

פרויקט גמר

הנדסת תוכנה מה"ט



מגישות: ליאל פרץ והדסה רגב.
מנחה: גב' מירי ויכלדר.



תוכן עניינים

4.....	מבוא
5.....	הגדרת דרישות ותיאור כללי
6.....	מדריך למתכנת
7.....	תיאור המערכת- תרשימים
7.....	Use case תרשימי
7.....	Data flow תרשימים
8.....	דיאגרמת פעילויות
8.....	תרשימי זרימת מסכים
9.....	תכנות
9.....	תיאור
9.....	טכנולוגיות מתקדמות
10.....	עקרונות התכנות
10.....	חלופות שפות תכנות:
11.....	בעיות ופתרון
12.....	אבטחה
12.....	שרידות המערכת ותיעוד
13.....	ארכיטקטורת המערכת
13.....	תיאור השכבות:
13.....	תיאור קוד פונקציות ומחלקות
23.....	מסד הנתונים
28.....	בדיקות תוכנה:
30.....	טבלת בדיקות מסכמת
30.....	Test case:
30.....	בדיקות פונקציונליות:
31.....	בדיקות שימושיות:
31.....	בדיקות GUI:
33.....	מדריך למשתמש
42.....	סיכום
43.....	מקורות מידע

רצינו לומר תודה...

לאחר חודשים ארוכים של השקעה, יגיעה ומאמץ חודשים של עליות וירידות, הצלחות וגם נפילות, אנו רוצות לומר תודה לכל המסייעים ואלו שהפכו את הפרויקט הזה לאפשרי.

תודה רבה לגב' מירי ויכלדר, רכזת המגמה על כל הדאגה, המסירות וההדרכה.

תודה מיוחדת לגב' תמר קארפ, על הליווי הצמוד במהלך הפרויקט, עוד משלב החשיבה וגיבוש הנושא המרכזי, היא הייתה שם לכוון אותנו, להקשיב גם לרעיונות פחות טובים ולמצוא גם בהם את הנקודות הטובות עד שהגענו לרעיון שאהבנו. לאורך כל הדרך היא הקשיבה הסבירה וכיוונה אותנו עד הפרטים הקטנים. תודה רבה!

כמו כן אנו רוצות להודות לגאולה המנחה שלנו, קודם כל תודה על הדאגה, הרגשנו שזה חשוב לה לא פחות משזה חשוב לנו.

הרגשנו שתמיד יש לנו למי לפנות בכל שאלה או התייעצות שעלתה. תמיד היה לה מענה מקצועי, היא לא ויתרה לנו לרגע על שום פרט אך עם זאת הייתה שם לעודד שלא נתייאש. ידעה להסביר והכל בסבלנות בלי סוף! מעריכות מאוד.

ונסיים בתודה ענקית לבורא עולם שבזכותו אנחנו כאן.

 תודה

מבוא

אנו חיים בעידן טכנולוגי בו כולנו מחפשים את הדבר הבא, את החידוש שיהפוך את החיים שלנו לקלים ונוחים יותר, הכל חייב להיות נגיש ומהיר- אם לא הוא עלול להפוך ללא רלוונטי.

כשחשבנו על רעיון לפרויקט רצינו ליצור משהו שיעמוד בקריטריונים של מהיר ונגיש, היה חשוב לנו שהוא יעסוק בנושא שיגע בעולמות של שתינו, משהו שנראה את עצמנו משתמשות בו גם כן. תמיד עמד מול עיננו החשיבות שהפרויקט יהיה שימושי וכמובן בנוי בצורה הנוחה ביותר למשתמש ועם זאת להתמקד בתהליך הלמידה וההתפתחות שלנו בעזרת הטכנולוגיות הרווחות בשוק כיום על מנת לצבור ידע וניסיון.

וכך נולד האתר **Quili**-Quick list רשימה מהירה.

כמה פעמים נתקלתם במתכון, רציתם להכין אותו ו... חסר לכם מצרך מסוים אז אתם מוותרים ועוברים לדבר הבא או מנסים לאלתר ממה שיש. מה אם נשנה את הגישה, נבנה את רשימת הקניות שלנו לפי המתכונים אותם אנו רוצים להכין.

האתר **Quili** בא לנהל לנו את התכנון היומיומי של ניהול המטבח שלנו, החל מתכנון ארוחות קבועות או חד פעמיות ועד לניהול רשימת הקניות שלנו בצורה נוחה והכי חשוב- מסודרת ויעילה יותר.

שם האתר **Quili**-Quick list רשימה מהירה משקף את הרעיון שעומד מאחורי הפרויקט- בניית רשימת קניות מהירה וקלה.

התרגשנו לראות את הפרויקט נבנה מאפס, למדנו דברים חדשים התנסינו בטכנולוגיות וספריות שלא הכרנו והשתדלנו לעשות את זה בדרך האוטנטית על מנת להביא את עצמנו לכתיבת אתר ברמה גבוהה בדרך הנהוגה בשוק העבודה כיום.

בכל שלב בפרויקט העמדנו את עצמנו במקום של המשתמש, ניסינו לחוות את האתר מהצד ולהבין איך להשתפר בהתאם. הקשיים והאתגרים הרבים במהלך הפרויקט גרמו לנו להתאים את עצמנו ואת תהליך העבודה שלנו אך לא לוותר בשום שלב על פיצ'רים שימושיים באתר או על חלקים העלולים לפגום בחוויית המשתמשים באתר.

הנה התוצאה לפניכם.

הגדרת דרישות ותיאור כללי

מטרת המערכת:

מטרת המערכת היא להפוך, בלחיצה אחת, כל מתכון או תפריט לרשימת קניות חכמה. המערכת מאפשרת למשתמש לשמור ולנהל מתכונים או להוסיף אותם לתפריט המזון השבועי שלו ויוצרת באופן מיידי רשימת קניות על פי מתכונים אלו- באפשרות המשתמש לנהל ולערוך רשימה זו.

היקף העבודה:

מספר השעות שהוקדשו עבור פרויקט זה עומד על כ- 850 שעות.

תיאור חומרת המערכת:

מחשב PC / מכשיר נייד בעל דפדפן וחיבור לרשת.

תיאור תוכנת המערכת:

כתיבת צד לקוח נעשה בשפת Angular8, שפה פופולארית הנוחה מאוד הן למתכנת והן למשתמש.

כתיבת צד שרת נעשה בשפת C#, שפה מתקדמת, שימושית ומלאת פונקציונאליות.

בניית מסד הנתונים נעשה באמצעות Server SQL בטכנולוגית Entity Framework.

מדריך למתכנת

תיאור המערכת:

אתר המאפשר לכל משתמש לנהל את המטבח שלו, החל מתכנון ארוחות וניהולן ועד לניהול רשימת הקניות בהתבסס על המתכונים/התפריט הנבחר. הלקוח בוחר את המתכונים הרצויים ושומר אותם לפי הימים בחודש, המערכת בונה לו על פי המתכונים הרצויים רשימת קניות מתאימה.

ממשק המערכת:

הפרויקט פותח בסביבת Web ומספק למשתמש יכולת לנהל את תכנון התפריט שלו ובהתבסס על כך את רשימת הקניות שלו. באפשרות המשתמש לשמור מתכונים ע"י התממשקות לAPI חיצוני עם מאגר מתכונים רחב בשם spoonacular ולפיהם לבנות את התפריט וכן את רשימת הקניות שלו על פי הנתונים שבחר. ממשק המשתמש באתר בנוי בצורה ידידותית למשתמש עם הסברים והוראות ברורות לשימוש.

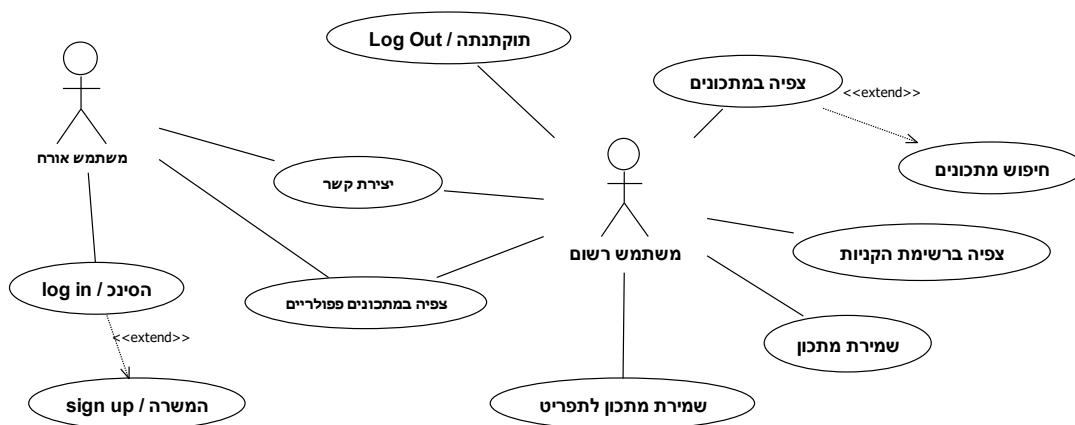
תיאור המערכת- תרשימים

Use case תרשימי

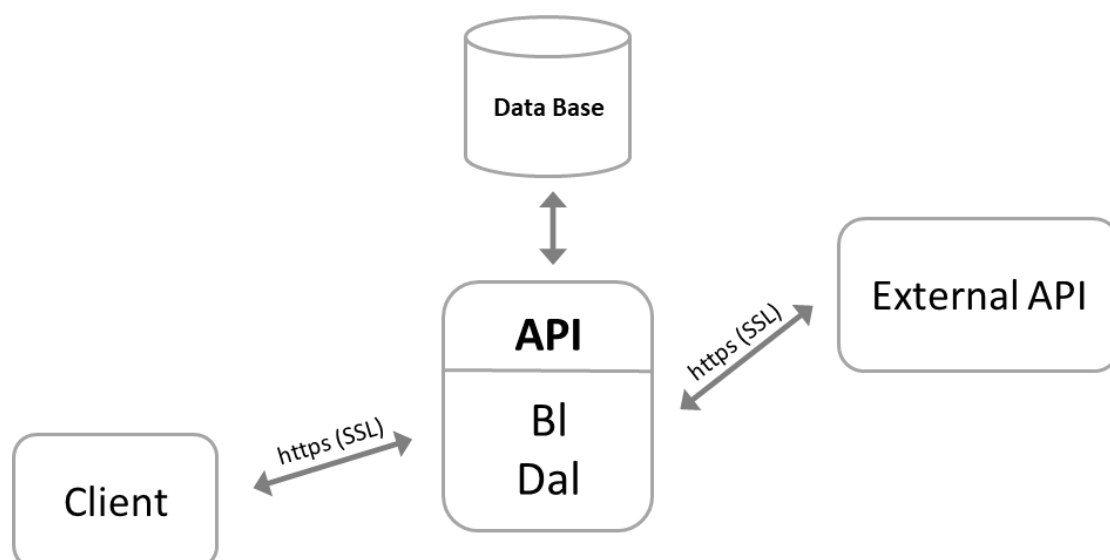
משתמש לא רשום-

ליצור קשר

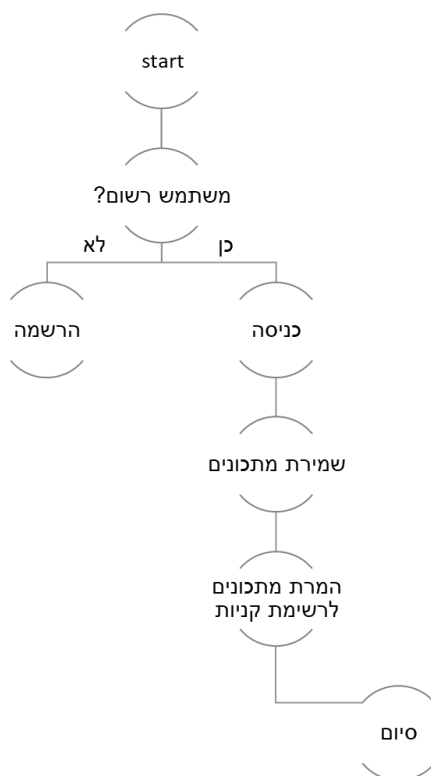
צפיה במתכונים



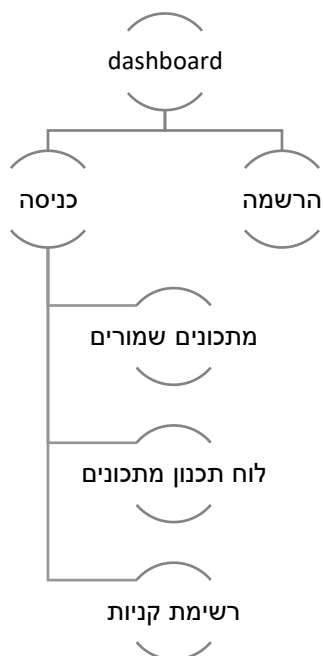
תרשימי Data flow



דיאגרמת פעילויות



תרשימי זרימת מסכים



תכנות

תיאור

המערכת נכתבה בשפת C#.NET בסביבת Visual Studio 2019 ועושה שימוש בטכנולוגיות מתקדמות, בתוכם: Web API, Angular8, Entity Framework וכן, בספריית Bootstrap.

