוכן לצורך הגשת הסטודנט להגנה על פרויקט הגמר.

שם ראש המגמה: \_\_\_\_\_ חתימה: \_\_\_\_\_ תאריך: \_\_\_\_\_

## מבוא

כיום כשנוסע מטעין את כרטיס הרב-קו שלו הוא עושה זאת לפני שהוא נוסע בפועל באוטובוס, על סמך השערה מה תהינה הנסיעות שלו לתקופה הקרובה. אך במקרים רבים התחזיות משתנות ולא מבצע את הנסיעות שתכנן או מבצע יותר נסיעות ממה שתכנן, במקרים כאלה קורה שאנשים רבים משלמים סכומי כסף גדולים שהיו יכולים לחסוך אותם אם רק היו משלמים לאחר שכבר נסעו.

MY-KAV אינה רק אופציית תשלום סטנדרטית, אלא, היא דרך להנגיש אזרח את מימוש האופציות הכדאיות עבורו לתשלום בתחבורה הציבורית.

החזון שלנו היא שכל נוסע ישלם לאחר שכבר נסע את הנסיעות, והמערכת תחשב עבורו מה אופציית התשלום הכי משתלמת בעבורו. זאת על ידי בדיקה של הנסיעות שאותם נסע במהלך החודש האחרון למול החוזים המתאימים עבורו מבחינת פרופיל הרב קו שלו ואזורי הנסיעה.

המערכת תבצע עבורו את החישוב המדויק כדי להגיע למחיר המינימלי שהנוסע יצטרך לשלם ותחסוך ממנו הימור על קניית חוזי נסיעה, לפני שמבצע את הנסיעה בפועל מבלי לדעת האם זה אכן החוזה הכי משתלם בעבורו.

### אז איך זה עובד?

כל אחד יכול ליצור משתמש פרטי משלו.

כעת מתבקש המשתמש להזין את מספר הרב-קו האישי שלו, והמערכת מתחילה בתהליך החישוב, התהליך מתבסס על הלוגיקה העומדת מאחורי חלוקת הארץ לאזורים, ותעריפי מחירים.

לאחר מכן, מציגה המערכת למשתמש את אופציית התשלום המתאימה והמשתלמת בשבילו, לאחר האישור, מקבל המשתמש למייל האישי שלו את פרוט התשלום והנסיעות שלו.

כל משתמש יכול לצפות בהיסטוריית הנסיעות וחישובי התשלום שלו באזור האישי באתר.

כך בצורה קלה וידידותית, יכול כל אחד לקבל את הזכויות המגיעות לו.

האתר יעודד נסיעה בתחבורה הציבורית, כל נוסע יהיה בטוח שהוא ישלם בסופו של דבר את המחיר הכי זול בעבור הנסיעות שנסע

אז נסיעה טובה, ומשתלמת!

# 1. מדריך למתכנת:

## 1.1. אסטרטגיות טכנולוגיות:

השתמשנו בשפות הבאות בכתיבת הפרויקט:

- **C#**: שפת C# היא שפת תכנות שפותחה ע"י מיקרוסופט ונחשבת לאחת משפות התכנות הפופולריות בעולם. היא C# מאתרי –מיועדת לפיתוח כללי של מגוון אפליקציות בכל התחומים, דרך משחקים, מאפליקציות Web למכשירי מובייל וטאבלטים ועד לשירותי ענן. התחביר והעקרונות שלה הם פשוטים מצד אחד אך עשירים ביכולות מצד שני.
- **Html**: html זוהי השפה הטבעית ליצירת דפי אינטרנט ברשת. זו שפה פשוטה, אוניברסלית, המאפשרת לעורכי אתרים ליצור דפים מורכבים שמכילים טקסט ותמונות, שיכולים להראות בידי כל המשתמשים ברשת האינטרנט ללא תלות בסוג המחשב או בסוג הדפדפן.
- **CSS**: CSS היא שפת עיצוב המגדירה את תבנית העיצוב של מסמכי HTML. CSS למשל מטפלת בגופנים, צבעים, גבולות, שורות, גובה, רוחב, תמונות רקע, מיקום מתקדם ודברים רבים נוספים.
- **ANGULAR7**: angular7 היא טכנולוגיה מתקדמת המאפשרת פיתוח מלא מקצה לקצה בצד לקוח, כלומר angular אספה התנהגויות שונות שעד היום נכתבו בצד השרת ואפשרה אותם בצד הלקוח (injectable, services, ועוד).
- **Angular** - תומכת ב pattern design של Angular. page single נכתבה בשפת typescript.
- **Typescript**: Typescript שפה חדשה בעלת כוח רב המומרת לשפת script java.
- **material-Angular**: material-Angular זוהי חבילת עיצוב,

השתמשנו בסביבות העבודה הבאות:

- **Visual Studio 2017**: סביבת הפיתוח המרכזית בעולם המייקרוסופטי נקראת Visual Studio. סביבה זאת מכילה מאות אפשרויות ותומכת במספר שפות תכנות. בנוסף, יש לציין כי רב שפות התכנות והטכנולוגיות הנתמכות על ידי סביבת פיתוח זו הן חלק מתשתית הפיתוח של חברת Microsoft framework .NET. תשתית זו מכילה מספר עצום של ספריות קוד וטכנולוגיות המקלות עלינו בתהליך יצירת התכנה.
- **SQL Server**: Microsoft SQL Server היא פלטפורמת שרת מידע ומבנה נתונים המיועדים למפתחים, ארגונים קטנים וגדולים כאחד, הפלטפורמה מצעה חבילה טכנולוגית שלמה של כלים ארגוניים המאפשרים



להשיג את הערך המרבי מתוך המידע שלנו במחיר ובעלות הכוללת הנמוכים ביותר. תוכנת ניהול השרת מיועדת למשתמשים הזקוקים לפתרונות ניהול מידע באמצעות בסיסי נתונים בצורה יעילה ומאובטחת. בנוסף ניתן לפתח ולנהל תוכנות למחשבים שולחניים ותוכנות מבוססות רשת העושות שימוש בבסיס נתונים המנוהל תחת שרת SQL Server במהלך העבודה עמן.

## 2. תיאור מבנה הפרויקט:

ה- solution שלנו כולל 3 פרויקטים:

DAL

פרויקט זה מורכב מקור נתונים- מסד הנתונים שלנו, וממערכת תוכנה entity framework אשר תפקידה לקרוא את המידע הנדרש למערכת, לשמור את העדכונים, ולהוסיף מידע חדש או למחוק פרטי מידע קיימים.

הפרויקט מובנה בשיטת database first entity framework ולכן db נבנה ראשון ועל סמך זה נבנו המחלקות והמאפיינים.

BL

הפרויקט שאחראי על הלוגיקה של המערכת עוסקת בעיבוד המידע, בחישובים שונים ושליחת לשכבת התצוגה.

בפרויקט זה נממש את הפונקציונליות של המערכת.

מסד הנתונים והממשק משתמש מתקשרים דרך שכבה זו

WEBAPI

הפרויקט שאחרי על חיבור בין ה server שזו שכבת ה bl לבין ה client שנמצא ב angular. בשכבה זו מתנהלות קריאות ה HTTP מה server ואילו.

## 2.1. עקרונות התכנון :

### 2.1.1. עקרונות תיאורטיים:

הפרדת שכבות

ישנם אתרים המבוססים על ארכיטקטורת שכבות הנקראת three-tier-architecture . בארכיטקטורה זו קיימת הפרדה בין השכבות במבנה של DAL-BL-GUI . זוהי תבנית עיצוב בסיסית שמגדירה הפרדת האפליקציה לשכבת הנתונים, שכבת הלוגיקה ושכבת ממשק המשתמש.

