

סדנה בתכנות מונחה עצמים - 20586

פרויקט

Eat2Fit



גיא כרמי

301726154

תוכן עניינים

2	תוכן עניינים
4	מסמך אפיון וניתוח
4	ייעוד המערכת
5	דרישות המערכת
6	מרכיבי המערכת
7	משתמשי המערכת
7	תזונאי - מטפל
7	לקוח - מטופל
7	מנהל מערכת - System Administrator
8	דיאגרמות תרחישים
8	משתמשים במערכת
8	השימוש במערכת
9	מסמך תכנון ועיצוב
9	תיאור כללי של המערכת
10	שכבת הנתונים
10	המחלקות
10	המחלקה Customer
11	המחלקה Food
12	המחלקה Meal
13	בסיס הנתונים
14	שכבת הלוגיקה
14	מחלקות השירות - namespace eat2fit.Services
14	המחלקה MongoService
14	המחלקה Base
15	מחלקות ה- ViewModel של אפליקציית האנדרואיד
15	namespace eat2fitApp.ViewModels
15	class MainPageVM : INotifyPropertyChanged
16	class AddMealPageVM : INotifyPropertyChanged
17	class LoginPageVM
18	מחלקות ה- ViewModel של אפליקציית הדסקטופ
18	namespace eat2fitDesktop.ViewModels
18	public class AddCustomerVM
19	public class AddMealVM : INotifyPropertyChanged
20	public class AddFoodVM
21	public class MainPageVM : INotifyPropertyChanged

22	שכבת התצוגה
22	מחלקות ה- View של אפליקציית האנדרואיד
22	namespace eat2fitApp.ViewModels
22	MainPage
23	AddMealPage
24	LogInPage
25	מחלקות ה- View של אפליקציית הדסקטופ
25	namespace eat2fitDesktop.ViewModels
25	MainPage
27	AddCustomerPage
28	AddMealPage
30	AddFoodPage
31	שינויים עתידיים אפשריים
32	דיאגרמת ישויות קשרים
33	דיאגרמת מחלקות
34	הוראות הפעלה ממשק תזונאי
35	חלון הוספת לקוח חדש
37	חלון הוספת ארוחה
40	חלון הוספת מאכל
41	השוואה בין התפריט לבין הצריכה בפועל
42	הוראות הפעלה ממשק לקוח
42	מסך התחברות
43	מסך ראשי
45	חלון הוספת ארוחה
48	יומן האכילה

מסמך אפיון וניתוח

ייעוד המערכת

מערכת זו נועדה לעזור בניהול מערכת היחסים בין המטופלים והמטופלים במקצועות התזונה. כיום אנשים רבים נעזרים בבעלי מקצוע בבחירותיהם התזונתיות, פגישה עם דיאטן להכנת תפריט הכנה לחתונה או להשלמת חוסרים תזונתיים לאחר תוצאות לא טובות של בדיקות דם היא דבר מקובל ונפוץ. אנשים אלו בדרך כלל מקבלים את התפריט שהוכן להם על דף, והמטופלים מחזיקים את המידע על כל מטופל בתיקייה (במגירה או במחשב). התקשורת ביניהם, בדרך כלל מבוססת על הודעות, במייל או בוואטסאפ, צורת תקשורת שעלולה להכביד על בעל המקצוע.

כמו כן, בעולם התזונה חשוב מאוד המעקב אחר המזון הנצרך, בין אם כתוב בתפריט ובין אם לא. כיום מטופלים רבים מבקשים ממטופליהם לרשום במשך מספר ימים את כל מה שהם אוכלים, טרם כתיבת התפריט. הבנת המצב הנוכחי משפיעה רבות על תכנון תפריט מתאים.

המערכת תעזור בניהול כל מערכת היחסים הזו, תאפשר למטופל להזין בנוחות את המזון אותו הוא צורך ותאפשר למטפל לכתוב תפריט ולשמור את כל המידע על כל מטופל באופן מרוכז.

דרישות המערכת

- המערכת תהיה זמינה למטפלים באמצעות תוכנת מחשב ולמטופלים באמצעות אפליקציית סלולר.
- המערכת תכיל כרטיס בסיסי המרכז את פרטי המטופל, מדדים שנלקחו, תוצאות של בדיקות שנעשו.
- המערכת תכיל את התוכנית התזונתית של כל מטופל.
- האפליקציה תאפשר למטופל לנהל יומן תזונה, ולמטפל אפשרות לראות אותו.

מרכיבי המערכת

המערכת תכלול שתי אפליקציות, אפליקציית מחשב למטפל ואפליקציית אנדרואיד למטופל. המערכת תשתמש בבסיס נתונים משותף עבור שתי האפליקציות וכך תשמור על סנכרון נתונים מהיר ביניהן.

במערכת יהיו משתמשים ללקוחות, התחברות הלקוח תהיה באמצעות שם וסיסמה כפי שהוכנסו למערכת על ידי התזונאי בעת יצירת המשתמש.

לכל משתמש יהיו שתי תפריטים (תפריטי מזון- דיאטות), תפריט שנאכל בפועל (יומן אכילה) ותפריט מוצע (כפי שנכתב על ידי התזונאי).

התפריטים יהיו בנויים מארוחות, כאשר ארוחה מורכבת מרשימה של מאכלים, כמות מכל מאכל, ושעה בה הארוחה נאכלה.

משתמשי המערכת

תזונאי - מטפל

המערכת מותקנת לתזונאי באופן אישי, כלומר לכל תזונאי יש מערכת משלו (בשלב זה המערכת מכירה בתזונאי אחד ואין משתמשים לתזונאים).

תזונאי מנהל את השימוש במערכת, ומבצע את הפעולות הבאות:

- פתיחת חשבון ללקוח
- צפייה בפרטי הלקוח
- כתיבת תפריט ללקוח
- הוספת מאכלים חדשים למערכת
- צפייה ביומן האכילה שהלקוח מזין

ממשק התזונאי הוא באפליקציית מחשב הנתמכת במערכת ההפעלה Windows 10.

לקוח - מטופל

לקוח יכול לבצע את הפעולות הבאות:

- צפייה בתפריט שנכתב לו על ידי התזונאי
- צפייה ביומן האכילה שהוא מזין
- הוספת ארוחה ליומן האכילה

ממשק הלקוח הוא באפליקציה לטלפון הנייד הנתמכת במערכת ההפעלה Android.