כאשר צד ה-Server פותח בטכנולוגיית Web API בשיטות המתקדמות ביותר ש-NET מציע וצד ה-Client פותח באמצעות Angular8.

טכנולוגיות מתקדמות

- א. API Web – ממשק תכנות יישומים לשרתי אינטרנט ו/או לדפדפני אינטרנט. מטרתו "לתווך" בין צד הלקוח (client) לצד שרת (server) על מנת לבצע את הפונקציונליות הנדרשת לצד לקוח. עיקר תפקודו הוא בקבלת נתונים מצד הלקוח והעברתם לשרת, ולהפך.
קטע קוד לדוגמא: קבלת בקשת GET לקבלת רשימת המתכונים של לקוח מסוים, קבלת בקשת post להוספת מתכון.
- ב. LINQ – שפת צד שרת, שיטת כתיבה חדשנית מובנית ב-NET, המיועדת לריצה על lists מערכים של אובייקטים וחיפוש בתוכם בצורה פשוטת, חכמה וקלילה.
קטע קוד לדוגמא: שליפת מתכון מסוים ע"י שפת LINQ בצורה ישירה וקלילה, ללא משפטי תנאי מיותרים.
- ג. Entity Framework – טכנולוגית עבודה מתקדמת מול DB. יצירת אובייקטים ומיפוי מול טבלאות ה-DB, כאשר אופן העבודה ועיבוד הנתונים בשפת LINQ כמוזכר לעיל.
- ד. Angular8 – 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 באפשרות ליצור אפליקציות של S.P.A – Single Page Application, כלומר, רק חלק בדף מסוים מתרענן ומתחלף וכל שאר החלקים נשארים אותו הדבר.
דבר זה מאפשר גלישה מהירה וחלקה אשר משפרת את חווית המשתמש. בכל קומפוננט (רכיב המיועד לתצוגה, המכיל את כללי התצוגה ואת הניהול שלה) בה רוצים להציג תוכן כלשהו משתמשים במאפיין – template. מאפיין זה אומר ל-Angular איזה תוכן HTML להציג עבור קומפוננט זו, templaten מכיר את הקומפוננט ואת כל השדות והפונקציות הכתובות בה. Angular משתמשת בשיטת ה-tow way data binding, כלומר, כל שינוי הנעשה ב-template משפיע ישירות על הקומפוננט ולהפך.
- ה. Typescript – שפה חדשה הדומה לתקן החדש של JavaScript.

1. components – חבילות Angular Material ,Syncfusion ej2 ,SweetAlert2
מעוצבות.

עקרונות התכנות

- א. מבחינה מקצועית – בחירת שפות וטכנולוגיות התואמות את צרכי האפליקציה, התמקדנו בחיפוש שפות וטכנולוגיות מהמתקדמות ביותר, המאפשרות מודולריות ופונקציונליות גבוהה, וכן ברצוננו לרכוש ידע מקצועי מהמובילים בתחום.
- ב. מבחינת מבנה הקוד – בנינו את הפרויקט בצורה מודולרית ביותר ע"י חלוקה לשכבות (DAL, BL, GUI) בכדי להפוך את הקוד לנהיר ודינאמי כך שמפתחים נוספים יוכלו להבין את הקוד בקלות, להוסיף על הפרויקט, ובמידת הצורך לשנות.
- ג. מסכי המערכת נכתבו בצורה פשטנית וברורה ביותר לעין, כך שניתנים לשימוש בקלות, ונותנים חווית משתמש.
- ד. התפיסה הרווחת כיום, לגבי אפליקציות Web היא כמה שיותר קוד בצד ה-client מכיוון שכל פנייה לשרת מכבידה על המערכת וגורמת לחוויית משתמש ירודה. השתדלנו שרוב הנתונים שלא צריכים עיבוד בשרת ימומשו בצד ה-client, ועל כן, השתמשנו בספריית angular כדי לחסוך פניות מיותרות לשרת.

חלופות שפות תכנות:

שפת HTML - שפה בסיסית שלא מאפשרת הצגת נתונים דינאמיים ומתאימה במיוחד לאתרים פשוטים. השפה מאפשרת למתכנתים לקשר בין מגוון שפות תכנות ולייצר קוד מורכב. בוני אתרים רבים משתמשים בשפה כדי לחתוך ולעצב תבניות ולהציג מידע באינטרנט. כמו כן, ניתן להשתמש בשפה כדי ליצור קישורים לנתונים, כדי להציג טקסטים, תמונות, סרטוני וידאו ומוסיקה וכדי לבצע מגוון פעולות נוספות.

שפת Java Script - הפקודות של JS משפיעות על הדפדפן והמחשב ומקלות על המתכנתים לבנות אלבומי תמונות, קישורים ותפריטים שנפתחים. חשוב לציין כי השפה לא מובנת למנועי החיפוש משום שהיא מתקשרת בין הדפדפן והמחשב ולא עם השרתים, דבר שעשוי לפגוע בתהליך הקידום של האתר.

MVC - מקובל לחלק יישום תוכנה למספר שכבות נפרדות: שכבת התצוגה (ממשק משתמש), שכבת התחום העסקי (לעיתים נקראת גם "שכבת הלוגיקה העסקית") ושכבת הגישה לנתונים.

בתבנית MVC שכבת התצוגה מחולקת בנוסף לתצוגה ובקר.

יש המחשיבים את התבנית כתבנית עיצוב, אך בהשוואה לתבניות עיצוב אחרות, MVC עוסקת במבנים בקנה מידה בינוני-גדול ולכן נחשבת גם כתבנית ארכיטקטורה.

מודל - המודל הוא ייצוג מסוים, מוכוון תחום עסקי, של המידע עליו פועל היישום. המודל, למרות הדעה הרווחת, אינו שם אחר לשכבת התחום העסקי והוא נפרד ממנה.

תבנית MVC אינה מזכירה במפורש את שכבת הגישה לנתונים, מכיוון ששכבה זו היא מתחת למודל, או נעטפת על ידו.

תצוגה – תפקידה להמיר את נתוני המודל לייצוג המאפשר למשתמש לבצע פעולות גומלין כלשהי. לרוב מדובר על המרה לממשק למשתמש כלשהו. תבנית MVC משמשת רבות ביישומי Web, בהם התצוגה היא דף HTML והקוד אוסף מידע דינאמי לדף.

בקר - תפקידו לעבד ולהגיב לאירועים המתרחשים בתצוגה, לרוב, כתגובה לפעולה של המשתמש. בעיבוד האירועים, הבקר עשוי לשנות את המידע במודל, באמצעות תפעול שירותים המוגדרים בו. בקרים מורכבים מתבססים לרוב על יישום של תבנית command.

לסיכום:

בחרנו לכתוב את המערכת בטכנולוגיית Angular8 ב- Client side,

Web API - C# ב- Server side, מול בסיס נתונים SQL Server, בטכנולוגיית Entity Framework.

בעיות ופתרון

במהלך הפרויקט נתקלנו במספר אתגרים שהיינו שצריכות להתמודד איתם.

המידע והנתונים שאיתם אנו מנהלות את האתר ואף חלק מהפונקציות מתבססים על API חיצוני. נוצרת מכך בעיה של תלות האתר בגורם חיצוני שלא בשליטתנו, במידה ואנו מתבססות רק על שליפות מה API במקרה של קריסה של ה API האתר שלנו יקרוס גם כן.

לאחר מחשבה, החלטנו על שמירת המידע באופן יעיל ב DB, בכך צמצמנו באופן משמעותי את מספר הקריאות ל API ואת הפונקציונליות שמחייבת חיבור אליו. בכך במקרה של קריסת ה API האתר שלנו ימשיך לפעול ולא יקרוס לחלוטין.

אתגר נוסף שהיינו צריכות להתמודד איתו הוא עבודת הצוות מרחוק. אומנם אנו יודעות להתנהל אחת עם השנייה ואין לנו קושי בהתמודדות עם עבודת הצוות אך המרחק היווה עבורנו אתגר גדול בחלוקת העבודה והסנכרון בין הפרויקטים. את אתגר זה לקחנו כהזדמנות, החלטנו ללמוד להשתמש בכלי לא מוכר- GIT ולעבוד איתו, מה שיפתור לנו את בעיית המרחק ועם זאת ייתן לנו ידע וניסיון בכלי שימושי בשוק העבודה כיום. בהתחלה זה לא היה קל אך צלחנו את זה.

אחת מהדרישות באתר הינה יצירת אתר רספונסיבי. מתחילת כתיבת הפרויקט נתקלנו בבעיות רספונסיביות מכיוון שגדלי המסך שלנו שונים, מה שגרם לנו להבין את חשיבות הרספונסיביות, יצר אצלנו צורך ורצון להשקיע בכך מההתחלה. פתרון בעיה זו ניהלנו באמצעות ספריות עיצוב של Bootstrap ובאמצעות תכונות רספונסיביות עבור כל רכיב שהשתמשנו בו.

אבטחה

השקענו עמל והשקעה רבה בהגנות מידע של המשתמשים באתר. האתר שלנו מאובטח ע"י פרוטוקול SSL המהווה תקן להצפנת ואבטחת האתר לגלישה בטוחה של המשתמשים. על מנת לאבטח את המידע על הנתונים שלנו בSQL ע"י כתיבת connectionStrings בקובץ Web.config, מה שהופך אותו למוגן יותר. כמו כן, דאגנו שהודעות השגיאה של המערכת יהיו ידידותיות למשתמש ולא חושפות מידע פנימי על המערכת שלנו.

שרידות המערכת ותיעוד

במהלך הפרויקט שמנו דגש על אבטחת הנתונים, לשם כך בחרנו להשתמש בטכנולוגית Web API מכיוון שהשימוש בטכנולוגיה זו מחייב שימוש בפרוטוקול HTTPS. פרוטוקול זה הינו בטוח לשימוש ומוכר בעולם התכנות. בשליחת נתונים מצד הלקוח לצד השרת (post) הנתונים מועברים כאובייקט DTO ולא כשרשור גלוי בשורת Url.

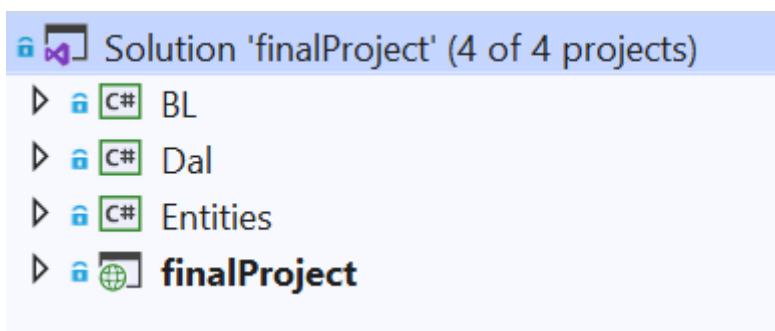
ארכיטקטורת המערכת

את כתיבת הקוד כתבנו במודל השכבות, כאשר לכל שכבה יש תפקיד משלה:

תיאור השכבות:

1. **Dal (Data Access Layer)** שכבה המקשרת את הפרויקט לDB.
2. **BI (Business Layer)** שכבה זו כוללת את הלוגיקה בפרויקט.
3. **Entities** שכבה זו ממירה משתנים מסוג DB לEntities ולהפך.
4. **Web API (Application Programming Interface)** תוכנת שרת המספקת אינפורמציה לתוכנת לקוח.
5. **GUI (Graphical User Interface)** שכבה המספקת ממשק משתמש נוח ונעים לעין, שכבה זו נכתבה ב Angular8.