לתבנית עיצוב זו יתרונות רבים:

תחזוקה:

ניתן להחליף או לתקן מימוש נימי של שכבה אחת בארכיטקטורה בלי לשנות שכבה אחרת .

נוחות פיתוח:

אדם אחד עובד על רכיב בתכנה, אדם אחר עובד על רכיב אחר, כל עוד שהחתימות זהות ניתן לשלב כוחות ולייעל זמני פיתוח

בדיקות:

כאשר ישנה תקלה היא מבודדת בשכבה שאלי היא שייכת. לדוגמא אם לא קיבלנו רשימת נתונים לתצוגה, נבדוק קודם בשכבת הנתונים (DAL) האם הנתונים שם תקינים. במידה שהם אכן תקינים, נעבור לבדיקת שכבת הביניים, שכבת הלוגיקה (BL) ואם גם שם הנתונים תקינים, נדע בוודאות שהתקלה היא בשכבת התצוגה (GUI) ונפתור אותה שם.

שימוש חוזר:

במידה ששכבות ה DAL וה BL עובדות היטב וברצוננו להחליף את פלטפורמת ה UI, ניתן לעשות זאת במינימום מאמץ.

אבטחה

נח יותר להגדיר interface-API לשכבה מסוימת בלי לחשוף מבני נתונים , logic או data שלא רלוונטיים למשתמש, כמו גם חסימה בפני אקרים (רלוונטי יותר בטכנולוגיות WEB) ברמות שונות.

הפרויקט נכתב בשפת c# ומחולק לשלוש שכבות: GUI,BL,DAL

DAL- שכבת מסד הנתונים

BL- שכבת הלוגיקה

GUI- שכבת התצוגה למשתמש

**Language Integrated Query**

ובקיצור - LINQ שפה הצהרתית דמוית SQL לביצוע שאילתות, ש-Microsoft שילבה והיבנתה בשפות האימפרטיביות של ה-.NET. כ-C# ו-VB.NET. שפה זו מיועדת לעשות הפשטה על פעולות שגרתיות ופעולות על מקור נתונים, ולתווך בין המשתמש בה לבין ביצוע הפעולה הנדרשת. לשפה יש גם מימוש פונקציונלי, ניתן להשתמש בה גם באופן הצהרתי וגם באופן פונקציונלי ואף לערבב בין שתי הדרכים.

שפה זו, מוטמעת בשורות הקוד עצמו, לביצוע שאילתות על מבני נתונים כמערך (array) אוסף (collection), רשימה (list) וכל המשתנים המוגדרים ב-.NET. כטיפוסי IEnumerable יעודה של

השפה לאפשר למתכנת להתמקד במהות הפעולה המבוקשת ולא בטכניקה התכנותית ליצירת הפעולה. יתרונה בדומה לשפת SQL שהיא מאפשרת לתמצת קוד לשורות בודדות, לזרז את מהירות הפיתוח, להביא את הקוד לקריאות גבוהה, ולהפחית את רמת השגיאות בו.

בפרויקט השתמשנו בlinq במקרים רבים.

נציג דוגמאות:

```
//The travels for a particular user
travelsById = db.Travels.Where(x => x.userID == id &&
    x.date.Year == time.Year &&
    x.date.Month == time.Month)
    .OrderBy(x => x.price).ThenByDescending(x => x.areaID).ToList();
```

```
//join with 3 tables
extntionContract = contracts.Select(x => x.AreaToContracts.Join
(areaI1, AreaToCon1 => AreaToCon1.areaID, AreI1 => AreI1.id, (AreaToCon1, AreI1) => new { AreaToCon1, AreI1 }).Join
(areaI2, AreaToCon2 => AreaToCon2.areaID, AreI2 => AreI2.id, (AreaToCon2, AreI2) => new { AreaToCon2, AreI2 })
.OrderBy(f => f.AreaToCon2.AreaToCon1.Contract.freeDay).FirstOrDefault().AreaToCon2.AreaToCon1.contractID).FirstOrDefault();
```

```
//Select the all contract who may suitable for currnt user
contracts = db.Contracts.Where(x => x.AreaToContracts.Any
(m => m.Area.Travels.Any
(f => (f.userID == id && f.date.Year == time.Year && f.date.Month == time.Month))))).ToList();
```

## הורשה

בהורשה משתמשים בתכנות מונחה-עצמים, ירושה היא דרך לבסס יחס "סוג-של" (is-a) בין עצמים. המימוש הנפוץ של הגדרה זו היא בעזרת מחלקות. מחלקה יכולה לקבל בירושה תכונות והתנהגות של מחלקת האם שלה (נקראת גם "מחלקת-על", "מחלקת-אב", "מחלקת בסיס"). היחסים בין מחלקות מגדירים היררכיה ל מחלקות. מנגנון הירושה מסייע במידול של תחומים בדרך דומה לזאת המוגדרת באופן טבעי על ידי העוסקים בהם.

לדוגמה: אם ריבוע הוא סוג-של (או מקרה פרטי של) מצולע, ניתן להגדיר יחס ירושה בין המחלקה מרובע למחלקה מצולע. על פי רב, אובייקט של המחלקה היורשת (מרובע) יתמוך בכל הפעולות המוגדרות על מצולע כלשהוא, ובנוסף יתמוך בפעולות נוספות, המיוחדות למרובע. בדומה ניתן להגדיר יחס ירושה בין המחלקה ריבוע למחלקה מרובע.

שימושים:

ירושה מאפשרת להשיג מספר מטרות:

1. שימוש חוזר בקוד. בעזרת ירושה ניתן לכתוב מחלקות בעלות טווח התנהגות נרחב, תוך כתיבת קטעי קוד קטנים יחסית שמרחיבים מחלקה קיימת.
2. מידול של העולם האמיתי באופן דומה לזה המוגדר על ידי האדם. למשל ניתן לכתוב היררכיה של מחלקות המקבילה לטקסונומיה המקובלת בביולוגיה (יען היא סוג של ציפור, האדם הוא סוג של יונק, וכן הלאה).
3. polymorphism (רב צורתיות) של זמן ריצה. משתנה מטיפוס (סטטי) מסוים יכול להתנהג בצורות שונות, לפי הטיפוס של האובייקט שהוא מתייחס אליו בזמן ריצה. למשל משתנה מהמחלקה בעל

חיים שנדרש לבצע פעולה מסוג "השמע קול" יבצע פעולות שונות לחלוטין במקרה שהאובייקט הוא מהמחלקה כלב או חתול. ניתן לבצע זאת גם ללא ירושה (בעזרת פקודות תנאי) אך ירושה מאפשרת מימוש מודולרי, גמיש וקל להרחבה.

בפרויקט השתמשו בהורשה במקרים הבאים:

- כל controller יורש ממחלקת Controller.