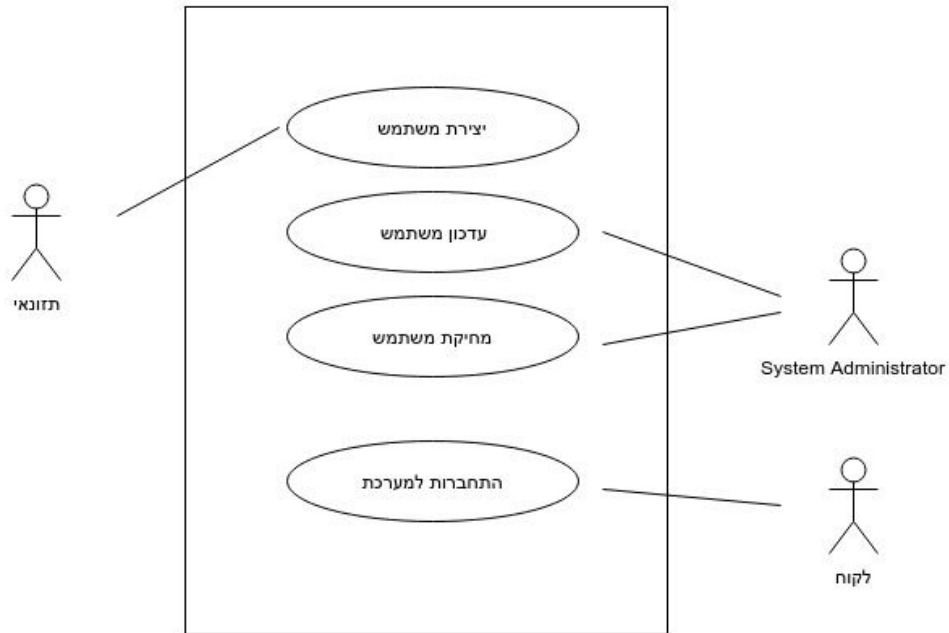
מנהל מערכת - System Administrator

ישנן פעולות שעדיין לא קיים להן ממשק, והן יתבצעו בדרישה ממנהל המערכת:

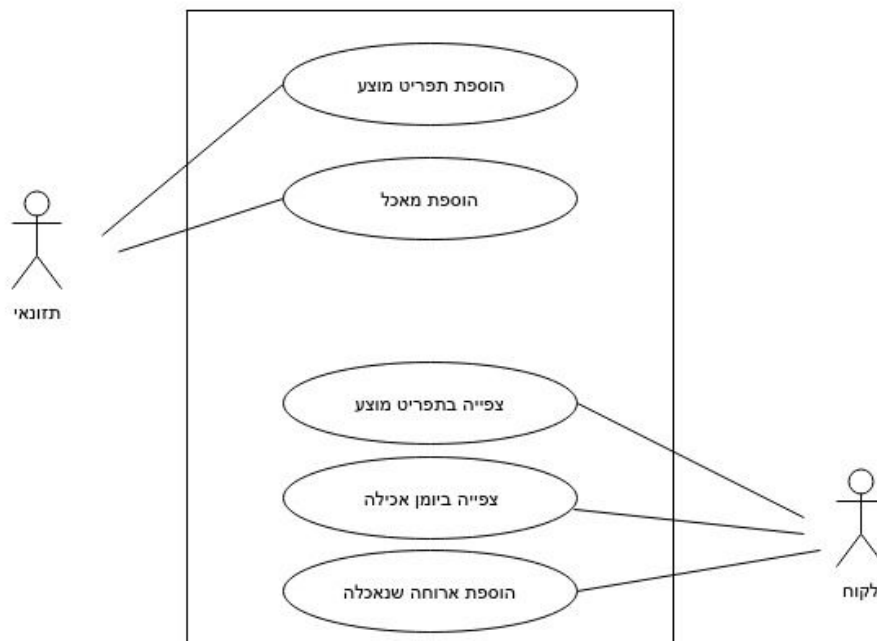
- מחיקת משתמש קיים
- עדכון פרטים למשתמש קיים
- מחיקת מאכל מהמערכת

דיאגרמות תרחישים

משתמשים במערכת



השימוש במערכת



מסמך תכנון ועיצוב

תיאור כללי של המערכת

המערכת נכתבה בשפת C-Sharp ותוך שימוש בטכנולוגיית Mono של חברת Xamarin. טכנולוגייה זו מאפשרת ניצול מירבי של קוד משותף עבור כמה פלטפורמות קצה. בעזרת בחירה זו היה באפשרותי לכתוב ממשק לתזונאי המטפל למערכת הפעלה Windows 10, כך שהתזונאי יוכל להתקין את התוכנה על מחשבו האישי ולעבוד בנוחות אל מול הלקוח. לעומת זאת ללקוח בימינו נוח הרבה יותר להשתמש באפליקצייה לנייד, ולכך ממשק המשתמש שלו נכתב למערכת ההפעלה Android.

דפוס העיצוב הנבחר הינו MVVM (שמו המלא Model View ViewModel), זהו דפוס מומלץ לכתובה עבור Xamarin בסי-שארפ בפרט, ובשפות מבוססות XAML אחרות בכלל, ולכן הבחירה בו הייתה טבעית.

דפוס זה מתבסס על כתיבת הלוגיקה של המחלקות ברכיב ה-Models, הלוגיקה של העמודים ברכיב ה-ViewModel (אשר מכיל אובייקטים של המודלים הנדרשים) ואילו כתיבת התצוגה ברכיב ה-View (אשר מכיל מופע של ה-VM המתאים), כך ניתן להחליף בקלות יחסית הממשק ללא שינוי משמעותי של רכיבי הלוגיקה.

המערכת בנויה משלושה Repositories, הראשון משותף ומכיל Models ו-Services אשר משותפים לכל רכיבי המערכת.

השני מכיל את קוד אפליקציית המחשב והשלישי את קוד אפליקציית האנדרואיד.

שכבת הנתונים

הנתונים נשמרים בשלושה מודלים (Models) המהווים את בסיס המערכת: לקוח, מאכל וארוחה.