מראה הSolution של הפרויקט:



תיאור קוד פונקציות ומחלקות

-Client Side

GUI (Graphical User Interface)

את צד הלקוח כתבנו בAngular8.

בניית צד הלקוח הינה חלק מרכזי וחשוב בפרויקט.

את עקרונות השימוש בשפה הכרנו במהלך הלימודים והעמקנו את הידע והכתיבה על ידי שימוש בספריות לעיצוב ולמידה מקוונת של פיצ'רים נוספים בשפה.

הבנייה כוללת שימוש ב-Components, Classes ו-Services.

כל Component מורכב מ:

- קובץ html- מראה הממשק.
- קובץ CSS- עיצוב קובץ html.

- קובץ (Type Script) TS - כתיבת הקוד.

Classes – קבצים המגדירים לנו תכונות לאובייקטים מסוימים.

Services – קבצים לוגיים המיועדים לשמירת נתונים ולניהול תקשורת עם Web API.

דוגמה ל-Component

```

✓ ingredients
  # ingredients.component.css
  <> ingredients.component.html
  TS ingredients.component.spec.ts
  TS ingredients.component.ts
  
```

בקומפוננט זה מוצגת למשתמש רשימת הקניות שלו, באופן ברירת מחדל על פי כל המתכונים ששמר לשבוע הקרוב ובאפשרותו לשנות את ההגדרות על ידי שינוי טווח התאריכים או על ידי סינון לפי מתכונים מתוזמנים. בנוסף על כל מוצר ברשימה המשתמש יכול לסמן האם יש לו צורך בו והאם יופיע ברשימה הסופית. בסיום העריכה המשתמש יכול להדפיס את הרשימה ולייצא אותה.

ingredients.component.css

```
# ingredients.component.css X
src > app > ingredients > # ingredients.component.css > ...
1  *{
2    list-style: none;
3    margin: 0;
4    padding: 0;
5    box-sizing: border-box;
6  }
7  .header {
8    padding: 10px;
9    text-align: center;
10   font-family: "cursive";
11   font-weight: bold;
12   font-size: 42px;
13 }
14 .export {
15   float: right;
16   padding-right: 20px;
17 }
18 .e {
19   background-color: orange;
20   border-radius: 3px;
21   margin: 3px;
22   padding: 5px;
23   cursor: pointer;
24   float: right;
25 }
26 .es {
27   padding: 2.1px 6px;
28 }
29 .icon {
30   font-size: large;
31 }
32 .s {
33   font-size: x-large;
34 }
35 input {
36   margin: 5px;
37   opacity: 0.6;
38   font-size: 16px;
39   font-weight: 300;
40   padding: 10px;
41   border-radius: 2px;
42   border: 0;
43 }
44 .notes-a {
45   color: #5e5873;
46   font-size: 20px;
47   padding: 12px;
48   background-color: #f1f1f1;
49   margin: auto 15px;
50   float: right;
```

```

219 .seperate {
220   height: 8px;
221   background-color: #f5f5f5;
222 }
223 .men a {
224   color: #6e6b7b;
225   padding: 10px 20px;
226   display: flex;
227   letter-spacing: 0.25px;
228 }
229 .men a:hover, .men a:active .l {
230   background-color: rgb(228, 227, 227);
231   text-decoration: underline;
232 }
233 .recipeDiv {
234   display: inline-block;
235 }
236 .l {
237   padding: 0;
238   font-size: 14px;
239   font-weight: 500;
240 }
241 .d {
242   font-size: 12px;
243   font-weight: 200;
244 }
245 menu li {
246   display: inline;
247   border-right: 1px solid #6e6b7b;
248   padding: 0px 10px;
249 }
250 menu li:last-child {
251   border: 0;
252 }
253 .add-p {
254   border: 1px solid #e2efef;
255   margin: auto;
256   padding: 10px;
257 }
258 .notes {
259   padding: 20px 10px;
260 }
261 .notes label {
262   font-size: medium;
263 }
264 @media print {
265   .export, .grid-list-buttons, .recipes-content,
266   display: none;
267 }

```



ingredients.component.html

```
ingredients.component.html M X
src > app > ingredients > ingredients.component.html > div.app-content.content
1 <!-- BEGIN: Recipes Menu-->
2 <div class="main-menu menu-fixed menu-light menu-accordion menu-shadow men" data-scroll-to-active="true">
3   <div class="row">
4     <ul class="col-sm-12">
5       <li class="col-sm-12" *ngFor="let item of viewRecipes;">
6         <div class="col-sm-12 separate"></div>
7         <a class="col-sm-12" routerLink="/site/showDetails/{{item.RecipeId}}" target="_blank">
8           <input class="col-sm-1" id="as{{item.Code}}" type="checkbox" [(ngModel)]="item.isChecked" (change)="createList()" />
9           <div class="recipeDiv">
10             <label class="col-sm-12 l" for="as{{item.Code}}">{{item.RecipeTitle}}</label>
11             <label class="col-sm-12 d">{{item.Date | date:'yyyy-MM-dd'}}</label>
12             
13           </div>
14         </a>
15       </li>
16     </ul>
17   </div>
18 </div>
19 <!-- END: Recipes Menu-->
20 <div class="app-content content">
21   <div class="content-wrapper">
22     <div class="content-body">
23       <div class="row">
24         <h1 class="col-sm-9 header">Ingredients List</h1>
25         <div class="col-sm-3 export">
26           <div class="e" title="Print" (click)="print()"><i class="fa fa-print icon"></i></div>
27         </div>
28         <div class="col-sm-6">
29           <input type="date" [value]="startDate | date:'yyyy-MM-dd'" (input)="startDate=parseDate($event.target.value)"
30             (change)="changeDate()" [min]="minDate | date:'yyyy-MM-dd'">
31           <input type="date" [value]="endDate | date:'yyyy-MM-dd'" (input)="endDate=parseDate($event.target.value)"
32             (change)="changeDate()" [min]="minDate | date:'yyyy-MM-dd'">
33         </div>
34         <div class="col-sm-6 grid-list-buttons">
35           <button id="list" data-view="list-view" class="g-l-b" [ngClass]="(lgridMode)?'active':''"
36             (click)="gridMode = lgridMode"><i class="fa fa-bars"></i>List</button>
37           <button id="grid" data-view="list-view" (click)="gridMode = lgridMode" class="g-l-b "
38             [ngClass]="(gridMode)?'active':''"><i class="fa fa-th-large"></i>Grid</button>
39           <a class="notes-a" href="/site/ingredients/{{startDate}}/{{endDate}}#additional_products"
40             title="Click here to add notes."><i class="fa fa-sticky-note"></i></a>
41         </div>
42       </div>
43       <div class="wrapper">
44         <div class="view_main" [ngClass]="(gridMode)?'grid-mode':'list-mode'">
45           <div class="view_wrap list-view" *ngFor="let item of viewData">
46             <div class="view_item">
47               <input class="checkbox" type="checkbox" [value]="item.code" (change)="ProductCheckbox($event)" />
48               <div class="vi_left">
49                 
50                 
51               </div>
52               <div class="vi_right">
53                 <p class="title">{{item.ProductName}}</p>
54               </div>
55               <div class="vi_right">
56                 <div class="content">
57                   <menu>
58                     <li *ngFor="let unit of item.Units;let i=index">{{unit.Amount}} {{unit.Unit}}</li>
59                   </menu>
60                 </div>
61                 <div class="btn" [value]="item.code" (click)="item.showRecipes = item.showRecipes">View Recipes
62                 <span class="badge badge-pill badge-primary is">{{item.Recipes.length}}</span>
63                 <div class="recipes-content" *ngIf="item.showRecipes">
64                   <a class="row" *ngFor="let r of item.Recipes; index as i" routerLink="/site/showDetails/{{r.RecipeId}}" target="_blank">
65                     
66                     <p class="col-sm-10"> {{r.RecipeTitle}} - {{r.Date | date:'yyyy-MM-dd'}}</p>
67                   </a>
68                 </div>
69               </div>
70             </div>
71           </div>
72           <div class="view_wrap grid-view" *ngFor="let item of viewData">
73             <div class="view_item">
74               <input class="checkbox" type="checkbox" [value]="item.code" (change)="ProductCheckbox($event)" />
75               <div class="vi_left">
76                 
77                 
78               </div>
79               <div class="vi_right">
80                 <p class="title">{{item.ProductName}}</p>
81               </div>
82               <div class="vi_right">
83                 <div class="content">
84                   <menu>
85                     <li *ngFor="let unit of item.Units;let i=index">{{unit.Amount}} {{unit.Unit}}</li>
86                   </menu>
87                 </div>
88                 <div class="btn" [value]="item.code" (click)="item.showRecipes = item.showRecipes">View Recipes
89                 <span class="badge badge-pill badge-primary is">{{item.Recipes.length}}</span>
90                 <div class="recipes-content" *ngIf="item.showRecipes">
91                   <a class="row" *ngFor="let r of item.Recipes; index as i" routerLink="/site/showDetails/{{r.RecipeId}}" target="_blank">
92                     
93                     <p class="col-sm-10"> {{r.RecipeTitle}} - {{r.Date | date:'yyyy-MM-dd'}}</p>
94                   </a>
95                 </div>
96               </div>
97             </div>
98           </div>
99         </div>
100       </div>
101       <div class="col-sm-12 notes">
102         <label for="additional_products">Notes: </label>
103         <textarea id="additional_products" class="col-sm-12 add-p" rows="10"></textarea>
104       </div>
105     </div>
106   </div>
107 </div>
```


ingredients.component.ts

```

TS ingredients.component.ts M X
src > app > ingredients > TS ingredients.component.ts > TS IngredientsComponent
1 import { formatDate } from '@angular/common';
2 import { Component, OnInit } from '@angular/core';
3 import { ActivatedRoute, Router } from '@angular/router';
4 import { addDays } from '@syncfusion/ej2-angular-schedule';
5 import { Product } from '../classes/product';
6 import { RecipesService } from '../services/recipes.service';
7 import { SchedulesService } from '../services/schedules.service';
8
9 @Component({
10   selector: 'app-ingredients',
11   templateUrl: './ingredients.component.html',
12   styleUrls: ['./ingredients.component.css']
13 })
14 export class IngredientsComponent implements OnInit {
15
16   schedulesRecipes = [];
17   viewRecipes = [ (alias) class Product
18                   import Product
19   listPro: Array<Product> = new Array<Product>()
20   viewData = []
21   gridMode = true;
22
23   startDate: Date = new Date(Date.now());
24   endDate: Date = new Date(addDays(this.startDate, 7));
25   minDate: Date = new Date(Date.now());
26
27   constructor(public schedulesService: SchedulesService, public recipesService: RecipesService, public activatedRoute: ActivatedRoute, public router: Router) {
28   }
29
30   ngOnInit(): void {
31     this.activatedRoute.params.subscribe(x => {
32       if (x["startDate"] && x["endDate"]) {
33         if (x["startDate"])
34           this.startDate = x["startDate"]
35         if (x["endDate"])
36           this.endDate = x["endDate"]
37         this.logPage()
38       } else {
39         this.router.navigate(['site/ingredients/' + formatDate(this.startDate, 'yyyy-MM-dd', 'en-US') + '/' + formatDate(this.endDate, 'yyyy-MM-dd', 'en-US')])
40       }
41     })
42   }
43
44   async logPage(): Promise<void> {
45     this.viewRecipes = [];
46     this.schedulesService.GetRecipesByUser(this.startDate, this.endDate).subscribe(
47       (recipeItem: any) => {
48         if (recipeItem.Status) {
49           this.schedulesRecipes = recipeItem.Data;
50           this.schedulesRecipes.forEach(si => {
51             let dup = Object.assign({}, si);
52             dup.isChecked = true;
53             this.viewRecipes.push(dup);
54           })
55         }
56         this.schedulesService.GetProductsByRange(this.startDate, this.endDate).subscribe(
57           (productItem: any) => {
58             if (productItem.Status) this.listPro = productItem.Data
59             this.createList()
60           })
61         });
62   }
63
64   ProductCheckbox(e) {
65     this.viewData.map(x => {
66       if (x.code == e.target.value) { x.checkbox = !x.checkbox; }
67     })
68   }
69
70   changeDate() {
71     this.navigateToDates();
72   }
73
74   parseDate(dateString: string): Date {
75     if (dateString)
76       return new Date(new Date(dateString).setHours(0));
77     return null;
78   }
79
80   navigateToDates() {
81     this.router.navigate(['site/ingredients/' + formatDate(this.startDate, 'yyyy-MM-dd', 'en-US') + '/' + formatDate(this.endDate, 'yyyy-MM-dd', 'en-US')])
82   }
83
84   createList() {
85     let groupedByProductName: Product[] = [];
86     this.listPro.forEach(item => {
87       var productItem = Object.assign({}, item)
88       const recipe = this.viewRecipes.find(x => x.isChecked && x.code == productItem.RecipeUniqueCode);
89       if (recipe) {
90         const existProduct = groupedByProductName.find(x => x.ProductName == productItem.ProductName);
91         const unitText = (productItem.Unit ? productItem.Unit : "");
92         if (!existProduct) {
93           productItem.Recipes = Array(recipe);
94           productItem.Units = Array({ Amount: productItem.Amount, Unit: unitText });
95           groupedByProductName.push(productItem);
96         } else {
97           //check for recipe
98           const currExistsRecipe = existProduct.Recipes.find(x => x["code"] == productItem.RecipeUniqueCode);
99           if (!currExistsRecipe) existProduct.Recipes.push(recipe);
100           //check for unit
101           const currSameUnit = existProduct.Units.find(x => x.Unit == unitText);
102           if (currSameUnit) {
103             currSameUnit.Amount = currSameUnit.Amount + item.Amount;
104           } else {
105             existProduct.Units.push({ Amount: item.Amount, Unit: unitText });
106           }
107         }
108       }
109     })
110     this.viewData = groupedByProductName;
111   }
112
113   print() {
114     window.print()
115   }
116 }