לדוגמא מחלקת TravelController.

```
public class TravelController:ApiController
```

- מחלקת RAVKAV יורשת ממחלקת DbContext המקושרת למאגר הנתונים.

```
public partial class RavKav : DbContext
```

### LocalStorgae

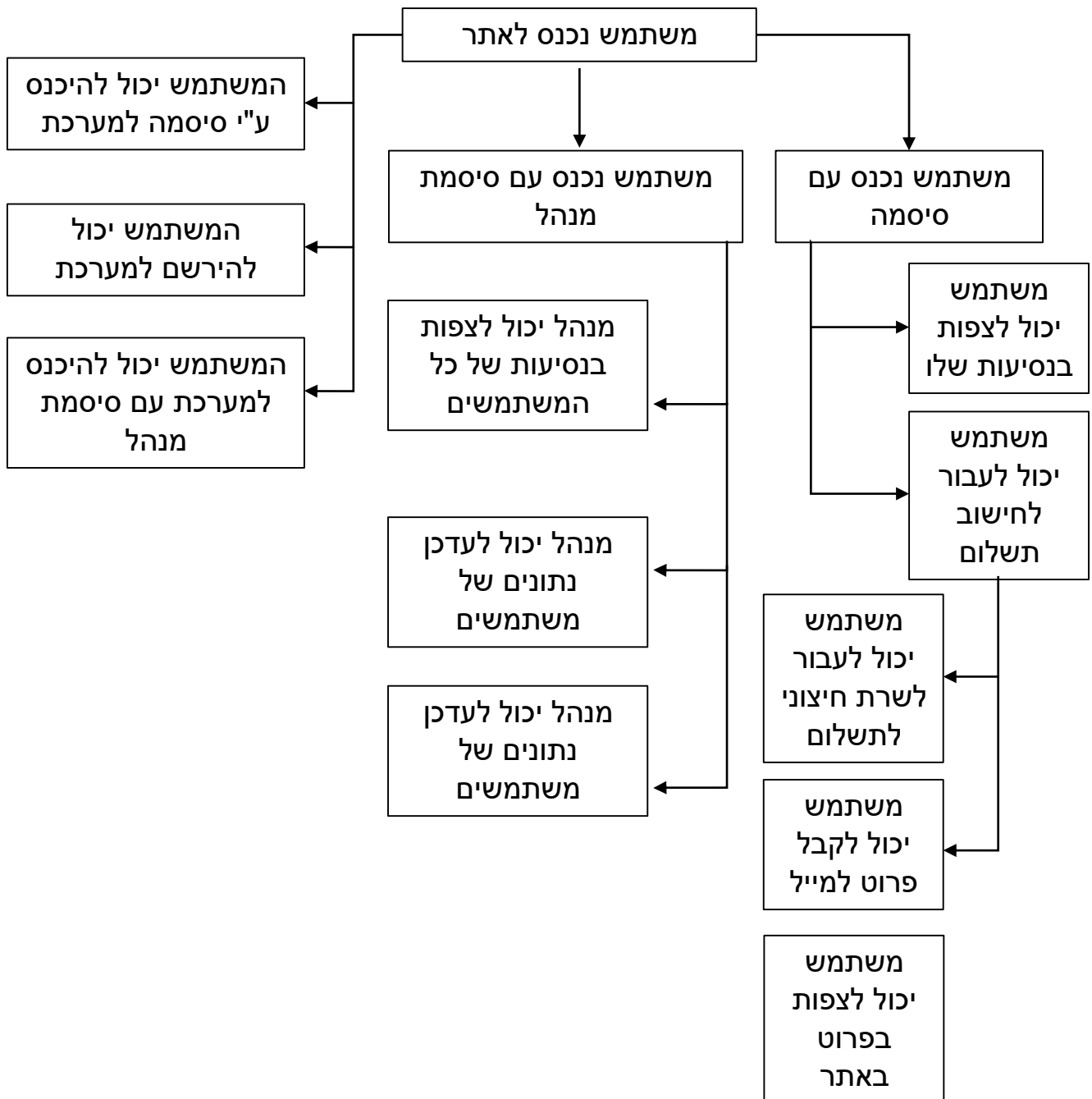
שמירת מידע בצד לקוח: זהו משאב אישי ללקוח, המידע אינו משותף ואינו נגיש ללקוחות האחרים. כאשר לקוח מבקר באתר, השרת מקצה לו אובייקט ב-localStorage ושומר אותו בזיכרון. האובייקט נשמר כל עוד המשתמש נשאר באתר, אם במשך זמן (מוגדר מראש) האתר ננטש, האובייקט נמחק. הוא נמחק גם כאשר סוגרים את הדפדפן, או כאשר נותנים הוראה למחיקתו.

יישום: בעת כניסה לאתר ע"י שם משתמש וסיסמה נשמר מספר המשתמש - localStorage כך במעבר בין דפים נשמרים הנתונים למשתמש הנוכחי.

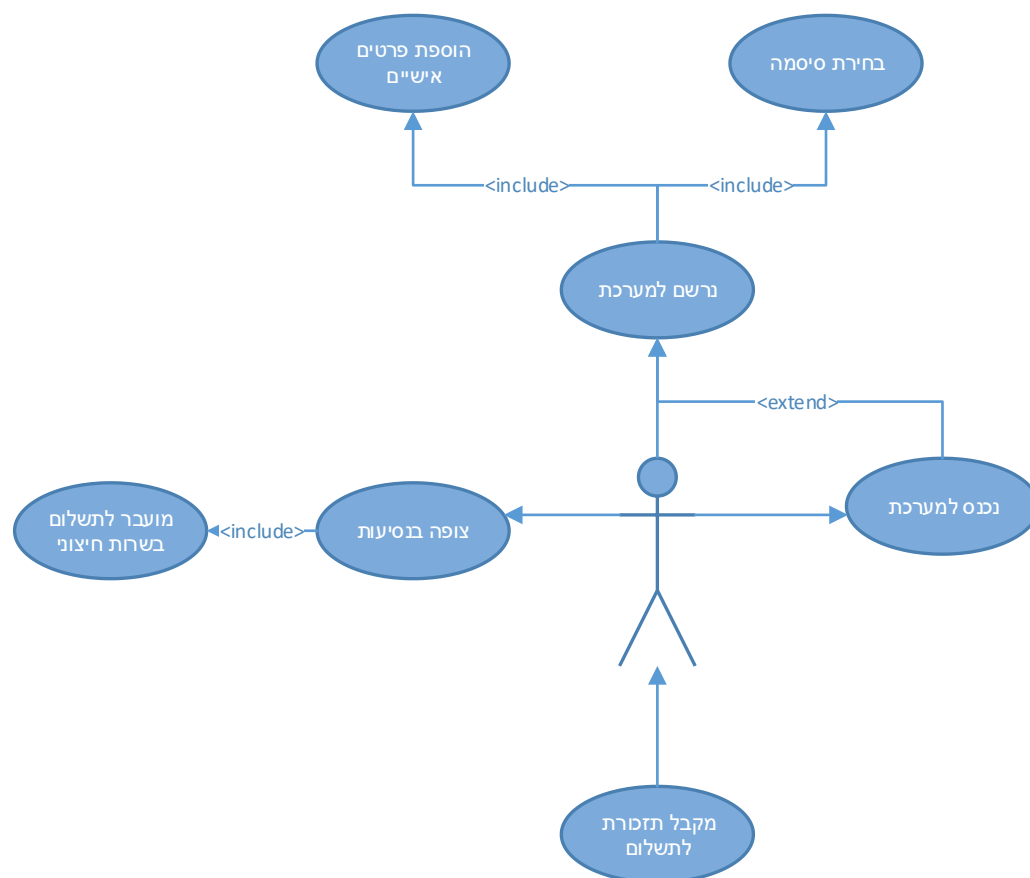
```
window.localStorage.setItem('ID',String(ID));
```