המחלקות

המחלקה Customer

מכילה שם

```
private string name;
public string Name { get { return name; } set { name = value; } }
```

סיסמא

```
private string password;
public string Password { get => password; set { password = value; } }
```

מחרוזת פרטים אשר מכילה את הפרטים הנבחרים לצורך הדפסה

```
private string details;
public string Details { get => details; }
```

גיל (ניתן לראות כי הsetter מעדכן גם את מחרוזת הפרטים הנ"ל)

```
private int age;
public int Age { get { return age; } set {
    age = value;
    details = "Age: " + value; } }
```

לכל לקוח רשימת מאכלים מוצעת (כפי שנכתבה על ידי התזונאי)

```
private List<Meal> suggestedDiet;
public List<Meal> SuggestedDiet { get { return suggestedDiet; } set { suggestedDiet = value; }
}
```

ורשימת מאכלים שנאכלו בפועל (כפי שהיא מוזנת על ידי)

```
private List<Meal> eatedDiet;
public List<Meal> EatedDiet { get { return eatedDiet; } set { eatedDiet = value; } }
```

כמו כן ישנן מתודות מתאימות להוספת ארוחה לרשימה הרלוונטית

```
public void AddSuggestedMeal(Meal meal)
public void AddEatedMeal(Meal meal)
```

המחלקה Food

מכילה שם

```
private string name;
```

לשם יש Getter and Setter שניתן לראות בקוד המלא, השם נשמר באותיות קטנות לצרכי

חיפושים והשוואת

```
public string Name
```

ניתן לקבל את השם עם אותיות ראשונות גדולות לצרכי הצגתו

```
public string NameTitle { get => Base.ToTitleCase(name); }
```

קיים שדה עם פרטי המאכל

```
private string details;
```

```
public string Details
```

וקיים שדה עם פרטי המאכל לשימוש בהצגתו כחלק מארוחה (כלומר, עם התייחסות לכמות)

```
private string detailsForMeal;
```

```
public string DetailsForMeal
```

מאכל מכיל את הערך הקלורי שלו ל- 100 גרם

```
private int calories;
```

```
public int Calories
```

ואת הכמות שבה הוא מופיע בארוחה מסוימת

```
private int amount;
```

```
public int Amount
```

המחלקה Meal

מכילה רשימה של מאכלים

```
public List<Food> Foods { get; set; }
```

זמן שבו נאכלה או מתוכננת להיאכל

```
private int time;
```

```
public int Time
```

מחרוזת פרטים המשתמש להצגה

```
private string details;
```

```
public string Details { get => details; }
```

המחלקה משמשת גם עבור ארוחות מתוכננות המוזנות כחלק מתפריט על ידי התזונאי, וגם עבור ארוחות ביומן האכילה המוזנות על ידי הלקוח.

בסיס הנתונים

בסיס הנתונים הנבחר הינו MongoDB. חשוב היה שבסיס הנתונים יהיה בענן ונגיש מכל מקום, בסיס נתונים זה נתמך על ידי מרבית ספקי הענן ומאפשר מעבר קל ביניהם. בתחילה השתמשתי בבסיס הנתונים על גבי פלטפורמת Azure אך בשלב בו נגמרה תקופת הניסיון החינמי היה באפשרותי לעבור בקלות רבה לבסיס נתונים זהה אצל ספק אחר (Mongo Atlas) על ידי החלפת כתובת בסיס הנתונים בלבד. היתרון של בחירה בבסיס נתונים שאינו רלציוני הוא קלות העבודה עם בתכנות מונחה עצמים, המבוסס על מחלקות. באמצעות Bson ניתן לשמור את הנתונים בתצורת Json בבסיס הנתונים, ולאחר מכן באופן קל לטעון את הנתונים לתוך האובייקטים, ולעבוד איתם כאובייקטים במערכת.

בסיס הנתונים מבוסס על Collections שהם המקבילה לטבלאות, ובכל Collection ישנם מסמכים (Documents) שהם המקבילה לשורות בטבלה. ההבדל המשמעותי הוא שהעמודות אינן קבועות וכל מסמך יכול להכיל רכיבים אחרים. כמובן שכדי להכל על השימוש וכדי לשמור על הסדר, רב הדומה על השונה בין המסמכים.

במערכת הזו ישנם 2 collections בלבד, וזאת למרות שישנם 3 מודלים בסיסיים. כפי שניתן לראות בדיאגרמת ישויות הקשרים, למודל ארוחה אין משמעות ללא לקוח שאכל אותו, ולכן לקוח מכיל ארוחות. גמישות בסיס הנתונים מאשרת שינויים, למשל, במידה ויהיה צורך לשמור ארוחות תבניות, כך שהתזונאי יכול להצמיד ארוחה קיימת מתוך תבנית, ללקוח בעת בניית התפריט, ניתן יהיה לעשות זאת בקלות.

שכבת הלוגיקה

הלוגיקה של הדפים (בדיקת קלטים, ביצוע חישובים וכד') מתבצעת בקבצי ה- ViewModel, ואילו הלוגיקה המשותפת (שימוש בבסיס הנתונים וכד') נמצא ב- Services שבפרוייקט הראשי.

מחלקות השירות- namespace eat2fit.Services

המחלקה MongoService

מכילה מתודות אסינכרוניות לעבודה מול בסיס הנתונים

```
public async Task<Customer> GetCustomerIfPasswordVerified(string name, string pass)
```

מתודה זו מחזיר את המשתמש אם השם משתמש והסיסמא נכונים

```
public async Task<List<Customer>> GetAllCustomers()
```

מתודה זו מחזירה רשימה של כל הלקוחות

```
public async Task<List<Food>> GetAllFoods()
```

מתודה זו מחזירה רשימה של כל המאכלים

```
public async Task CreateCustomer(Customer customer)
```

```
public async Task CreateFood(Food food)
```

מתודות אלו מוסיפות לקוח או מאכל ל- Collection המתאים

```
public async Task EditCustomer(Customer customer)
```

מתודה זו מאפשרת החלפה של הלקוח בבסיס הנתונים לאובייקט הלקוח המועבר, וכך מאפשרת עריכה.