```

Services:

```

✓ services
  TS auth.guard.ts
  TS recipes.service.spec.ts
  TS recipes.service.ts
  TS saved-recipes.service.spec.ts
  TS saved-recipes.service.ts
  TS schedules.service.spec.ts
  TS schedules.service.ts
  TS user.service.spec.ts
  TS user.service.ts

```

recipes.service.ts

```

TS recipes.service.ts X
src > app > services > TS recipes.service.ts > ...
1  import { HttpClient, HttpResponse, HttpHeaders } from '@angular/common/http';
2  import { Injectable } from '@angular/core';
3  import { CookieService } from 'ngx-cookie-service';
4  import { catchError, Observable, throwError } from 'rxjs';
5  import { Recipe } from '../classes/Recipe';
6
7  @Injectable({
8    providedIn: 'root'
9  })
10 export class RecipesService {
11   isEdit: boolean = false
12   url: string = "http://localhost:44376/Api/Recipes/"
13   constructor(private httpClient: HttpClient, private cookies: CookieService) { }
14   AddRecipe(rec: Recipe): Observable<any> {
15     const header = new HttpHeaders({ 'Content-Type': 'application/json' }).set('Authorization', this.cookies.get('Token'))
16     return this.httpClient.put<any>(this.url + "AddRecipe", JSON.stringify(rec), { headers: header }).pipe(catchError(this.handleError))
17   }
18   SearchRecipe(searchWord: string): Observable<any> {
19     return this.httpClient.get<any>(this.url + "SearchRecipe/" + searchWord).pipe(catchError(this.handleError));
20   }
21   GetRecipeById(id: string): Observable<any> {
22     return this.httpClient.get<any>(this.url + "GetRecipeById/" + id).pipe(catchError(this.handleError))
23   }
24   GetSavedRecipe(): Observable<any> {
25     const header = new HttpHeaders().set('Authorization', this.cookies.get('Token'))
26     return this.httpClient.get<any>(this.url + "GetSavedRecipe", { headers: header }).pipe(catchError(this.handleError))
27   }
28   GetRecipeByLocalId(id: number): Observable<any> {
29     return this.httpClient.get<any>(this.url + "GetRecipeByLocalId/" + id).pipe(catchError(this.handleError))
30   }
31   GetRandom(num: number): Observable<any> {
32     return this.httpClient.get<any>(this.url + "GetRandom/" + num).pipe(catchError(this.handleError))
33   }
34   GetPopular(): Observable<any> {
35     console.log("pop")
36     return this.httpClient.get<any>(this.url + "GetPopular").pipe(catchError(this.handleError))
37   }
38   handleError(errorResponse: HttpResponse) {
39     console.log(errorResponse);
40     return throwError(() => new Error('test'));
41   }
42 }

```

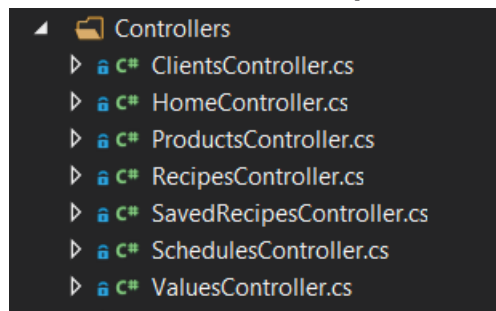
Web API (Application Programming Interface)

בשכבה זו השתמשנו בשירות המאפשר גישה לסביבת הלקוח דרך הדפדפן, שירות זה מאפשר קריאות מסוג POST ומסוג GET.

עבור כל מחלקה בנינו controller המכיל פונקציות שונות מסוגים: GET, PUT, POST ו DELETE.

כל פונקציה בcontroller מתממשקת לסביבת הclient.

מחלקות ה- Controllers:



מחלקת Recipes Controller:

```

6 using BL;
7 using Entities;
8
9 namespace finalProject.Controllers
10 {
11
12     [RoutePrefix("Api/Recipes")]
13     public class RecipesController : ApiController
14     {
15         [HttpPost]
16         [Route("AddRecipe")]
17         public JsonResult<ReturnObject> AddRecipe(RecipesEntities r)
18         {
19             System.Net.Http.Headers.HttpRequestHeaders headers = this.Request.Headers;
20             if (headers.Contains("Authorization"))
21             {
22                 r.Mail = headers.GetValues("Authorization").First();
23             }
24             else
25             {
26                 return Json(new ReturnObject() { Status = false, Error = "Token is necessary" });
27             }
28             try
29             {
30                 r.Date = r.Date.ToLocalTime();
31                 RecipesBl.AddRecipe(r);
32                 var rCode = r.Code;
33                 SchedulesEntities s = new SchedulesEntities() { RecipeCode = rCode, Mail = r.Mail, RecipeTitle = r.RecipeTitle,
34                 switch (r.SchedulingStatus) {
35                     case "w":
36                     }
37                 if (!ProductsController.AddProducts(r.RecipeId, rCode))
38                     return Json(new ReturnObject() { Status = false, Data = "AddProducts failed" });
39                 return Json(new ReturnObject() { Status = true, Data = r.Mail });
40             }
41             catch (Exception ex)
42             {
43                 return Json(new ReturnObject() { Status = false, Error = ex.ToString() });
44             }
45         }
46
47         [HttpGet]
48         [Route("SearchRecipe/{w}")]
49         public JsonResult<ReturnObject> SearchRecipe(string w)
50         {
51             try
52             {
53                 return Json(new ReturnObject() { Status = true, Data = ApiRecipes.SearchRecipe(w) });
54             }
55             catch (Exception ex)
56             {
57                 return Json(new ReturnObject() { Status = false, Error = ex.Message });
58             }
59         }
60
61         [HttpGet]
62         [Route("GetRecipeById/{id}")]
63         public JsonResult<ReturnObject> GetRecipeById(string id)
64         {
65             try
66             {
67                 return Json(new ReturnObject() { Status = true, Data = ApiRecipes.GetRecipeById(id) });
68             }
69             catch (Exception ex)
70             {
71                 return Json(new ReturnObject() { Status = false, Error = ex.Message });
72             }
73         }
74     }
75 }

```

BI (Business Layer)

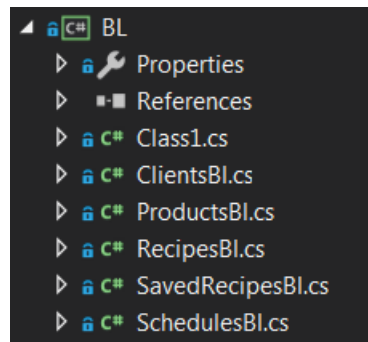
שכבה זו כוללת את הלוגיקה בפרויקט.

בשכבה זו בנינו מחלקת BI עבור כל מחלקה.

מחלקות ה-BI מגשרות בין מחלקות ה-Dal למחלקות ה-Controller.

בנוסף בשכבה זו מתבצע החיבור ל-API, עבור כל פעולת התממשקות עם ה-API. יצרנו פונקציה מתאימה המאפשרת לנו גישה קלה בעזרת פונקציות פשוטות לשליפות הרצויות מה-API.

מחלקות ה-BL:



מחלקת SchedulesBl:

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using Dal;
5 using Entities;
6
7 namespace BL
8 {
9     6 references | List, 18 days ago | 2 authors, 9 changes
10     public class SchedulesBl
11     {
12         3 references | List, 18 days ago | 2 authors, 4 changes
13         public static void AddSchedules(SchedulesEntities s)
14         {
15             SCHEDULES s2 = SchedulesEntities.ConvertToDB(s);
16             DalCode.AddSchedules(s2);
17         }
18         2 references | List, 18 days ago | 2 authors, 6 changes
19         public static List<SchedulesEntities> GetSchedulesByRange(DateTime d1, DateTime d2, string mail)
20         {
21             List<SCHEDULES> listS = DalCode.GetSchedules().Where(x => x.RECIPES!=null && x.RECIPES.MAIL == mail && x.DATE >= d1 && x.DATE <= d2).ToList();
22             var l = SchedulesEntities.ConvertToEntities(listS);
23             return l;
24         }
25         1 reference | List, 18 days ago | 1 author, 2 changes
26         public static void RemoveSchedule(short id, int rec)
27         {
28             short idR = DalCode.GetSchedules().FirstOrDefault(x => x.CODE == id).RECIPE_CODE;
29             switch (rec)
30             {
31                 case 1:
32                     DalCode.RemoveSchedules(id);
33                     break;
34                 case 2:
35                     var l = DalCode.GetSchedules().FindAll(x => x.RECIPE_CODE == idR);
36                     foreach (var item in l)
37                     {
38                         DalCode.RemoveSchedules(item.CODE);
39                     }
40                     break;
41                 case 3:
42                     var l2 = DalCode.GetSchedules().FindAll(x => x.RECIPE_CODE == idR && x.CODE >= id);
43                     foreach (var item in l2)
44                     {
45                         DalCode.RemoveSchedules(item.CODE);
46                     }
47                     break;
48             }
49             bool hasS = DalCode.GetSchedules().Any(x => x.RECIPE_CODE == idR);
50             if (!hasS)
51             {
52                 var listP = DalCode.GetProduct().Where(x => x.RECIPE_CODE == idR);
53                 foreach (var item in listP)
54                 {
55                     DalCode.RemoveProducts(item.CODE);
56                 }
57                 DalCode.RemoveRecipe(idR);
58             }
59         }
60     }
61 }

```

Entities

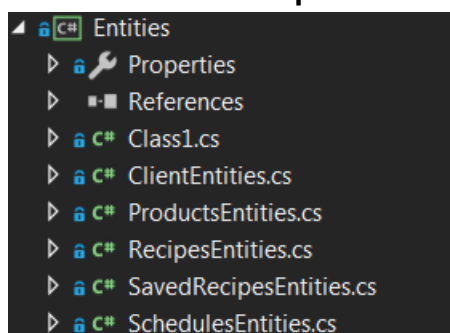
בשכבה זו עבור כל טבלה בDB בנינו מחלקת Entities.

בכל מחלקה קיימות ארבע פונקציות המרה:

1. פונקציה המקבלת משתנה מסוג DB וממירה אותו לסוג Entities.
2. פונקציה המקבלת משתנה מסוג Entities וממירה אותו לסוג DB.
3. פונקציה המקבלת רשימה מסוג DB וממירה אותה לרשימה מסוג Entities.
4. פונקציה המקבלת רשימה מסוג Entities וממירה אותה לרשימה מסוג DB.

המרות אלו מאפשרות לנו "לתקשר" בין צד השרת לצד הלקוח.

מחלקת ה- Entities:



מחלקת ה- ClientsEntities:

```

1 using System.Collections.Generic;
2 using Dal;
3 namespace Entities
4 {
5     11 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
6     public class ClientsEntities
7     {
8         7 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
9         public string Mail { get; set; }
10        4 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
11        public string Password { get; set; }
12
13        //המרה מאובייקט מסוג מסד נתונים לאובייקט מסוג אנטיטיז
14        1 reference | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
15        public static ClientsEntities ConvertToEntities(CLIENTS c)
16        {
17            return new ClientsEntities() { Mail = c.MAIL, Password = c.PASSWORD };
18        }
19
20        //המרה מסוג אנטיטיז לסוג מסד נתונים
21        2 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
22        public static CLIENTS ConvertToDB(ClientsEntities c)
23        {
24            return new CLIENTS() { MAIL = c.Mail, PASSWORD = c.Password };
25        }
26
27        //המרה מסוג רשימת מסד נתונים לרשימת אנטיטיז
28        0 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
29        public static List<ClientsEntities> ConvertToListEntities(List<CLIENTS> listC)
30        {
31            List<ClientsEntities> lc = new List<ClientsEntities>();
32            foreach (var item in listC)
33            {
34                lc.Add(ConvertToEntities(item));
35            }
36            return lc;
37        }
38
39        //המרה מסוג רשימת אנטיטיז לרשימת מסד נתונים
40        0 references | DESKTOP-0N1C103, 113 days ago | 1 author, 1 change
41        public static List<CLIENTS> ConvaertToListDB(List<ClientsEntities> listC)
42        {
43            List<CLIENTS> lc = new List<CLIENTS>();
44            foreach (var item in listC)
45            {
46                lc.Add(ConvertToDB(item));
47            }
48            return lc;
49        }
50    }
51 }

```

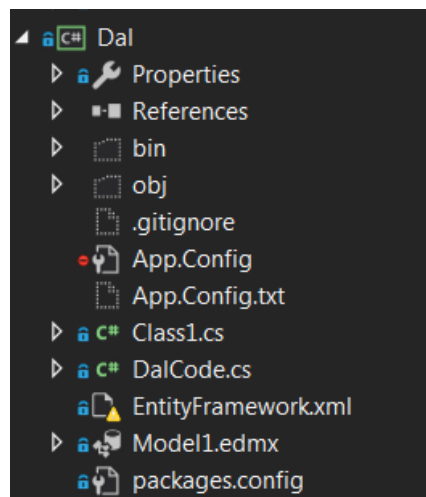
Dal (Data Access Layer)

בשכבה זו אנו מתחברים ל־DataBase (מסד הנתונים).

במחלקת Dal נמצאות כל הטבלאות הקיימות ב־DB כולל קשרי הגומלין שלהן.

טבלאות אלו הינן הקרובות ביותר ל־DataBase ומייצגות אותו, הן המגשרות לבין DB ל־Bl.

שכבת ה-Dal:



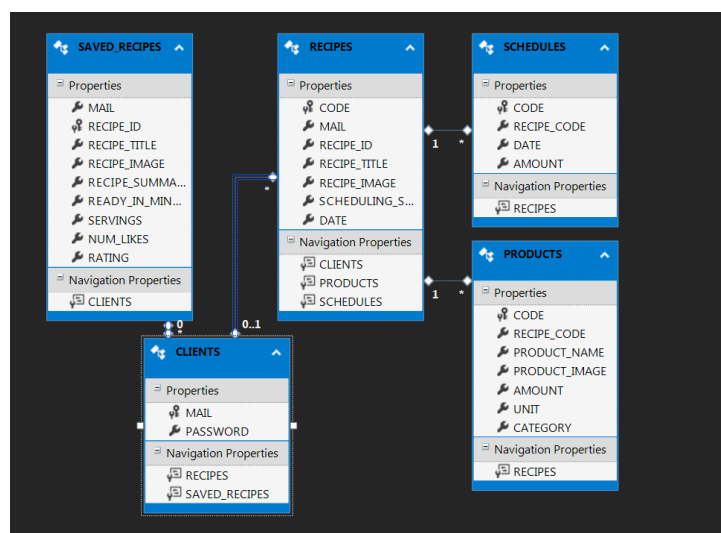
Entity Framework

כלי המאפשר גישה קלה לנתונים הנמצאים ב־DataBase.

כלי זה יוצר לנו מודל לכל ישות וכל מודל מקושר לטבלה ב־DB.

בתרשים הבא ניתן לראות את הטבלאות שנוצרו ב-Entity Framework:

Model1.edmx

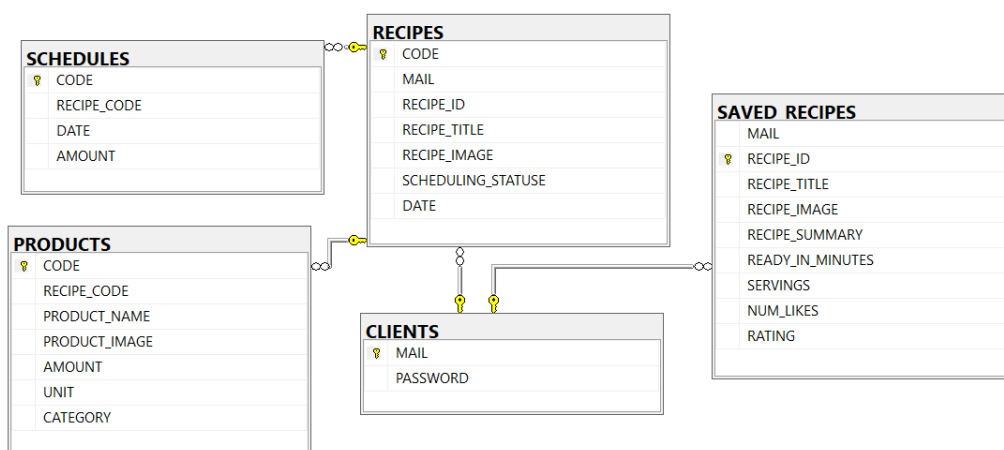


מסד הנתונים

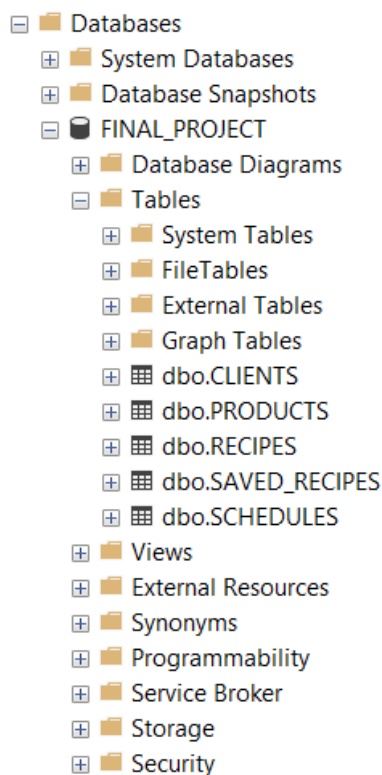
בבניית מסד הנתונים השקענו מחשבה ותכנון על מנת להגיע לתוצאה של מבנה נתונים יעיל ונוח לעבודה.

מסד הנתונים מכיל טבלאות המקושרות ביניהן בקשרי גומלין בכדי לשמור על חוקיות הנתונים ותאימות בין הטבלאות השונות. מסד הנתונים מנומל ומכיל מפתחות ראשיים וזרים שתפקידם למנוע כפילויות ולזרז את שליפת הנתונים.

בתרשים הבא ניתן לראות את מבנה הטבלאות הקיימות בDB ואת קשרי הגומלין שלהן:



מבנה הנתונים נשמר ב-SQL Server:



הגדרת הטבלאות:

Clients

	Column Name	Data Type	Allow Nulls
🔑	MAIL	varchar(50)	<input type="checkbox"/>
	PASSWORD	varchar(20)	<input type="checkbox"/>


	MAIL	PASSWORD
	lylp141@gmail.com	0123456789
	perez@gmail.com	1111

Recipes

	Column Name	Data Type	Allow Nulls
🔑	CODE	smallint	<input type="checkbox"/>
	MAIL	varchar(50)	<input checked="" type="checkbox"/>
	RECIPE_ID	varchar(30)	<input type="checkbox"/>
	RECIPE_TITLE	varchar(150)	<input checked="" type="checkbox"/>
	RECIPE_IMAGE	varchar(150)	<input checked="" type="checkbox"/>
	SCHEDULING_STATUSE	int	<input checked="" type="checkbox"/>
	DATE	datetime	<input type="checkbox"/>


	CODE	MAIL	RECIPE_ID	RECIPE_TIT...	RECIPE_IMAGE	SCHEDULI...	DATE
	119	lylp141@g...	647555	Hotcakes	https://spoonacular.com/recipeImages/647...	1	2022-02-04 ...
	120	lylp141@g...	64694	Gingerbread	https://spoonacular.com/recipeImages/646...	1	2021-12-26 ...
	122	lylp141@g...	716276	Doughnuts	https://spoonacular.com/recipeImages/716...	3	2022-01-04 ...
	123	lylp141@g...	660015	Shrimp Udo...	https://spoonacular.com/recipeImages/660...	3	2022-01-02 ...
	125	lylp141@g...	654495	Pancakes	https://spoonacular.com/recipeImages/654...	1	2022-01-29 ...
	126	lylp141@g...	642638	Fava Crostini	https://spoonacular.com/recipeImages/642...	1	2022-01-29 ...