## 2.2. תרשימים:

### 2.2.1. עץ תהליכים:



## 2.2.2. תרשים Uml:



### 2.2.3. תרשים מראה המחלקות:

BL •



DAL •



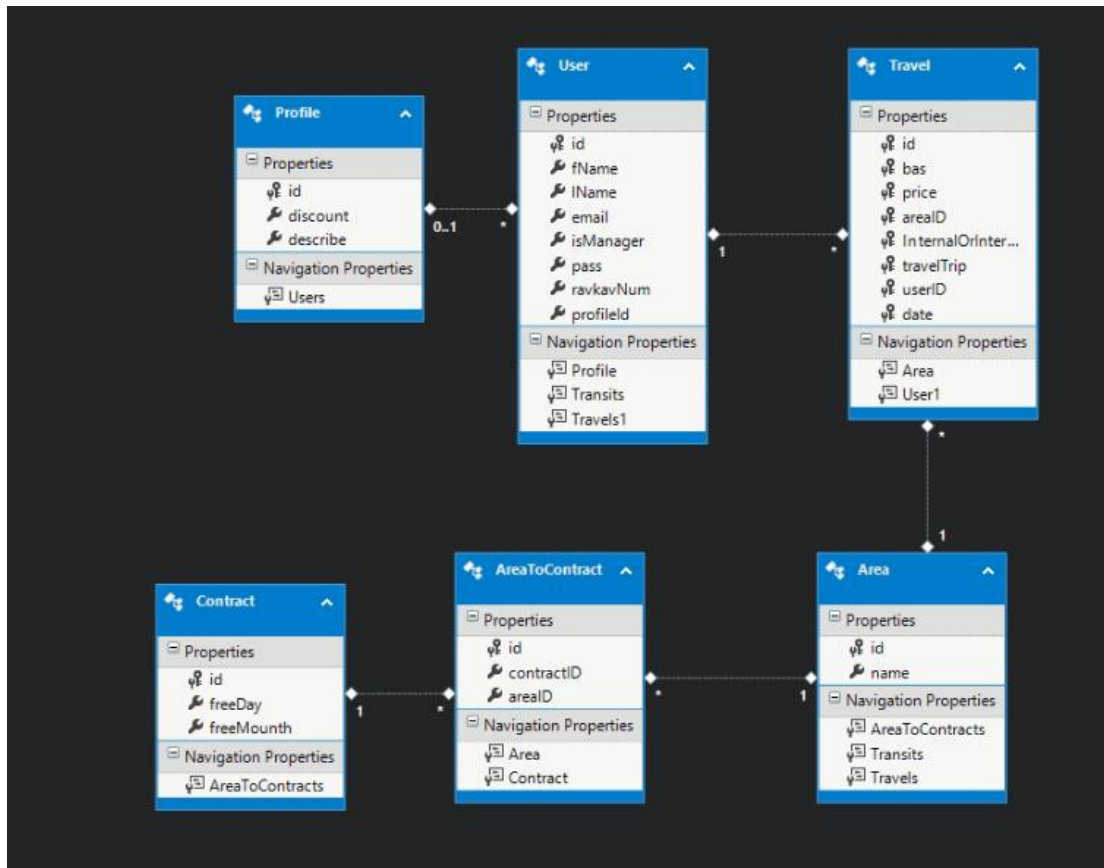
API •





### 3. מבנה נתונים מאוכסנים:

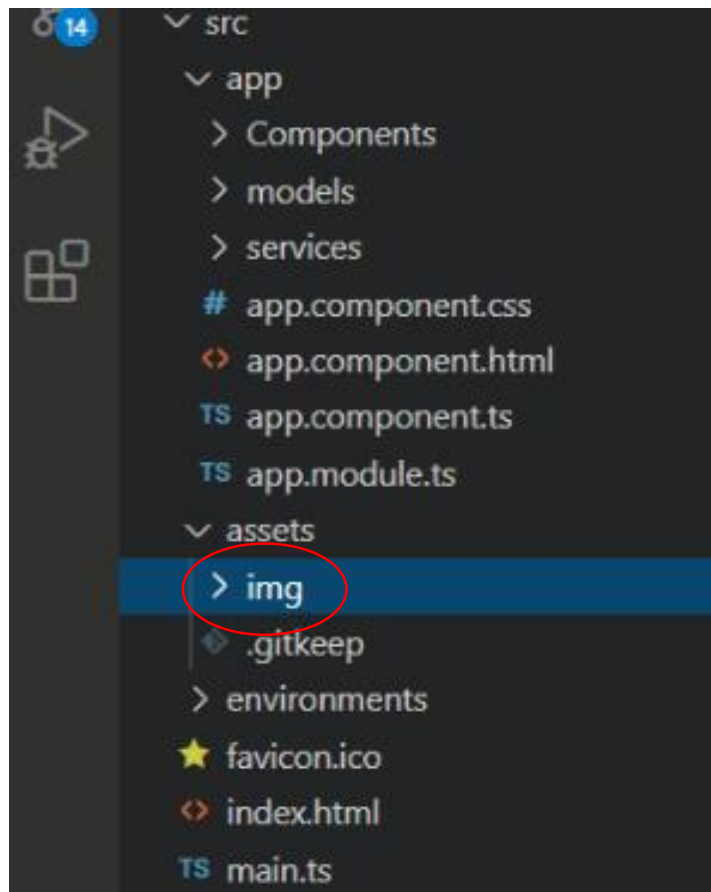
#### 3.1.1. תרשים SQL



טבלת User: פרטי משתמש במערכת  
 טבלת Travels: פרטי נסיעות למשתמשים  
 טבלת Profile: סוגי הפרופיל הקיימים ברב-קו  
 טבלת Contract: מחירי חוזים  
 טבלת Area: פרטי אזורים בארץ  
 טבלת AreaToContract: אזורים הקיימים בכל חוזה נסיעה

### 3.1.2. מבנה קבצים ותיקיות:

#### תיקיית תמונות לעיצוב האתר-



## 2.6 תוכן הפרויקט:

### 2.6.1 תיאור המחלקות:

#### מחלקות ב- Dal:

##### User

מחלקה זו מייצגת משתמש בודד ומכילה פרטים עליו-

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    14 references | tfkolitz, 14 days ago | 3 authors, 5 changes
    public partial class User
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        1 reference | tfkolitz, 14 days ago | 3 authors, 4 changes
        public User()
        {
            this.Travels = new HashSet<Travel>();
            this.Travels1 = new HashSet<Travel>();
        }

        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int id { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string fName { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string lName { get; set; }
        0 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string email { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public bool isManager { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string pass { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string ravkavNum { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public Nullable<int> profileId { get; set; }

        0 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public virtual Profile Profile { get; set; }
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual ICollection<Travel> Travels { get; set; }
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual ICollection<Travel> Travels1 { get; set; }
    }
}
```

## Travel

מחלקה זו מייצגת נסיעה ומכילה פרטים עליה-

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    17 references | tovikolitz1, 18 days ago | 1 author, 1 change
    public partial class Travel
    {
        1 reference | 0 changes | 0 authors, 0 changes
        public Travel(string bas, double price, int areaID, bool InternalOrIntermediate, int userID, System.DateTime date)
        {
            this.bas = bas;
            this.price = price;
            this.areaID = areaID;
            this.InternalOrIntermediate = InternalOrIntermediate;
            this.userID = userID;
            this.date = date;
        }

        2 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public int id { get; set; }
        3 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public string bas { get; set; }
        4 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public double price { get; set; }
        6 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public int areaID { get; set; }
        3 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public bool InternalOrIntermediate { get; set; }
        2 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public bool travelTrip { get; set; }
        6 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public int userID { get; set; }
        11 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public System.DateTime date { get; set; }

        2 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual Area Area { get; set; }
        0 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual User User { get; set; }
        0 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual User User1 { get; set; }
    }
}
```

## Area

מחלקה זו מייצגת אזור ומכילה פרטים עליו-

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    12 references | tfkolitz, 14 days ago | 3 authors, 5 changes
    public partial class Area
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        1 reference | tfkolitz, 14 days ago | 3 authors, 4 changes
        public Area()
        {
            this.AreaToContracts = new HashSet<AreaToContract>();
            this.Travels = new HashSet<Travel>();
        }

        6 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int id { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string name { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        3 references | tovikolitz1, 18 days ago | 3 authors, 3 changes
        public virtual ICollection<AreaToContract> AreaToContracts { get; set; }
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        3 references | tovikolitz1, 18 days ago | 1 author, 1 change
        public virtual ICollection<Travel> Travels { get; set; }
    }
}
```

## Contract

מחלקה זו מייצגת חוזה ומכילה את המחירים שלו

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    10 references | tovikolitz1, 18 days ago | 3 authors, 4 changes
    public partial class Contract
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        1 reference | tovikolitz1, 18 days ago | 3 authors, 3 changes
        public Contract()
        {
            this.AreaToContracts = new HashSet<AreaToContract>();
        }

        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int id { get; set; }
        4 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public double freeDay { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public double freeMounth { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        4 references | tovikolitz1, 18 days ago | 3 authors, 3 changes
        public virtual ICollection<AreaToContract> AreaToContracts { get; set; }
    }
}
```