המחלקה Base

מכילה מתודות כלליות שיכול להיות בהן שימוש במקומות רבים, כרגע מכילה פונקציה אחת

```
public static string ToTitleCase(this string s) =>
```

```
CultureInfo.InvariantCulture.TextInfo.ToTitleCase(s.ToLower());
```

פונקציה זו מגדירה את מוסכמת המערכת להצגת כותרות (אותיות גדולות בראש מילה)

מחלקות ה- ViewModel של אפליקציית האנדרואיד

namespace eat2fitApp.ViewModels

class MainPageVM : INotifyPropertyChanged

המחלקה ממשמת את הממשק INotifyPropertyChanged וכך מודיעה לרכיבי ה- View שמהו השתנה ויש לעדכן תצוגה. (כל מחלקות ה- VM ממשות ממשק זה, לכן לא אחזור על הסבר זה)

private Customer customer;

המחלקה מכילה מופע של לקוח

private ObservableCollection<Meal> mealList;
public ObservableCollection<Meal> MealList { get => mealList; set { mealList = value;
OnPropertyChanged(); } }

המחלקה מכילה רשימה ObservableCollection כך שניתן להציג את התפריט (רשימת ארוחות)
הרלוונטי

public event PropertyChangedEventHandler PropertyChanged;
void OnPropertyChanged([CallerMemberName] string name = "")

המתודה הנדרשת על מנת לממש את הממשק INotifyPropertyChanged. כאמור.

public void SetCustomer(Customer c)

השמת הלקוח שהתחבר במסך.

public Command MyDietClickedCommand { get; }
void MyDietClicked()

הפקודה והמתודה האחראית על הצגת התפריט כפי שהוזן על ידי הדיאטן

public Command MyEatingLogClickedCommand { get; }
void MyEatingLogClicked()

הפקודה והמתודה האחראית על הצגת יומן האכילה

public Command AddMealClickedCommand { get; }
async void AddMealClicked()

הפקודה והמתודה האחראית על הוספת ארוחה ליומן האכילה

public MainPageVM()

הבנאי מבצע את הקישור בין הפקודות למתודות המתאימות

class AddMealPageVM : INotifyPropertyChanged

מחלקת ה- VM של דף הוספת ארוחה, ממשמת את הממשק INotifyPropertyChanged

private Meal meal = new Meal();

המחלקה מייצרת ארוחה חדשה

Customer customer;

המחלקה מכילה לקוח

public string Amount { get; set; }

קבלת הכמות מהמאכל

public string Hrs { get; set; }

קבלת השעות עבור השעה בה הארוחה נצרכה

public string Mins { get; set; }

קבלת הדקות עבור השעה בה הארוחה נצרכה

private ObservableCollection<Food> foodList;

רשימת המאכלים מהם ניתן לבחור

private ObservableCollection<Food> mealFoodList;

רשימת המאכלים שנבחרו

public object SelectedFood { get; set; }

קבלת המאכל הנבחר

public void SetCustomer(Customer c)

הגדרת המשתמש המבצע את הפעולה

async void RefreshPage()

מתודה האחראית על רענון המידע המוצג בדף

public event PropertyChangedEventHandler PropertyChanged;

void OnPropertyChanged([CallerMemberName] string name = "")

כאמור, המתודה עבור מימוש הממשק INotifyPropertyChanged

public Command AddFoodClickedCommand { get; }

void AddFoodClicked()

הפקודה והמתודה האחראית על הוספת מאכל

public Command FinishMealClickedCommand { get; }

async void FinishMealClicked()

הפקודה והמתודה האחראית על סיום יצירת הארוחה

```
public AddMealPageVM()
```

בנאי המחלקה מרענן את תצוגת הדף בעת טעינתו ומבצע את הקישור בין הפקודות למתודות המתאימות

```
class LoginPageVM
```

מחלקת ה- VM עבור דף ההתחברות, זוהי מחלקה פשוטה למדי

```
public string Name { get; set; }
```

קבלת השם

```
public string Password { get; set; }
```

קבלת הסיסמה

```
public Command OnConnectClickedCommand { get; }
```

```
async void OnConnectClicked()
```

הפקודה והמתודה לכפתור ההתחברות, מבצעת בדיקה שהשם והסיסמה מתאימים אל מול מסד הנתונים

```
public LoginPageVM()
```

הבנאי מכיל בדיקה שהתקשורת עם מסד הנתונים הושלמה ולאחר מכן מעביר את הכפתור להיות לחיצה, כלומר מאפשר התחברות. כמו כן מבצע את הקישור בין הפקודה למתודה המתאימה

מחלקות ה- ViewModel של אפליקציית הדסקטופ

namespace eat2fitDesktop.ViewModels

public class AddCustomerVM

מחלקת ה- VM עבור עמוד הוספת לקוח חדש

Customer customer = new Customer();

יצירת אובייקט לקוח חדש

public string Name { get; set; }

קבלת השם

public string Age { get; set; }

קבלת הגיל

public string Password { get; set; }

קבלת הסיסמה

public Command OnAddCustomerClickedCommand { get; }

async void OnAddCustomerClicked()

הפקודה והמתודה עבור הוספת משתמש. המתודה מוודאת כי כל הפרטים הנדרשים הוזנו

public AddCustomerVM()

הבנאי מבצע את הקישור בין הפקודה למתודה המתאימה

```
public class AddMealVM : INotifyPropertyChanged
```

מחלקת ה-VM עבור עמוד הוספת ארוחה. המחלקה מממשת את הממשק
INotifyPropertyChanged

```
Meal meal = new Meal();
```

המחלקה מייצרת אובייקט ארוחה חדש

```
MongoService mongoService = new MongoService();
```

המחלקה משתמש במחלקת השירות לעבודה עם בסיס הנתונים

```
public ObservableCollection<Food> Foods
```

המחלקה מכילה רשימה ניתנת להצגה של מאכלים ממסד הנתונים

```
public ObservableCollection<Food> ThisMealFoods
```

המחלקה מכילה רשימה ניתנת להצגה של המאכלים שנבחרו עבור ארוחה זו

```
private Customer Customer;
```

המחלקה מכילה את אובייקט הלקוח עבורו מיועדת הארוחה

```
public string Hrs { get; set; }
```

קבלת שעות השעה לה הארוחה מתוכננת

```
public string Mins { get; set; }
```

קבלת דקות השעה לה הארוחה מתוכננת

```
private string search;
```

מאשר חיפוש מאכלים מהרשימה וסינון על פי החיפוש

```
public string Amount { get; set; }
```