	130	lylp141@g...	1096165	Sage Gimlet	https://spoonacular.com/recipeImages/109...	1	2022-02-09 ...
	142	lylp141@g...	660128	Simple Ren...	https://spoonacular.com/recipeImages/660...	3	2022-02-15 ...
	146	lylp141@g...	631751	Hot Crab Dip	https://spoonacular.com/recipeImages/631...	1	2022-01-29 ...
	149	lylp141@g...	655570	Penne Con F...	https://spoonacular.com/recipeImages/655...	2	2022-02-24 ...

Schedules

	Column Name	Data Type	Allow Nulls
	CODE	smallint	<input type="checkbox"/>
	RECIPE_CODE	smallint	<input type="checkbox"/>
	DATE	datetime	<input checked="" type="checkbox"/>
	AMOUNT	int	<input checked="" type="checkbox"/>

	CODE	RECIPE_CO...	DATE	AMOUNT
	460	120	2021-12-26 ...	1
	462	122	2022-01-04 ...	1
	466	122	2022-05-04 ...	1
	467	122	2022-06-04 ...	1
	468	122	2022-07-04 ...	1
	469	122	2022-08-04 ...	1
	470	122	2022-09-04 ...	1
	474	123	2022-01-02 ...	1

Products

	Column Name	Data Type	Allow Nulls
	CODE	smallint	<input type="checkbox"/>
	RECIPE_CODE	smallint	<input type="checkbox"/>
	PRODUCT_NAME	varchar(50)	<input type="checkbox"/>
	PRODUCT_IMAGE	varchar(100)	<input checked="" type="checkbox"/>
	AMOUNT	int	<input type="checkbox"/>
	UNIT	varchar(50)	<input type="checkbox"/>
	CATEGORY	smallint	<input checked="" type="checkbox"/>

	CODE	RECIPE_CO...	PRODUCT_...	PRODUCT_I...	AMOUNT	UNIT
	681	119	baking pow...	white-powd...	2	Tbsps
	682	119	egg	egg.png	1	large
	683	119	flour	flour.png	500	g
	684	119	milk	milk.png	488	ml
	685	119	sugar	sugar-in-bo...	1	Tbsp
	686	119	unsalted bu...	butter-slice...	8	Tbsps
	687	119	vanilla extra...	vanilla-extr...	2	tsps
	688	119	yellow food...	food-colori...	0	tsps
	689	120	molasses	molasses.jpg	337	ml
	690	120	salt	salt.jpg	1	Dash
	691	120	eggs	egg.png	2	
	692	120	salad oil	olive-oil.jpg	224	ml
	693	120	sugar	sugar-in-bo...	200	g
	694	120	diet soda	soda-can.jpg	2	tsps
	695	120	water	water.png	355	ml
	696	120	flour	flour.png	250	g
	697	120	ginger	ginger.png	1	tsp
	698	120	cinnamon	cinnamon.jpg	1	tsp
	715	122	flour	flour.png	188	g
	716	122	honey	honey.png	30	ml

SavedRecipes

	Column Name	Data Type	Allow Nulls
	MAIL	varchar(50)	<input checked="" type="checkbox"/>
🔑	RECIPE_ID	varchar(30)	<input type="checkbox"/>
	RECIPE_TITLE	varchar(150)	<input checked="" type="checkbox"/>
	RECIPE_IMAGE	varchar(150)	<input checked="" type="checkbox"/>
	RECIPE_SUMMARY	varchar(MAX)	<input checked="" type="checkbox"/>
	READY_IN_MINUTES	int	<input checked="" type="checkbox"/>
	SERVINGS	int	<input checked="" type="checkbox"/>
	NUM_LIKES	int	<input checked="" type="checkbox"/>
	RATING	float	<input checked="" type="checkbox"/>

MAIL	RECIPE_ID	RECIPE_TITLE	RECIPE_IMAGE	RECIPE_SUMMARY	READY_IN_...	SERVINGS	NUM_LIKES	RATING
lylp141@g...	1697657	Easy Fudge	https://spoonacular.co...	Easy Fudge is a gluten f...	5	50	0	6
lylp141@g...	636461	Burdock Root	https://spoonacular.co...	Need a gluten free, fod...	45	3	1	53
lylp141@g...	639203	Chocolate Soup	https://spoonacular.co...	If you want to add more <b...	45	6	1	35
lylp141@g...	641447	Detox slaw	https://spoonacular.co...	Detox slaw requires roughly...	45	2	2	83
lylp141@g...	641717	Duck Rumaki	https://spoonacular.co...	Duck Rumaki might be just ...	45	24	1	16
lylp141@g...	654435	Pan Seared Salmon	https://spoonacular.co...	Pan Seared Salmon might b...	25	2	2	89

כמו שכתבנו לעיל השתמשנו בטכנולוגיית Entity Framework על מנת לעבוד עם מסד הנתונים.

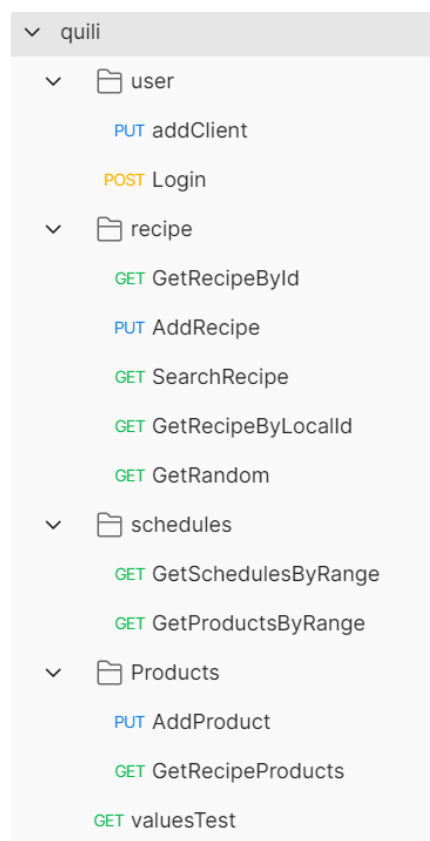
המנגנון של Entity Framework יוצר את Classes על פי החוקיות:

- טבלאות בקשר של יחיד ליחיד- יוצר שני Classes בקשר של הורשה.
- טבלאות בקשר של יחיד לרבים- יוצר Class לכל טבלה, בClass של הרבים יוצר מופע מסוג Class של היחיד ובClass של היחיד יוצר אוסף מסוג Class של הרבים.
- טבלאות בקשר של רבים לרבים- יוצר Class ובו אוספים של Classes המקושרים.

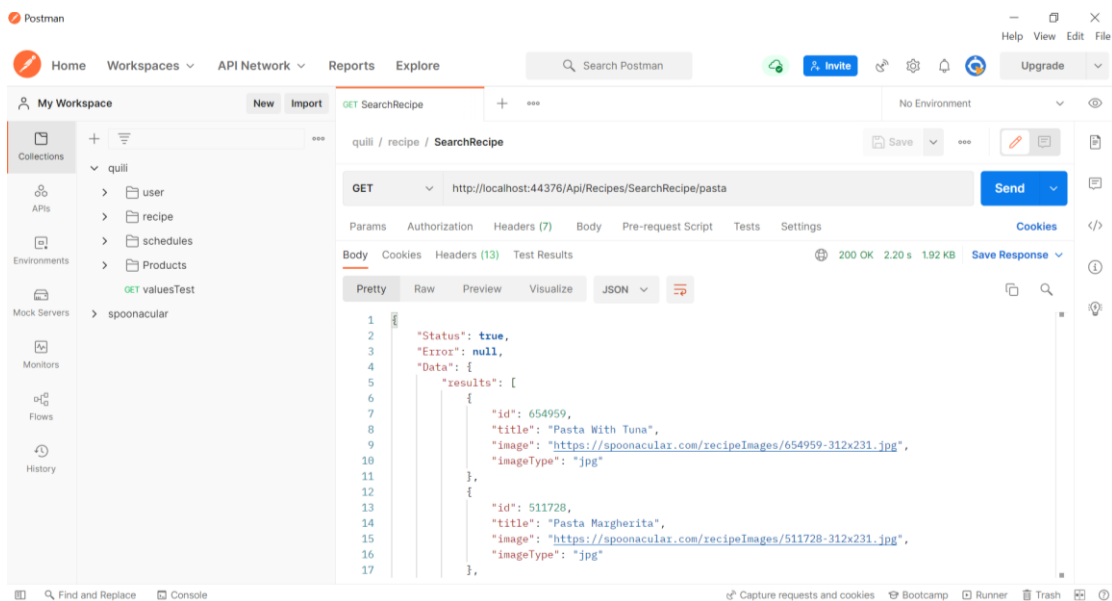
בדיקות תוכנה:

השתמשנו בPostman על מנת לבדוק את שכבת הWeb API, הרצנו כל פונקציה על מנת לעקוב אחרי הפלט שלה ולבדוק את תקינות המידע, הן מהAPI החיצוני והן בדיקת המידע הנשלף והנשמר בDB.

קבצי הבדיקה בPostman:



הרצת הפונקציה SearchRecipe ובדיקת תקינות הפלט.



The screenshot shows the Postman interface with a GET request to `http://localhost:44376/Api/Recipes/SearchRecipe/pasta` successfully executed. The response is a 200 OK status with a body containing a JSON array of recipe data.