## AreaToContract

מחלקה זו משייכת בין כל חוזה לאזורים הכלולים בו

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    10 references | tovikolitz1, 18 days ago | 3 authors, 4 changes
    public partial class AreaToContract
    {
        1 reference | 0 changes | 0 authors, 0 changes
        public AreaToContract()
        {
        }

        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int id { get; set; }
        4 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int contractID { get; set; }
        3 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int areaID { get; set; }

        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public virtual Area Area { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public virtual Contract Contract { get; set; }
    }
}
```

## Profile

מחלקה זו מייצגת חוזה ומכילה את הפרטים עליו

```
namespace DAL
{
    using System;
    using System.Collections.Generic;

    7 references | tovikolitz1, 18 days ago | 3 authors, 4 changes
    public partial class Profile
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]
        1 reference | tovikolitz1, 18 days ago | 3 authors, 3 changes
        public Profile()
        {
            this.Users = new HashSet<User>();
        }

        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int id { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public int discount { get; set; }
        2 references | tovikolitz1, 84 days ago | 1 author, 1 change
        public string describe { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]
        1 reference | tovikolitz1, 18 days ago | 3 authors, 3 changes
        public virtual ICollection<User> Users { get; set; }
    }
}
```

## מחלקות ב- web API:

## UserController

```

namespace Web_Api.Controllers
{
    ///[RoutePrefix("api/user")]
    [EnableCors(origins: "*", headers: "*", methods: "*", SupportsCredentials = true)]
    public class UserController : ApiController
    {
        [HttpGet]
        0 references | tovikolitz1, 85 days ago | 1 author, 1 change
        public IHttpActionResult IfExsistRavKav(string ravKav, string pass)
        {
            return Ok(UsersLogic.IfExsistRavKav(ravKav, pass));
        }

        [HttpGet]
        0 references | tovikolitz1, 85 days ago | 1 author, 1 change
        public IHttpActionResult GetNameById(int id)
        {
            return Ok(UsersLogic.GetNameById(id));
        }

        [HttpPost]
        0 references | tovikolitz1, 85 days ago | 1 author, 1 change
        public IHttpActionResult AddUser(UserDTO user)
        {
            return Ok(UsersLogic.AddUser(user));
        }

        [HttpGet]
        0 references | tfkolitz, 1 hour ago | 2 authors, 2 changes
        public IHttpActionResult AddUser(int id, string password)
        {
            var user = new UserDTO()
            {
                profileId = id,
                pass = password
            };
            return Ok(UsersLogic.AddUser(user));
        }
    }
}

```

## TravelController

```

4 using System.Web.Http;
5 using System.Web.Http.Cors;
6
7
8 namespace Web_Api.Controllers
9 {
10     ///[RoutePrefix("api/travel")]
11     [EnableCors(origins: "*", headers: "*", methods: "*", SupportsCredentials = true)]
12     0 references | tovikolitz1, 85 days ago | 1 author, 1 change
13     public class TravelController:ApiController
14     {
15         [HttpGet]
16         0 references | 0 changes | 0 authors, 0 changes
17         public IHttpActionResult calaulateThePayment(int id, DateTime time)
18         {
19             return Ok(TravelLogic.calaulateThePayment(id,time));
20         }
21     }
22 }

```

## 2.6.2 תיאור הפונקציות:

- הפונקציה: IfExsistRavKav()  
 הפונקציה מקבלת מספר רב-קו וסיסמא.  
 מטרת הפונקציה לוודא שהכניסה למערכת תקינה,  
 הפונקציה מחזירה את פרטי המשתמש, ובמידה ולא קיים, או שהסיסמה שגויה, ערך ריק.

```
public static UserDTO IfExsistRavKav(string ravKav, string pass)
{
    //Get user details
    User user = db.Users.ToList().FirstOrDefault(x => x.ravkavNum == ravKav &&
    x.pass == pass);
    //Security
    if (user != null)
    {
        return Conversions.Conversion(user);
    }

    return null;
}
```

- הפונקציה : ContractBase  
 פונקציה זו מקבלת את רשימת הנסיעות של נוסע מסוים ממוינת לפי: תאריך, אזור ותעריף,  
 ורשימת החוזים שיכולים להתאים לנוסע, מטרת הפונקציה למצוא את החוזה הבסיסי ביותר  
 הקיים לכל אזור לפי תעריף של חופשי יומי,  
 חוזה בסיסי הוא חוזה המורכב מנסיעת הלוך – חזור, ולפחות עוד נסיעה פנימית אחת באותו  
 אזור,  
 השילוב של נסיעת הלוך חזור עם נסיעה פנימית מבטיח מחיר משתלם יותר בקניית חוזה  
 חופשי לאזור.

```
//find base contract
//the rule of base contract is:
//contract who as Travels back and forth
//With at least one more internal trip
public static void ContractBase(List<Contract> contracts, List<Travel> travelsById)
{
    IDictionary<Travel, int> travelUsed = new Dictionary<Travel, int>();
    int cnt = 0;
    List<Travel> travelsToCurrentContract;
```



```

//Go over the travel list for the user
for (int i = 0; i < travelsById.Count; i++)
{
    //Go over the travel list for the user
    if (travelsById[i].areaID == travelsById[i + 1].areaID)
    {
        //Finding the the travels in a right and cheapest contract
        travelsToCurrentContract = GetTravelOfCheapestContract(travelsById[i]);

        foreach (var item in travelsToCurrentContract)
        {
            //Check if some of the trips have already been realized in other contracts
            if (!travelUsed.ContainsKey(item))
                cnt++;
        }

        //Check that there are more than three trips that have not been realized in any
        contract
        if (cnt >= 3)
        {
            //Add the travels to the travels list that for them a contract has been found
            foreach (var item in travelsToCurrentContract)
            {
                if (!travelUsed.ContainsKey(item))
                    travelUsed.Add(item, 0);
            }
        }
    }
}

```

• הפונקציה: GetTravelOfCheapestContract()

פונקציה זו מקבלת נסיעה, הפונקציה מחשבת מהו החוזה הבסיסי לנסיעה, ומחזירה את הנסיעות הנוספות של הנוסע הנוכחי, שהתבצעו באותו אזור של החוזה הנבחר.

```

public static List<Travel> GetTravelOfCheapestContract(Travel currentTravel)
{
    //Finding the right and cheapest contract

    var currentContractID = currentTravel.Area.AreaToContracts.OrderBy(x =>
        x.Contract.freeDay).FirstOrDefault().id;

    //Pulling out all travel appropriate to the contract

    var travelsToCurrentContract = travelsById.Where(x =>
        x.Area.AreaToContracts.Any(m => m.contractID == currentContractID)).ToList();

    return travelsToCurrentContract;
}

```

- הפונקציה: findcontractExtentionOfToSmallContracts()  
 הפונקציה מקבלת אינדקס של מיקום חוזה  
 מטרת הפונקציה להרחיב את החוזה שבמיקום האינדקס, לחוזה גדול יותר, ע"י צירוף של  
 שני חוזים צמודים לחוזה שלישי,  
 הפונקציה מחזירה את החוזה המורחב, ובמידה ואין אפשרות להרחיב את החוזה, הפונקציה  
 תחזיר 0.