קבלת הכמות מהמאכל

```
public object SelectedFood { get; set; }
```

קבלת המאכל הנבחר מהרשימה

```
public event PropertyChangedEventHandler PropertyChanged;
```

```
void OnPropertyChanged([CallerMemberName] string name = "")
```

המתודה הנדרשת עבור מימוש הממשק INotifyPropertyChanged

```
public Command OnNewFoodClickedCommand { get; }
```

```
async void OnNewFoodClicked()
```

הפקודה והמתודה עבור יצירת מאכל חדש

```
public Command OnAddFoodToMealClickedCommand { get; }  
async void OnAddFoodToMealClicked()
```

הפקודה והמתודה עבור הוספת מאכל לארוחה

```
public AddMealVM()
```

בנאי המחלקה מכיל את הקישור בין הפקודות למתודות המתאימות

```
public class AddFoodVM
```

מחלקת ה- VM עבור דף הוספת מאכל

```
Food food = new Food();
```

מייצרת אובייקט מאכל

```
public string Calories { get; set; }
```

קבלת הקלוריות

```
public string Name { get; set; }
```

קבלת שם המאכל

```
public Command OnAddFoodClickedCommand { get; }
```

```
async void OnAddFoodClicked()
```

הפקודה והמתודה להוספת מאכל

```
public AddFoodVM()
```

הבנאי מבצע את הקישור בין הפקודה למתודה המתאימה

public class MainPageVM : INotifyPropertyChanged

מחלקת ה-VM עבור העמוד הראשי בממשק התזונאי. המחלקה מממשת את הממשק
INotifyPropertyChanged

private Customer selectedCustomer;

המחלקה מכילה אובייקט לקוח אליו היא מתייחסת, המידע והתפריטים של לקוח זה יוצגו

MongoService mongoService = new MongoService();

המחלקה מכילה אובייקט של מחלקת השירות לתקשורת עם בסיס הנתונים

private ObservableCollection<Customer> customers = new

ObservableCollection<Customer>();

המחלקה מכילה רשימה של כל הלקוחות, כך שהתזונאי יכול לבחור מביניהם

private ObservableCollection<Meal> suggestedDiet = new ObservableCollection<Meal>();

המחלקה מכילה את רשימת הארוחות שהוא התפריט המוצע

private ObservableCollection<Meal> eatedDiet = new ObservableCollection<Meal>();

המחלקה מכילה את רשימת הארוחות שהוא יומן האכילה של הלקוח

public event PropertyChangedEventHandler PropertyChanged;

void OnPropertyChanged([CallerMemberName] string name = "")

המתודה הנדרשת עבור מימוש הממשק INotifyPropertyChanged

public async void RefreshMainPage()

מתודה האחראית על רענון המידע המוצג בדף

void CustomerChanged()

מתודה האחראית על עדכון המידע הרלוונטי בעת שינוי הלקוח הנבחר אליו יתייחס

public Command OnNewCustomerClickedCommand { get; }

async void OnNewCustomerClicked()

הפקודה והמתודה עבור הוספת לקוח חדש

public Command OnAddMealClickedCommand { get; }

async void OnAddMealClicked()

הפקודה והמתודה עבור הוספת ארוחה

public MainPageVM()

הבנאי מבצע את הקישור בין הפקודות למתודות המתאימות

שכבת התצוגה

שכבת התצוגה מכילה את רכיבי ה-View, רכיבי התצוגה כתובים ב-Xaml ומגדירים את מיקום האובייקטים על המסך. כמו כן לכל דף יש קובץ בסיומת .xaml.cs שתוכנו מינימלי ולכן לא אציגו פה, ובו מתבצע הקישור בין קובץ ה-xaml לבין מחלקת ה-VM הרלוונטית. אציג את עיקרי קבצי ה-xaml מהם ניתן להבין את מבנה הדפים

מחלקות ה-View של אפליקציית האנדרואיד

namespace eat2fitApp. ViewModels

MainPage

זהו הדף הראשי אותו רואה הלקוח לאחר ההתחברות

```
<StackLayout Spacing="20" Padding="20">
```

```
<StackLayout Orientation="Horizontal">
```

בראש הדף ישנם כפתורי המעבר בין התפריט ליומן האכילה

```
<Button Text="My Diet" Command="{Binding MyDietClickedCommand}"/>
```

```
<Button Text="My Eating Log" Command="{Binding
```

```
MyEatingLogClickedCommand}"/>
```

```
</StackLayout>
```

בהמשך הדף ישנה רשימת הארוחות (שנבחרה בכפתור)

```
<ListView ItemsSource="{Binding MealList}">
```

```
<ListView.ItemTemplate>
```

```
<DataTemplate>
```

```
<TextCell Text="{Binding Name}" Detail="{Binding Details}"/>
```

כל אובייקט ברשימה מכיל את שם הארוחה (שהוא זמן הארוחה) ואת פרטי הארוחה

```
</DataTemplate>
```

```
</ListView.ItemTemplate>
```

```
</ListView>
```

```
<Button Text="Add Meal" Command="{Binding AddMealClickedCommand}"/>
```

בתחתית העמוד ישנו כפתור הוספת ארוחה ליומן האכילה

```
</StackLayout>
```

AddMealPage

דף הוספת ארוחה ליומן האכילה

```
<StackLayout Orientation="Horizontal">
```

בראש הדף ניתן להזין את השעה

```
<Label Text="Time: "/>
```

```
<Entry Text="{Binding Hrs}" HorizontalOptions="FillAndExpand"/>
```

```
<Label Text=":"/>
```

```
<Entry Text="{Binding Mins}" HorizontalOptions="FillAndExpand"/>
```

```
</StackLayout>
```

לאחר מכן לבחור מאכלים מתוך רשימה

```
<Picker Title="Select Food" ItemsSource="{Binding FoodList}"
```

```
SelectedItem="{Binding SelectedFood}" ItemDisplayBinding="{Binding Name}"/>
```

```
<StackLayout Orientation="Horizontal">
```

להזין את הכמות מהמאכל

```
<Label Text="Amount: "/>
```

```
<Entry Text="{Binding Amount}" HorizontalOptions="FillAndExpand"/>
```

```
</StackLayout>
```

כפתור הוספת מאכל

```
<Button Text="Add Food" HorizontalOptions="Fill" Command="{Binding
```

```
AddFoodClickedCommand}"/>
```

```
<ListView ItemsSource="{Binding MealFoodList}">
```

```
<ListView.ItemTemplate>
```

```
<DataTemplate>
```

תצוגה של רשימת המאכלים שתרכיב את הארוחה תוך כדי יצירתה

```
<TextCell Text="{Binding Name}" Detail="{Binding DetailsForMeal}"/>
```

```
</DataTemplate>
```

```
</ListView.ItemTemplate>
```

```
</ListView>
```

כפתור בתחתית הדף לסיום הפעולה, הוספת הארוחה ליומן האכילה

```
<Button Text="Finish Meal" Command="{Binding FinishMealClickedCommand}"/>
```

LoginPage

זהו דף ההתחברות, מבנהו פשוט למדי.