```

1  {
2    "Status": true,
3    "Error": null,
4    "Data": [
5      {
6        "id": 654959,
7        "title": "Pasta With Tuna",
8        "image": "https://spoonacular.com/recipeImages/654959-312x231.jpg",
9        "imageType": "jpg"
10     },
11     {
12       "id": 511728,
13       "title": "Pasta Margherita",
14       "image": "https://spoonacular.com/recipeImages/511728-312x231.jpg",
15       "imageType": "jpg"
16     }
17   ]

```

טבלת בדיקות מסכמת

סוג הבדיקה	מס' בדיקה	תיאור קצר	עבר/נכשל	באגים
בדיקת פונקציונאליות	1	בדיקה שהמערכת פועלת כמצופה מימנה	עבר	0
בדיקת שימושיות	2	בדיקת נוחות השימוש וחוויית המשתמש	עבר	0
בדיקת GUI	3	בדיקת הזדהות נכונה וניווט דפים תקין	עבר	0

:Test case

בדיקות פונקציונאליות:

בדיקה מס'	שלב	תוצאה צפויה	תוצאה בפועל	עבר/נכשל	באגים
1	חיפוש מתכון לפי מילת מפתח	הפונקציה תחזיר רשימת מתכונים מתאימים	קיבלתי רשימת מתכונים המתאימים למילת החיפוש	עבר	0
2	טעינת דף הלוח שנה	המתכונים של המשתמש יופיעו	המתכונים הופיעו בהתאם	עבר	0

			בתאריכים המתאימים		
0	עבר	רשימת הרכיבים הופיע כראוי	רשימת הקניות תופיע לפי טווח התאריכים התקין	יצירת רשימת קניות	3

בדיקות שימושיות:

בדיקה מס'	שלב	תוצאה צפויה	תוצאה בפועל	עבר/נכשל	באגים
1	לחיצה על לחצן 	כל המתכונים שנשמרו ע"י המשתמש הנוכחי יופיעו	המתכונים שנשמרו ע"י המשתמש הופיעו	עבר	0
2	לחיצה על לחצן Create Shopping List	רשימת הקניות תופיע לפי המתכונים הרלוונטיים	המצרכים הופיעו לפי המתכונים הקשורים	עבר	0
3	לחיצה על מתכון	דף המתכון המתאים יפתח בחלונית חדשה	דף המתכון נפתח בהתאם	עבר	0

בדיקות GUI:

בדיקה מס'	שלב	תוצאה צפויה	תוצאה בפועל	עבר/נכשל	באגים
-----------	-----	----------------	----------------	----------	-------

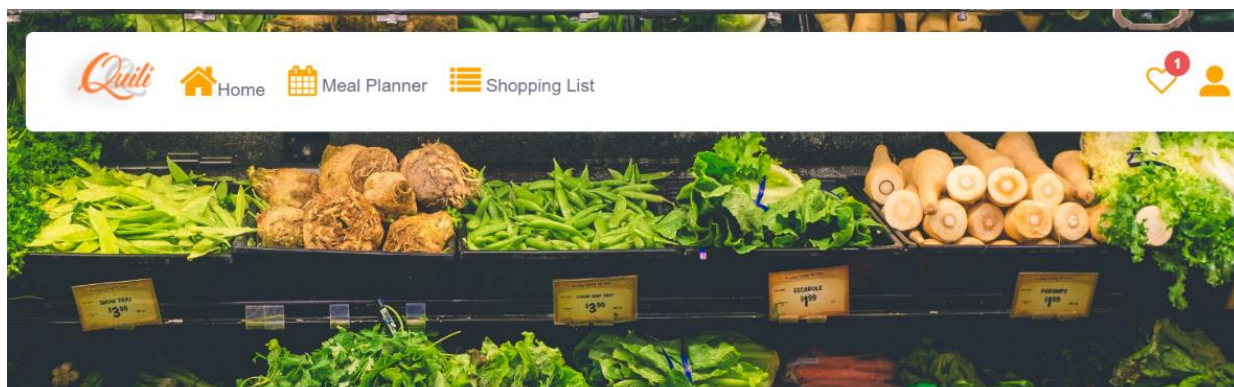
0	עבר	אכן תוצאה מתאימה האתר אפשר כניסה רק למשתמש רשום	כניסה בשל הזדהות תקינה. הודעת שגיאה בעקבות הזדהות לא תקינה	בדיקת תקינות של הזדהות משתמש קיים	1
0	עבר	אכן המערכת אפשרה להירשם רק עם נתונים מתאימים	הרשמה עם מייל תקין וסיסמה מספיק מאובטחת	בדיקת תקינות מידע בהרשמת משתמש חדש	2

מדריך למשתמש

היי ברוכים הבאים ל Quili

אתר המאפשר לך לנהל מתכונים, לתכנן את תפריט ארוחות וליצור רשימת קניות בהתאם בצורה קלה מסודרת ומהירה.

בדף הבית תוכלו למצוא את המתכונים המועדפים ביותר בעבור המשתמשים שלנו.



Name

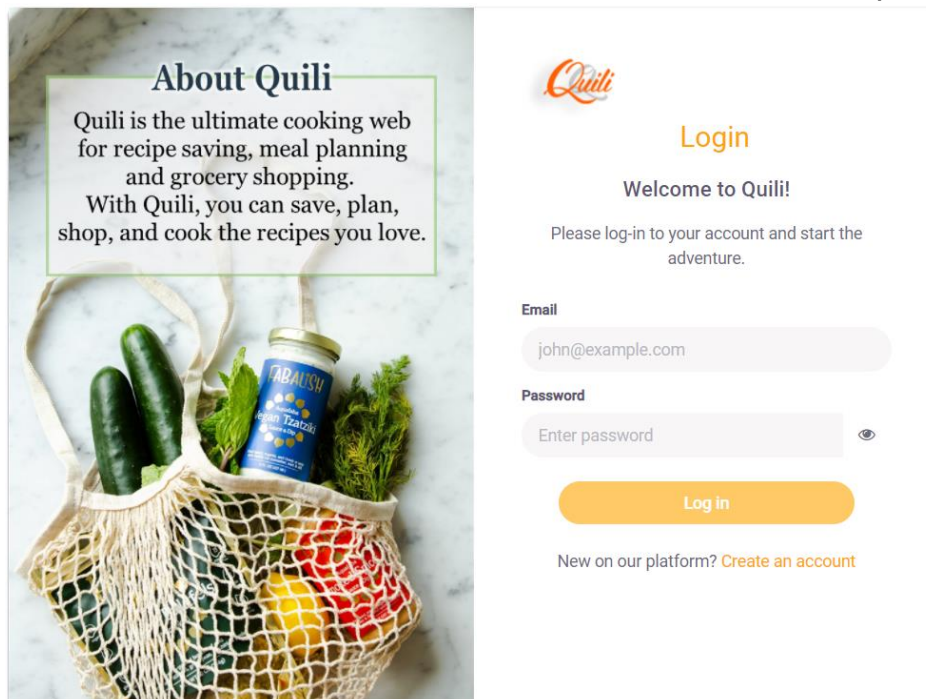
Mail

Message

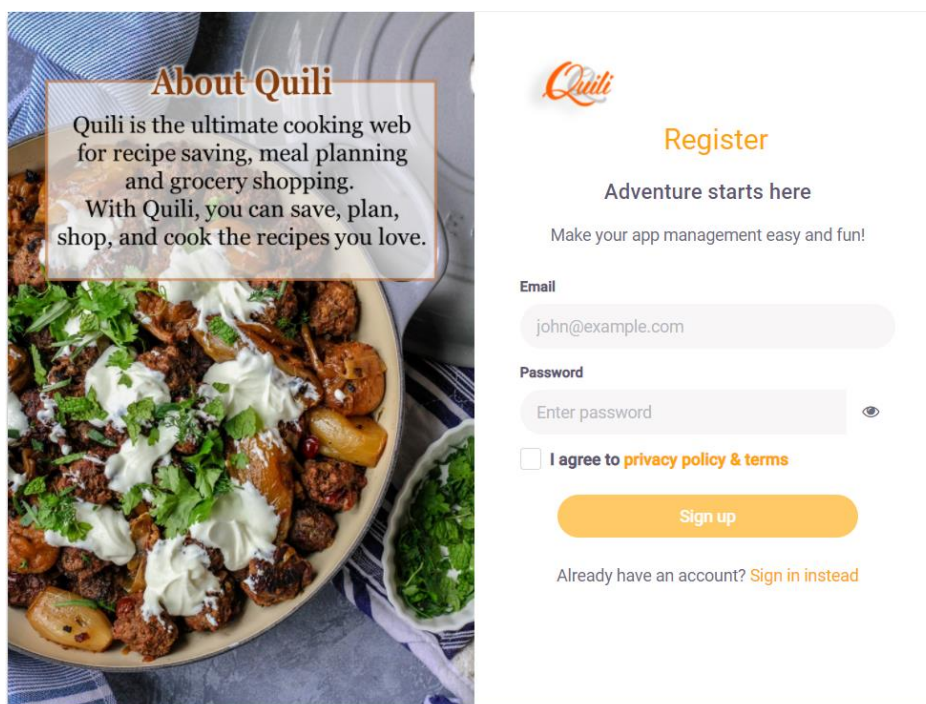
SEND

בכדי להתחיל לנהל את התפריט השבועי שלך, עליך ללחוץ על הלחצן "Meal Planner" בסרגל התפריט.

אם עדיין לא התחברת, באופן אוטומטי תועבד להתחברות לאתר- עליך להתחבר עם הפרטים שלך.



במידה ואינך משתמש רשום עליך לעבור להרשמה ע"י הקשה על הלחצן לחצן " Create an account"



מעולה! עכשיו אתה משתמש רשום באתר.


כעת תוכל להתחיל לנהל את התפריט שלך.

בכניסה לMeal Planner יופיע לך לוח שנה בו תוכל לשמור ולערוך את התפריט שלך בתצוגה חודשית.

Create shopping list

There is no recipes in the menu.

To add a new recipe - press the button above 🍴 or press the required day in the Meal Planner ➔


[Home](#)
[Meal Planner](#)
[Shopping List](#)

<

>

November 2021

▼

TODAY

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
31	Nov 1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	Dec 1	2	3	4

Facebook

Instagram

WhatsApp

Name

Mail

Message


SEND

בלחיצה על כל יום בלוח תפתח חלונית המאפשרת לך לחפש מתכון ולשמור אותו לפי ההגדרות שתבחר.

Create shopping list

There is no recipes in the menu.

To add a new recipe - press the button above 🍴 or press the required day in the Meal Planner ➔


[Home](#)
[Meal Planner](#)
[Shopping List](#)

<

>

March 2023

▼

TODAY

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
26	27			2	3	4
5	6			9	10	11
12	13			16	17	18
19	20			23	24	25
26	27			30	31	Apr 1


×

Add Recipe

Search

Title

Butternut Squash Pizza




Start Date

03/02/2023

Repeat

Never

Save



Name










Mail

Message

SEND

אם ברצונך להסיר מתכון מלוח התכנון שלך, תוכל לעשות זאת ע"י לחיצה על לחצן הִסָּל שליד כל מתכון.

באפשרותך לבחור יום או טווח ימים.


						
13 	14	15	16	17 	18 	19 
20 	21	22	23	24 	25	26

פירוט הבחירה שלך יופיע בצד העמוד בצורה כזו:

Create shopping list

2022-02-27


Burdock Root



Repeat: Weekly Amount: 1

2022-02-28


Easy Fudge



Repeat: Never Amount: 1

על מנת ליצור את רשימת הקניות שלך תוכל ללחוץ על הלחצן "Create shopping list" ותועבר לרשימת הקניות לפי טווח התאריכים הנבחרים. או לעבור בתפריט ל "shopping list" ותיווצר לך רשימת הקניות באופן ברירת מחדל לשבוע הקרוב.

Create shopping list

 Shopping List

יצרת את רשימת הקניות, עכשיו ניתן לערוך אותה.

תוכל לשנות את טווח התאריכים בראש העמוד

03/15/2022



03/22/2022



בצד העמוד מופיעה רשימת המתכונים המרכיבים את רשימת הקניות, באופן ברירת המחדל כולם משפיעים על רשימת הקניות שלך, תוכל לשנות זאת כרצונך.

☒ **Pan Seared Salmon**

2022-03-19



☒ **Oreo Cake**

2022-03-17



☒ **Burdock Root**

2022-03-20

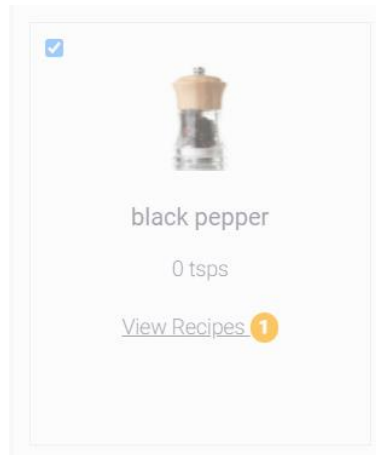


☒ **Duck Rumaki**

2022-03-18



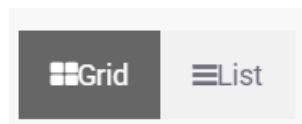
במידה ותרצה להתעלם ממוצר מסוים או שכבר רכשת אותו, תוכל לסמנו ב-V- הוא לא יופיע ברשימה הסופית שלך.




אתה רוצה להוסיף הערה לרשימת הקניות? תוכל להוסיף זאת בסוף הרשימה.

Notes:

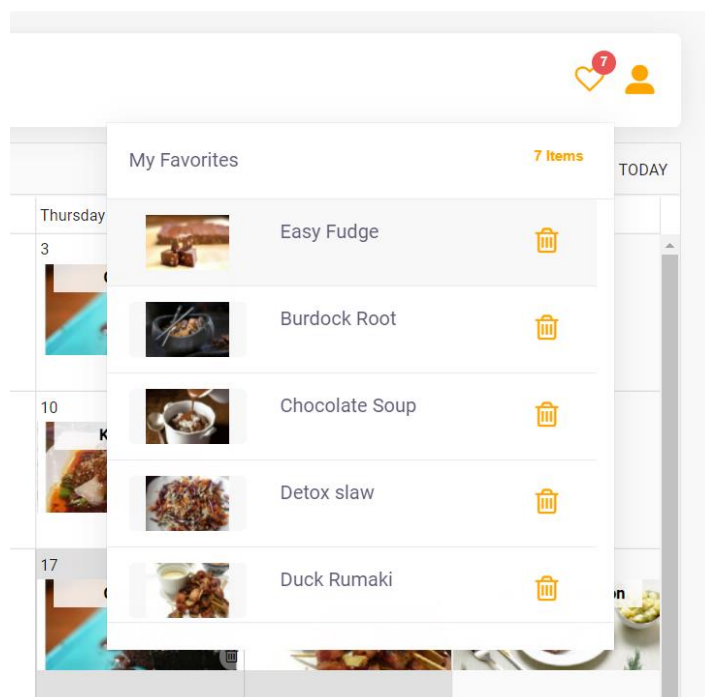
רגע לפני שסיימת, יש לך אפשרות נוספת, תוכל להגדיר את התצוגה של רשימת הקניות בתצוגה הנוחה ביותר עבורך.