```

public static int findcontractExtentionOfToSmallContracts(int index)
{
    int extntionContract = 0;
    List<Area> areal1 = contractUsed.ElementAt(index).Value;
    List<Area> areal2 = contractUsed.ElementAt(index + 1).Value;

    extntionContract = contracts.Select(x => x.AreaToContracts.Join(areal1,
        AreaToCon1 => AreaToCon1.arealID, Arel1 => Arel1.id, (AreaToCon1, Arel1) => new {
        AreaToCon1, Arel1 }).Join(areal2, AreaToCon2 => AreaToCon2.Arel1.id, Arel2 =>
        Arel2.id, (AreaToCon2, Arel2) => new { AreaToCon2, Arel2 }).OrderBy(f =>
        f.AreaToCon2.AreaToCon1.Contract.freeDay).FirstOrDefault().AreaToCon2.AreaToCo
        n1.contractID).FirstOrDefault();

    return extntionContract;
}

```

## 4. מדריך למשתמש:

### 4.1. הוראות כלליות לשימוש באתר:

#### 4.1.1. מדריך למשתמש רגיל

כאשר משתמש נרשם למערכת יש באפשרותו לבצע מספר פעולות

1. הוא יכול לעדכן את הפרטים האישיים שלו
2. המשתמש יכול לצפות בכל הנסיעות שנסע במהלך החודש האחרון
3. המשתמש יכול לצפות בסה"כ לתשלום עבור הנסיעות של החודש האחרון
4. המשתמש יכול לעבור לתשלום עבור הנסיעה
5. המשתמש יכול לקבל למייל פרוט על הנסיעות שלו ועל התשלום עליו לשלם

#### 4.1.2. מדריך למנהל:

כאשר מנהל נכנס למערכת יש באפשרותו לבצע את הפעולות הבאות על כל המשתמשים במערכת

1. הוא יכול לצפות בנסיעות של כל המשתמשים.
2. לעדכן נתונים על המשתמש
3. לעדכן משתמש כמנהל

## 4.2. צילומי מסכים:

בכניסה לאתר המשתמש נדרש להזין מספר רב-קו וסיסמה בנוסף קיימת אפשרות של שחתי סיסמה והמערכת תאפשר למשתמש להחליף סיסמה וכן משתמש שאינו רשום, הלחיצה על הקישור משתמש חדש הוא יועבר ליצירת חשבון





**MY-KAV**

משתלם בדרך שלי

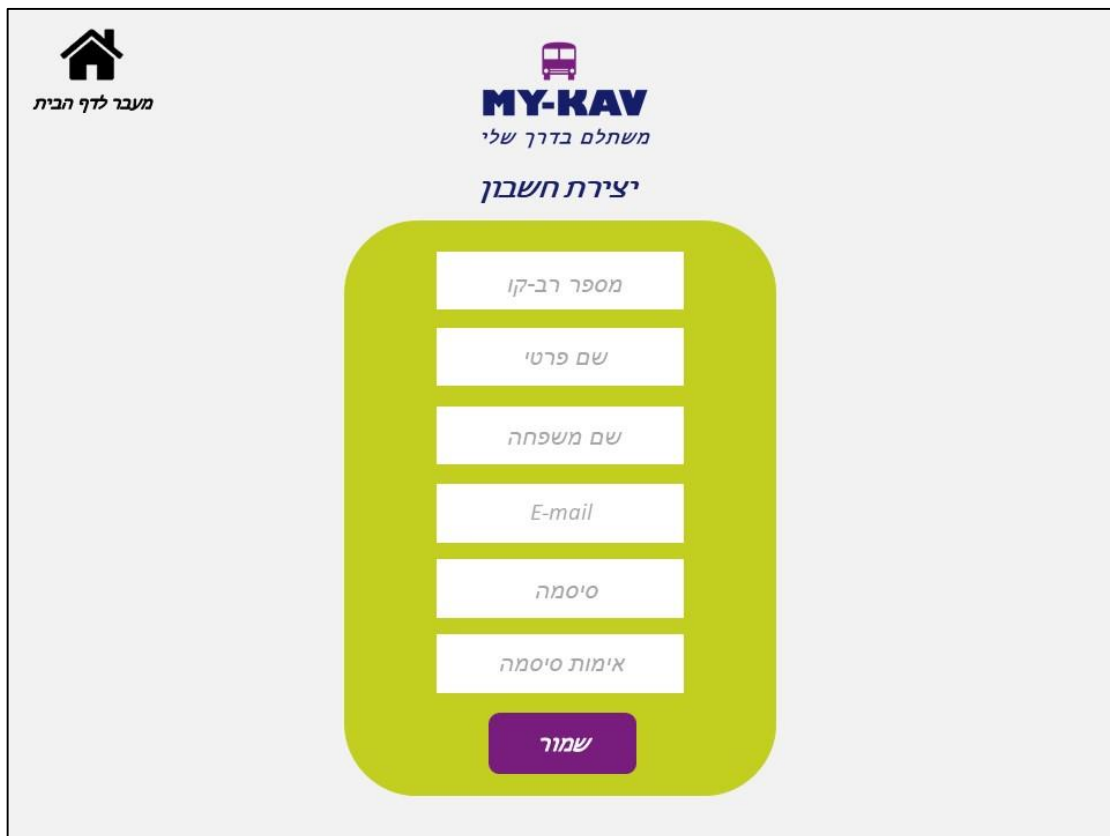
מספר רב-קו

סיסמה

**כניסה**

[שחתי סיסמה](#) | [משתמש חדש](#)

## יצירת חשבון למשתמש חדש



מעבר לדף הבית

**MY-KAV**  
משתלם בדרך שלי

**יצירת חשבון**

מספר רב-קו

שם פרטי

שם משפחה

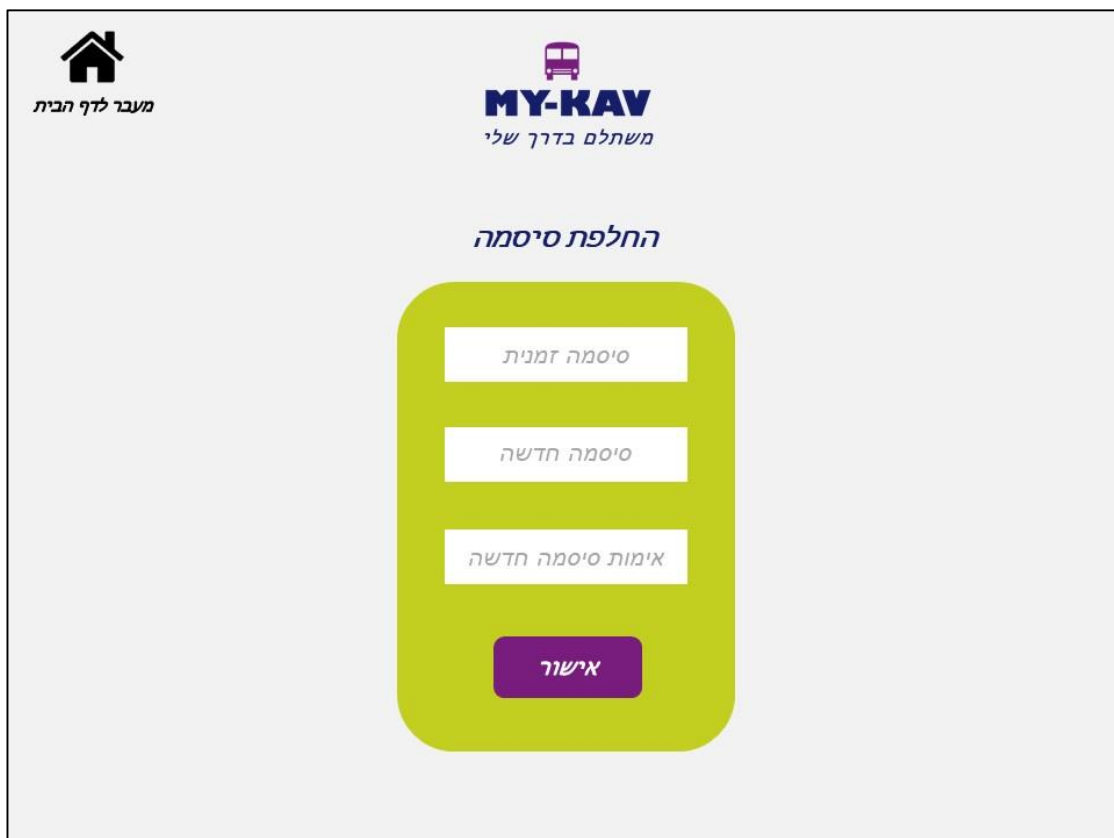
E-mail

סיסמה

אימות סיסמה

שמור

במסך החלפת סיסמה המערכת תשלח למייל המופיע במערכת למשתמש לפי מספר רב קו סיסמא זמנית והמשתמש יוכל להחליף את הסיסמה שלו