<StackLayout>

<StackLayout Orientation="Horizontal">

בראש הדף הזנת שם ולאחריו סיסמה

<Label Text="Name: "/>

<Entry Text="{Binding Name}" HorizontalOptions="FillAndExpand"/>

</StackLayout>

<StackLayout Orientation="Horizontal">

<Label Text="Password: "/>

<Entry Text="{Binding Password}" IsPassword="True"

HorizontalOptions="FillAndExpand"/>

</StackLayout>

לאחריהם כפתור ההתחברות

<Button Text="Connect" Command="{Binding OnConnectClickedCommand}"/>

</StackLayout>

מחלקות ה-View של אפליקציית הדסקטופ

namespace eat2fitDesktop.ViewModels

MainPage

הדף הראשי המוצג לתזונאי העת פתיחת המערכת

```
<StackLayout Spacing="30" Padding="30">
    <StackLayout Orientation="Horizontal" >
        בראש הדף ישנה אפשרות בחירה בין הלקוחות הקיימים ולידו כפתור הוספת לקוח חדש
        <Label Text="Change Customer: " HorizontalOptions="StartAndExpand"/>
        <Picker SelectedItem="{Binding SelectedCustomer}" ItemsSource="{Binding
Customers}" WidthRequest="800"/>
        <Button Text="New Customer" Command="{Binding
OnNewCustomerClickedCommand}"/>
    </StackLayout>
    <StackLayout Orientation="Horizontal" >
        בהמשך העמוד מוצגים פרטי הלקוח הנבחר
        <Label Text="Selected Customer: "/>
        <Label Text="{Binding SelectedCustomerString}" Style="{DynamicResource
TitleStyle}"/>
    </StackLayout>
    <Grid Padding="20" RowSpacing="20" ColumnSpacing="20">
        חלקו המרכזי של העמוד מורכב מ- Grid (צמצמתי שורות הגדרות שאינן מהותיות, ניתן לראות
        את הקוד המלא).
        בחלקו השמאלי יומן האכילה של הלקוח
        <Label Text="Actual Costumer Intake" Grid.Row="0" Grid.Column="0"/>
        ובחלקו הימני התפריט המוצע ללקוח
        <Label Text="Suggested Diet For Customer" Grid.Row="0" Grid.Column="1"/>
        <ListView ItemsSource="{Binding EatedDiet}" Grid.Row="1" Grid.Column="0" >
            <ListView.ItemTemplate>
                <DataTemplate>
```

```
<TextCell Text="{Binding Name}" Detail="{Binding Details}"/>
</DataTemplate>
</ListView.ItemTemplate>
</ListView>
<ListView ItemsSource="{Binding SuggestedDiet}" Grid.Row="1"
Grid.Column="1">
  <ListView.ItemTemplate>
    <DataTemplate>
      <TextCell Text="{Binding Name}" Detail="{Binding Details}"/>
    </DataTemplate>
  </ListView.ItemTemplate>
</ListView>


בתחתית העמוד כפתור הוספת ארוחה לתפריט המוצע


<Button Text="Add Meal" Grid.Row="2" Grid.Column="1" Command="{Binding
OnAddMealClickedCommand}"/>
</Grid>
</StackLayout>
```

AddCustomerPage

דף הוספת לקוח הינו דף פשוט למדי

```
<StackLayout Spacing="30" Padding="30">  
  <StackLayout Orientation="Horizontal">
```

שדה הכנסת שם הלקוח

```
    <Label Text="Name: "/>  
    <Entry Text="{Binding Name}" WidthRequest="500"/>  
  </StackLayout>  
  <StackLayout Orientation="Horizontal">
```

שדה הכנסת סיסמה

```
    <Label Text="Password: "/>  
    <Entry Text="{Binding Password}" WidthRequest="500" IsPassword="True"/>  
  </StackLayout>  
  <StackLayout Orientation="Horizontal">
```

שדה הכנסת גיל

```
    <Label Text="Age: "/>  
    <Entry Text="{Binding Age}" WidthRequest="500"/>  
  </StackLayout>
```

כפתור ליצירת הלקוח

```
    <Button Text="Add" Command="{Binding OnAddCustomerClickedCommand}"  
HorizontalOptions="Center"/>  
  </StackLayout>
```

AddMealPage

זהו דף הוספת ארוחה לתפריט הלקוח

```
<StackLayout Spacing="30" Padding="30">
  <StackLayout Orientation="Horizontal" >
```

בראש העמוד שעת הארוחה

```
    <Label Text="Time: " />
    <Entry Text="{Binding Hrs}" />
    <Label Text=":" />
    <Entry Text="{Binding Mins}" />
  </StackLayout>
```

```
<Grid Padding="20" RowSpacing="20" ColumnSpacing="20">
```

חלקו המרכזי של העמוד בתצורת Grid

```
<Label Text="Foods:" Grid.Column="0" Grid.Row="0" />
```

בחלקו השמאלי רשימת המאכלים האפשריים

```
<Label Text="In This Meal:" Grid.Column="2" Grid.Row="0" />
```

בחלקו הימני רשימת המאכלים שמרכיבים את הארוחה

```
<StackLayout Orientation="Horizontal" Grid.Column="0" Grid.Row="1">
```