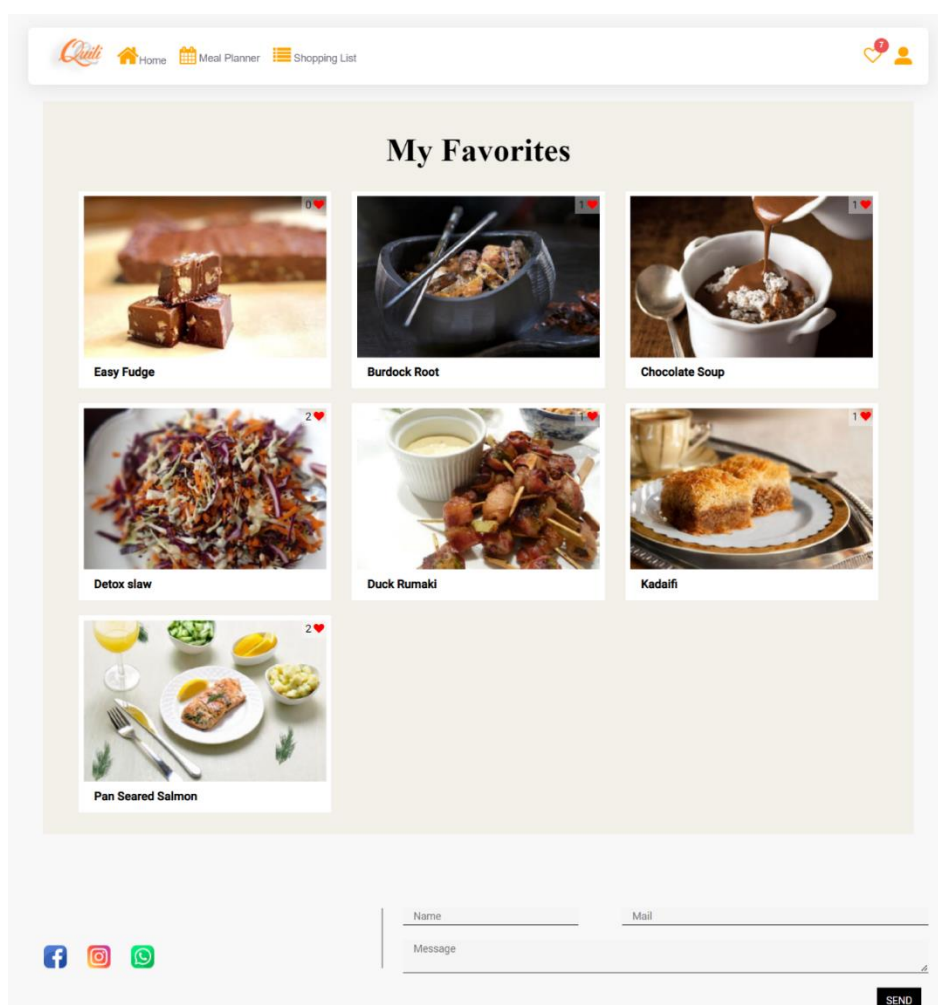
סיימת לערוך את הרשימה כעת תוכל להדפיס אותה ע"י לחיצה על: 

כמשתמש רשום תוכל לנהל רשימת מתכונים שמורים. ניתן לשמור מתכון ע"י לחיצה על לחצן ה-♥ בדף המתכון.


אם ברצונך לצפות ברשימת המתכונים ששמרת, תוכל לעבור על ללחצן ה-♥ בסרגל התפריט כך תקבל הצצה על רשימת המתכונים ששמרת.





בלחיצה על הלחצן תעבור לדף המתכונים השמורים.




בכל שלב ניתן לבחור כל מתכון ולקבל עליו את כל המידע הדרוש, החל מצידוד והרכיבים הנדרשים להכנתו ועד לשלבי ההכנה ועוד.


[Home](#)
[Meal Planner](#)
[Shopping List](#)

Chocolate Soup



🕒 Prep: 45 minutes











👤 Servings: 6 servings

👍 Likes: 1




★ Rating: 35%

If you want to add more **gluten free** recipes to your repertoire, Chocolate Soup might be a recipe you should try. This recipe serves 6. This dessert has **858 calories, 9g of protein, and 74g of fat** per serving. For **\$1.93 per serving**, this recipe **covers 16%** of your daily requirements of vitamins and minerals. This recipe from Foodista requires bittersweet chocolate chips, , vanillin extract, and confectioner's sugar. From preparation to the plate, this recipe takes approximately **approximately 45 minutes**. 1 person has tried and liked this recipe. **Autumn** will be even more special with this recipe. Taking all factors into account, this recipe **earns a spoonacular score of 32%**, which is rather bad. [Chocolate soup](#), [Chocolate Soup For Two](#), and [Johnny Iuzzini's Chocolate Soup](#) are very similar to this recipe.

Ingredients:

3 cups	0.5 cups	1 tsp	5 oz	2 oz	3 tps
					
heavy whipping cream	sugar	vanilla extract	bittersweet chocolate	baking chocolate	chocolate
0.5 cups	2 tps	0.5 cups	0.25 cups		
					
heavy whipping cream	sugar	almonds	bittersweet chocolate chips		

Equipment:

		
sauce pan	whisk	bowl

Instructions:

Step 1

In a medium saucepan over medium heat, whisk together cream, sugar and vanilla. Simmer, stirring several times, for about 40 minutes until reduced by 1/2

Step 2

Stir in chopped chocolates until well blended.

Step 3




Remove from heat and whisk in liqueur.

Step 4

To serve: Whip the cream until frothy then add the confectioners sugar and vanilla. Continue to whip until the cream holds soft peaks. Fold in chopped nuts and chocolate bits.

Step 5

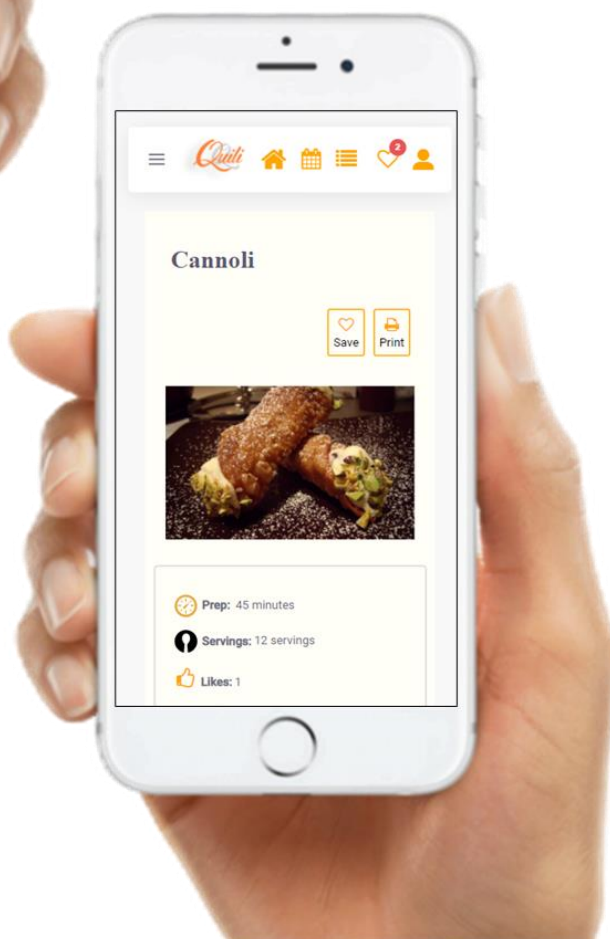
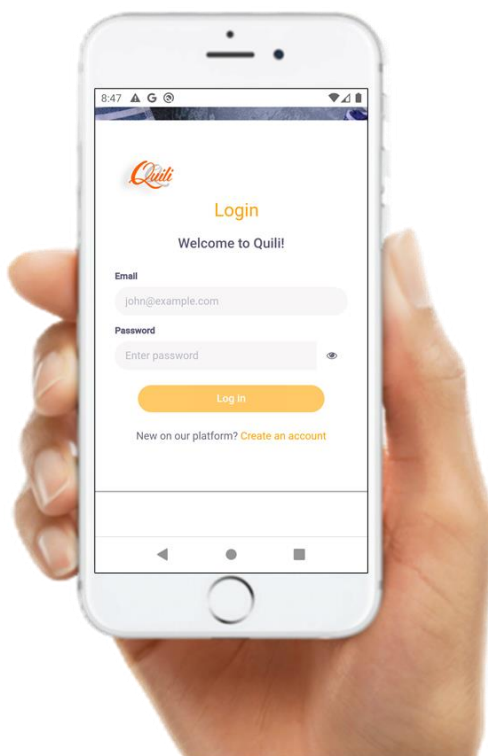
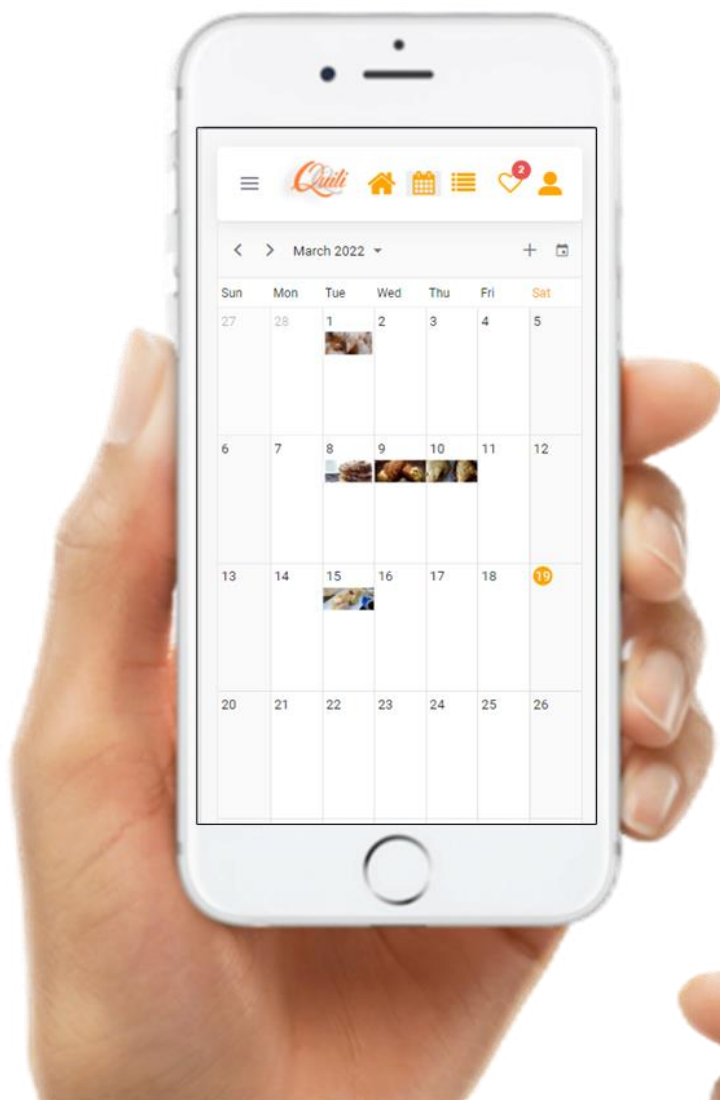
Place a large scoop of the whipped cream in the center of each serving bowl and pour the warm chocolate soup around the whipped cream.

SEND

כמובן, נשמח לשמוע את חוות דעתך על האתר בטופס המופיע בתחתית כל עמוד.

תוכל להיכנס לאתר שלנו בצורה נגישה גם מהנייד.



סיכום

אנו מאוד מרוצות מהתוצאה הסופית שעלתה על הציפיות שלנו, הן מבחינה עיצובית והן מבחינת פיצ'רים תכנותיים הקיימים בה.

במהלך העבודה על הפרויקט הועלו שאלות בנוגע לביצוע בפועל של התוכניות והניתוח הראשוני של הפרויקט, ביצענו שינויים לפי הצורך ושיפרנו את האלגוריתמים בכדי ליצור ממשק מהיר ונוח למשתמש.

לפני שניגשנו לביצוע של כל חלק בפרויקט העמקנו ולמדנו על ספריות חדשות באותו הנושא על מנת לשפר את נראות הפרויקט, למצוא דרכים מגוונות ויעילות יותר לביצוע וכן בכדי ללמוד להשתמש בטכנולוגיות וספריות שונות שלא תמיד היו מוכרות לנו.

הפרויקט היווה עבורנו התנסות משמעותית בכתיבת פרויקט Full Stack, בלמידה עצמית ושילוב ספריות לעיצוב ומידע בפרויקט וכן בשימוש בכלים ואלגוריתמים מתקדמים ומוכרים בשוק העבודה כיום.

אנו מרגישות שרכשנו כלים מעשיים מכתובת הפרויקט ובטוחות שהידע שרכשנו יסיע לנו רבות בשוק העבודה.

רכשנו ידע נרחב בשפת C# וב-Angular8, ידע נרחב ומעמיק במבני נתונים, בשימוש בWeb API ובקריאת מידע מDataBases, למדנו על שימוש בAPI חיצוני, שילובו באתר שלנו בצורה הנכונה ביותר ועל שליפת המידע, שימוש בו ושמירתו בצורה יעילה.

כלי נוסף שהשתמשנו בו היה GitHub, אומנם לא היה קל להתרגל לשימוש בו, אך לא ויתרנו עליו וחוונו את העבודה בו כיעילה ונוחה הרבה יותר ובנוסף הרחבנו ידע וניסיון בשימוש בכלי שימושי ומתקדם זה.

למדנו על התמודדות עם פרויקט בהיקף גדול, למדנו על סדר העבודה בצורה נכונה, על חשיבות הניתוח והמיקוד בשלב הראשוני- תכנון מוקדם, נכון וטוב יחסוך לנו בעיות בעת הפיתוח. וכן על חשיבות הצבת יעדים וחלוקת משימות בצורה מסודרת בכל שלב בתהליך על מנת למזער קונפליקטים.

היו גם אתגרים וחששות, בעיקר מהדברים שהיינו צריכות ללמוד לבד. לקראת השימוש בAPI למתכונים, בדקנו וחקרנו על מגוון רחב של API הקיימים בשוק, התייעצנו רבות והגענו להחלטה שהייתה הנכונה ביותר.

נתקלנו אף בקשיים במהלך הפיתוח כמו הקושי בשימוש בGitHub ולא תמיד הכל הלך חלק, למדנו לנסות שוב ולפעמים לשנות כיון עד שלבסוף הגענו לתוצאה הרצויה.

מקורות מידע

[/https://angular.io](https://angular.io)

[/https://www.w3schools.com](https://www.w3schools.com)

[/https://stackoverflow.com](https://stackoverflow.com)

[/https://github.com](https://github.com)

<https://spoonacular.com/food-api>

<https://ej2.syncfusion.com/home/angular.html>

[/https://sweetalert2.github.io](https://sweetalert2.github.io)

[/https://pixinvent.com/demo/vuexy-vuejs-admin-dashboard-template/landing](https://pixinvent.com/demo/vuexy-vuejs-admin-dashboard-template/landing)

[/https://developer.mozilla.org/en-US](https://developer.mozilla.org/en-US)