מעבר לדף הבית

**MY-KAV**  
משתלם בדרך שלי

**החלפת סיסמה**

סיסמה זמנית

סיסמה חדשה

אימות סיסמה חדשה

אישור

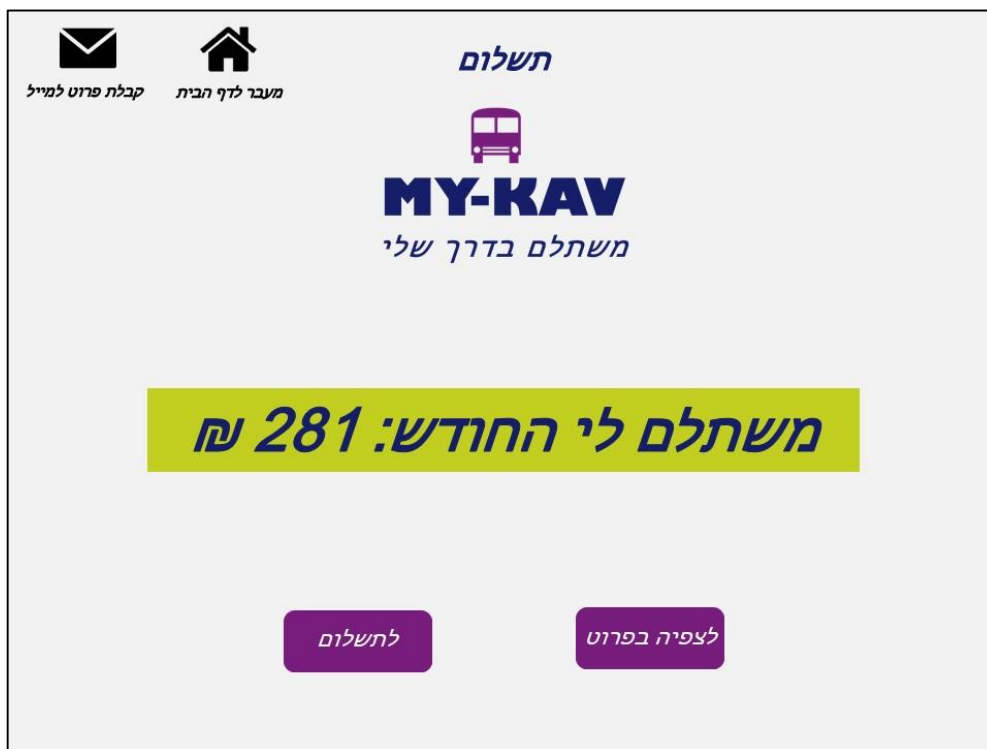
לאחר הכניסה למערכת המערכת תאפשר למשתמש לצפות בנסיעות שלו או לעבור לחישוב תשלום וכן מכאן בכל שלב יוכל המשתמש לבחור באופציה של קבלת פרוט למייל



בצפייה בנסיעות יפתח לו חלון בו יכול לראות את כל הנסיעות שלו




במעבר לחישוב תשלום המערכת תציג למשתמש את התשלום הסופי הזול ביותר בעבורו  
עם אופציות לצפייה בפירוט ולמעבר לתשלום



במסך צפייה בפרוט תהיינה למשתמש שלוש אפשרויות



בחוזים שנבחרו עבורו יוכל המשתמש לראות את כל החוזים אותם יצא לו משתלם לרכוש החודש

קבלת פרוט למייל
מעבר לדף הבית

## החוזים שנבחרו עבורי



# MY-KAV

משתלם בדרך שלי

שם	מחיר
גוש דן חודשי	₪ 213
סובב ירושלים יומי	₪ 13.5
אזור בית שמש יומי	₪ 13.5
השרון ונתניה יומי	₪ 21.5

נסיעות הנכללות בחוזים

נסיעות בודדות

לתשלום

בנסיעות הנכללות בחוזים יוכל המשתמש לראות את כל הנסיעות שנכללות בתוך החוזים הנבחרים




קבלת פרוט למייל
מעבר לדף הבית

## נסיעות הנכללות בחוזים



# MY-KAV

משתלם בדרך שלי

תאריך	שעה	קו	נכלל בחוזה
02/11/2020	09:00	5	גוש דן חודשי
04/11/2020	10:02	9	גוש דן חודשי
04/11/2020	23:15	8	גוש דן חודשי
06/11/2020	12:41	11	גוש דן חודשי
07/11/2020	10:35	32	סובב ירושלים יומי
07/11/2020	11:06	57	סובב ירושלים יומי
07/11/2020	15:54	71	סובב ירושלים יומי

החוזים שנבחרו עבורי

נסיעות בודדות

לתשלום



בנסיעות בודדות יוכל המשתמש לראות את כל הנסיעות שלא נכללו באף חוזה ושולמו כנסיעות בודדות

 קבלת פרוט למייל

 מעבר לדף הבית

## נסיעות בודדות



# MY-KAV

משתלם בדרך שלי

תאריך	שעה	קו	מחיר
01/11/2020	15:30	402	₪ 16
01/11/2020	17:40	402	₪ 16
02/11/2020	09:00	5	₪ 5.9
04/11/2020	22:28	292	₪ 5.9
07/11/2020	08:32	497	₪ 14
09/11/2020	10:45	15	₪ 7.5
11/11/2020	18:09	980	₪ 23

נסיעות הנכללות בחוזים

החוזים שנבחרו עבורי

לתשלום

בכניסת משתמש רשום כשהמערכת תזהה שמדובר במנהל היא תכניס אותו למסך שונה ממשתמש רגיל עם אופציות נרחבות יותר

 מעבר לדף הבית

## סביבת מנהלים



# MY-KAV

משתלם בדרך שלי

עדכון משתמש כמנהל

עדכון נתוני משתמש

צפייה בנסיעות משתמשים

עדכון פרטים

חישוב תשלום

צפייה בנסיעות שלי

במסך צפייה בנסיעות משתמשים יתאפשר למנהל לצפות בנסיעות לפי בחירת שם משתמש מסוים



מעבר לדף הבית

**צפייה בנסיעות משתמשים**




**MY-KAV**

משתלם בדרך שלי

בחר שם משתמש ▼


תאריך	שעה	קו
01/11/2020	15:30	402
01/11/2020	17:40	402
02/11/2020	09:00	5
04/11/2020	10:02	9
04/11/2020	22:28	292

בעדכון נתוני משתמש המערכת תאפשר למנהל האם לבחור משתמש על ידי שם או על ידי מספר רב-קו



מעבר לדף הבית

**עדכון פרטי משתמש**



**MY-KAV**

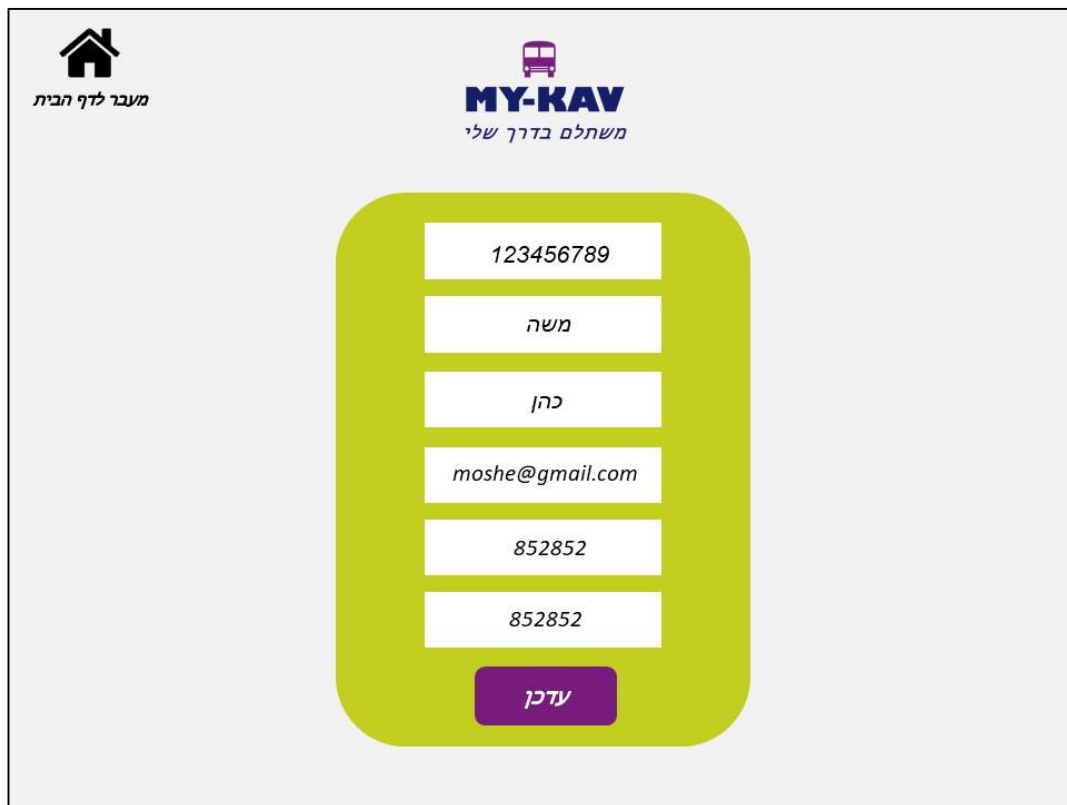
משתלם בדרך שלי

שם משתמש ▼

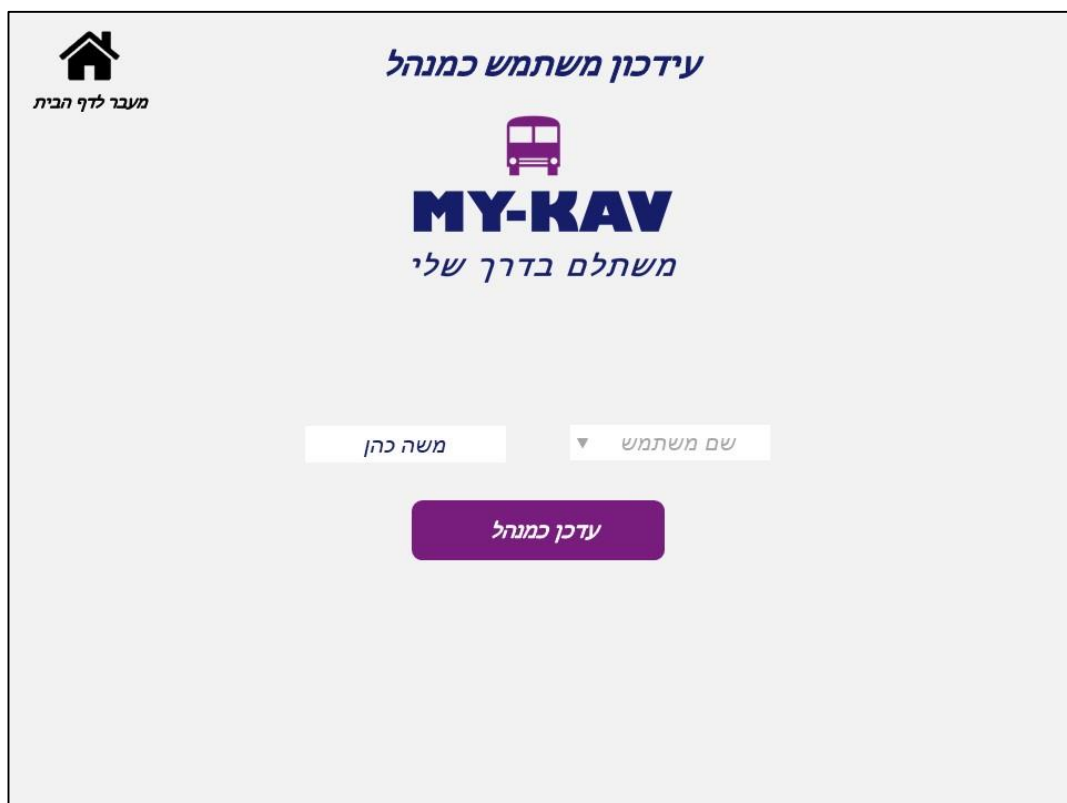
משה כהן

צפייה ועדכון פרטים

המערכת תעלה את פרטי המשתמש הנבחר ותיתן אפשרות לעדכון פרטים



במסך עדכון משתמש כמנהל המערכת תאפשר למנהל לבחור משתמש ולעדכן אותו כמנהל



שאר האופציות של מנהל הם כשל משתמש רגיל

## 5. סיכום ומסקנות:

כל יום של כתיבת הפרויקט היוותה עבורינו התקדמות בהכרת עולם ההייטק, ובזכות ההיקף של הפרויקט נתקלנו באתגרים רבים שלשמחתינו ההתמודדות איתם תרמה רבות לידע שלנו והפכה את העבודה למהנה ומעניינת יותר.

אחרי עבודה רבה על האתר אנחנו שמחות לראות שאכן הוא מממש את היכולות שתכננו עבורו. אנשים יכולים להרשם לאתר שלנו ולהיות בטוחים שהם משלמים את המחיר הכי זול בעבור הנסיעות שנסעו במהלך החודש האחרון ואינם משלמים מחיר שיכלו לחסוך אותו לו היו יודעים בדיוק את תכנון הנסיעות שלהם

בפרויקט הכנסנו נתונים בסיסיים לדוגמא כדי להראות את אופן השימוש באתר. האתר יכול לעלות לרשת האינטרנט עם הסכמה של משרד התחבורה ולעודד אנשים להגביר את השימוש בתחבורה ציבורית בידיעה שהם משלמים את המחיר הזול ביותר

## 6. נספחים:

בפרויקט השתמשנו ב-dictionaty כדי להקל על מערך החיפוש שצריך לבצע עבור כל משתמש

```
//Dictionary of contracts, with their own areas  
static IDictionary<int, List<Area>> contractUsed = new Dictionary<int, List<Area>>();
```

```
//Dictionary of used travels  
IDictionary<Travel, int> travelUsed = new Dictionary<Travel, int>();
```

## 7. ביבליוגרפיה:

אתרים עבור העיצוב:

- angularjswiki.com •

אתרים בנושא תכנות:

- stackoverflow.com •

- webmaster.org.il •

- w3schools.com •

- docs.microsoft.com •

- developer.mozilla.org •

- geeksforgeeks.org •