ישנה אפשרות לחפש מאכל (בחלק השמאלי) להוספה

```
  <Label Text="Search" />
  <Entry Text="{Binding Search}" HorizontalOptions="CenterAndExpand"
WidthRequest="300" />
```

```
</StackLayout>
```

```
<ListView ItemsSource="{Binding Foods}" SelectedItem="{Binding SelectedFood}"
Grid.Column="0" Grid.Row="2">
```

```
  <ListView.ItemTemplate>
```

```
    <DataTemplate>
```

```
      <TextCell Text="{Binding NameTitle}" Detail="{Binding Details}" />
```

```
    </DataTemplate>
```

```
  </ListView.ItemTemplate>
```

```
</ListView>
```

```

<ListView ItemsSource="{Binding ThisMealFoods}" Grid.Column="2"
Grid.Row="2">
    <ListView.ItemTemplate>
        <DataTemplate>
            <TextCell Text="{Binding Name}" Detail="{Binding DetailsForMeal}"/>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
<StackLayout Grid.Column="1" Grid.Row="2">
    במרכז מוזנת הכמות של המזון הנבחר להוספה
    <Label Text="Amount"/>
    <Entry Text="{Binding Amount}"/>
    <Button Text="->" VerticalOptions="Center" Command="{Binding
OnAddFoodToMealClickedCommand}"/>
</StackLayout>
</Grid>
<StackLayout Orientation="Horizontal">
    בתחתית העמוד ישנם כפתורים ליצירה מאכל חדש (הוספת מאכל למאגר, מאכל שלאחר מכן
    יופיע גם ללקוחות), ולסיום הפעולה והוספת הארוחה לתפריט הלקוח
    <Button Text="New Food" HorizontalOptions="Start" Command="{Binding
OnNewFoodClickedCommand}"/>
    <Button Text="Create Meal" HorizontalOptions="End" Command="{Binding
OnCreateMealClickedCommand}"/>
</StackLayout>
</StackLayout>

```

AddFoodPage

דף להוספת מאכל למערכת. דף פשוט למדי

```
<StackLayout Padding="30" Spacing="30">
```

```
<StackLayout Orientation="Horizontal">
```

```
<Label Text="Food Name"/>
```

הזנת שם המאכל

```
<Entry Text="{Binding Name}" WidthRequest="500"/>
```

```
</StackLayout>
```

```
<StackLayout Orientation="Horizontal">
```

```
<Label Text="Calories"/>
```

הזנת פרטי המאכל (קלוריות)

```
<Entry Text="{Binding Calories}" WidthRequest="500"/>
```

```
</StackLayout>
```

כפתור ליצירת המאכל

```
<Button Text="Add" Command="{Binding OnAddFoodClickedCommand}"
```

```
HorizontalOptions="Center"/>
```

```
</StackLayout>
```

שינויים עתידיים אפשריים

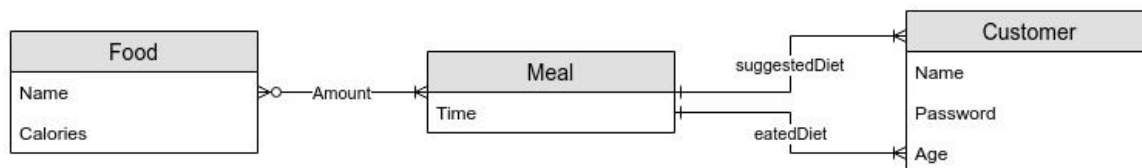
ישנם מספר שינויים הכרחיים ומספר שינויים אפשריים תחילה, כרגע למערכת אין צד שרת, זאת עקב קוצר זמן ובחירה להתמקד במתן ערך, זוהי בחירה המוגדרת כ- Bad Practice מבחינת אבטחת התוכנה, ובעצם מאפשרת (ואף מתבססת על) קשר ישיר בין צד הלקוח לבסיס הנתונים. בנוסף, יש צורך להוסיף פרמטרים רבים נוספים לאובייקט המאכל, אשר בלעדיהם המערכת אינה שימושית לתזונאים, כגון חלבון, פחמימה, שוחמן, ערכי ויטמינים ועוד. שינויים כאלה הם קלים עקב תכנון נכון של מערכת בצורה מונחת עצמים. שינוי ה Model של מאכל לא אמור לפגוע בתפקוד התקין של המערכת, ויצריך שינויים קלים בלבד בשכבות התצוגה. בנוסף, הוספת הפרמטרים יכולה לקרות רק בחלק מהדפים, והערכים הנוספים לא יפגעו בתפקודה התקין של המערכת, כך שאפשר להטמיע את הוספת ה- feature שלב אחרי שלב ואין צורך לעשות את כל השינויים במכה אחת. שינוי הכרחי נוסף מבחנת אבטחת תוכנה הוא לשמור את הסיסמאות באמצעות פונקציית Hash, שמירת סיסמאות כטקסט פשוט היא בחירה גרועה, אך שוב, מאילוצי זמן בחרתי קודם לאפשר מוצר שעובד.

שינויים אפשריים נוספים במערכת:

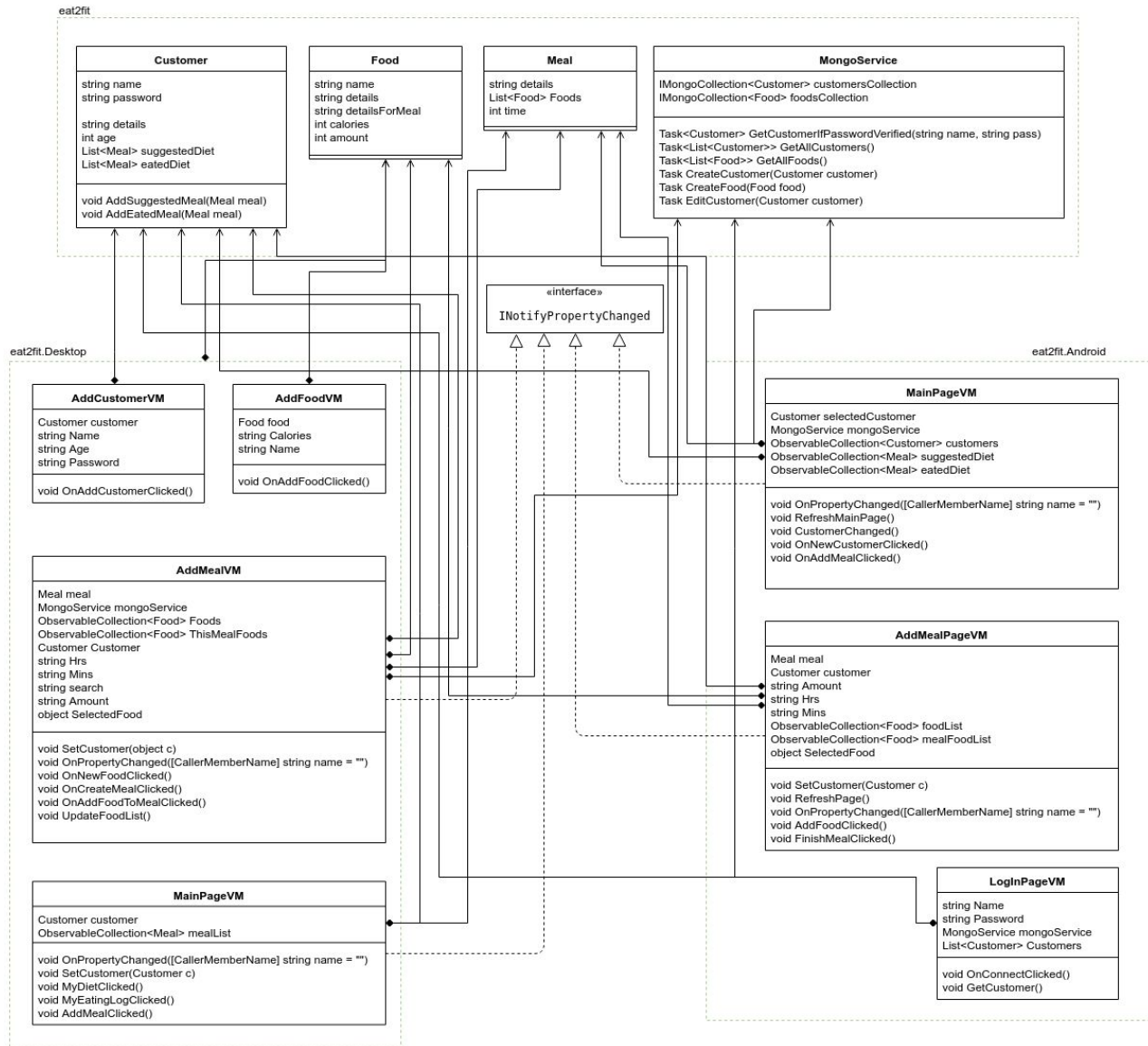
- יכולת העברת הודעות בין תזונאי ללקוחותיו
- יכולת שמירת מסמכים נוספים ב"תיק" וירטואלי לכל לקוח (תוצאות בדיקות דם ובדיקות אחרות, המלצות רופא מטפל וכד')
- אפשרות לניהול המפגשים עם הלקוח, ואפשרות לשמור סיכומי מפגשים כחלק מפרטי הלקוח

דיאגרמת ישויות קשרים

דיאגרמת ישויות הקשרים של המערכת פשוטה למדי, כפי שמתואר בשכבת הנתונים.



דיאגרמת מחלקות



הוראות הפעלה ממשק תזונאי

זהו המסך הראשי עבור התזונאי לפני בחירת לקוח.

The screenshot shows a desktop application window titled "eat2fitDesktop.UWP". The interface is designed for a nutritionist to manage customer intake and diet suggestions. It features a "Change Customer:" label followed by a text input field and a dropdown arrow. To the right of the input field is a "New Customer" button. Below this, there is a "Selected Customer:" label. The main area is divided into two columns: "Actual Customer Intake" on the left and "Suggested Diet For Customer" on the right. At the bottom right, there is an "Add Meal" button.

חלון הוספת לקוח חדש

בלחיצה על "New Customer" מהמסך הראשי ניתן להוסיף לקוח



The screenshot shows a Windows-style application window titled "eat2fitDesktop.UWP". Inside the window, there is a form for adding a new customer. The form consists of three text input fields, each preceded by a label: "Name:", "Password:", and "Age:". Below the "Age:" field, there is a small, light gray button labeled "Add". The window has standard Windows controls (back arrow, maximize, close) in the top right corner.

לחיצה על "Add" תוסיף את הלקוח לבסיס הנתונים.

לחיצה על חץ חזרה מאפשרת חזרה לאחור ללא הוספת לקוח חדש.

לאחר הוספת לקוח חדש ניתן לבחור בו ולטפל בו

The screenshot shows a web application window titled "eat2fitDesktop.UWP". At the top, there is a "Change Customer:" label followed by a dropdown menu containing "Moran Carmy" and a "New Customer" button. Below this, the "Selected Customer:" label is followed by the text "Moran Carmy, Age: 27". The main area of the application is divided into two columns: "Actual Customer Intake" on the left and "Suggested Diet For Customer" on the right. At the bottom right, there is a button labeled "Add Meal".

בחלון זה ניתן לראות את פרטי הלקוח הנבחר, וכן שתי רשימות.
רשימה של תפריט מוצע, ורשימה של צריכת המזון בפועל כפי שהוזנה על ידי הלקוח.
ניתן להוסיף ארוחה ללקוח בלחיצה על כפתור "Add Meal"

חלון הוספת ארוחה

eat2fitDesktop.UWP

Time: :

Foods:

Search

Rice
Calories: 120
Cottage Cheese
Calories: 100
Olive Oil
Calories: 800

In This Meal:

Amount

->

New Food Create Meal

יש להזין את שעת הארוחה, ולבחור לכל מאכל את הכמות אותה יש לצרוך בארוחה זו. בחלון זה ניתן לבחור מתוך מאכלים קיימים או להוסיף מאכל. בעת בחירת מאכל יש לציין את הכמות. כמו כן ניתן להיעזר בחיפוש למציאת מאכל במידה. לאחר בחירת מאכל והזנת כמות ניתן ללחוץ על חץ ההוספה. לאחר הוספת מאכלים והזנת כמויות ניתן ללחוץ על "Create Meal" ולהוסיף את הארוחה לתפריט הלקוח.

← eat2fitDesktop.UWP

— □ ×

Time: :

Foods:

In This Meal:

Search

Rice
Calories: 120

Cottage Cheese
Calories: 100

Olive Oil
Calories: 800

Amount

->

rice

Calories: 120 Amount: 100

New Food

Create Meal

לאחר יצירת הארוחה היא תופיע בעמוד הלקוח

eat2fitDesktop.UWP

Change Customer:

Moran Carmy

▼

New Customer

Selected Customer:

Moran Carmy, Age: 27

Actual Costumer Intake

Suggested Diet For Customer

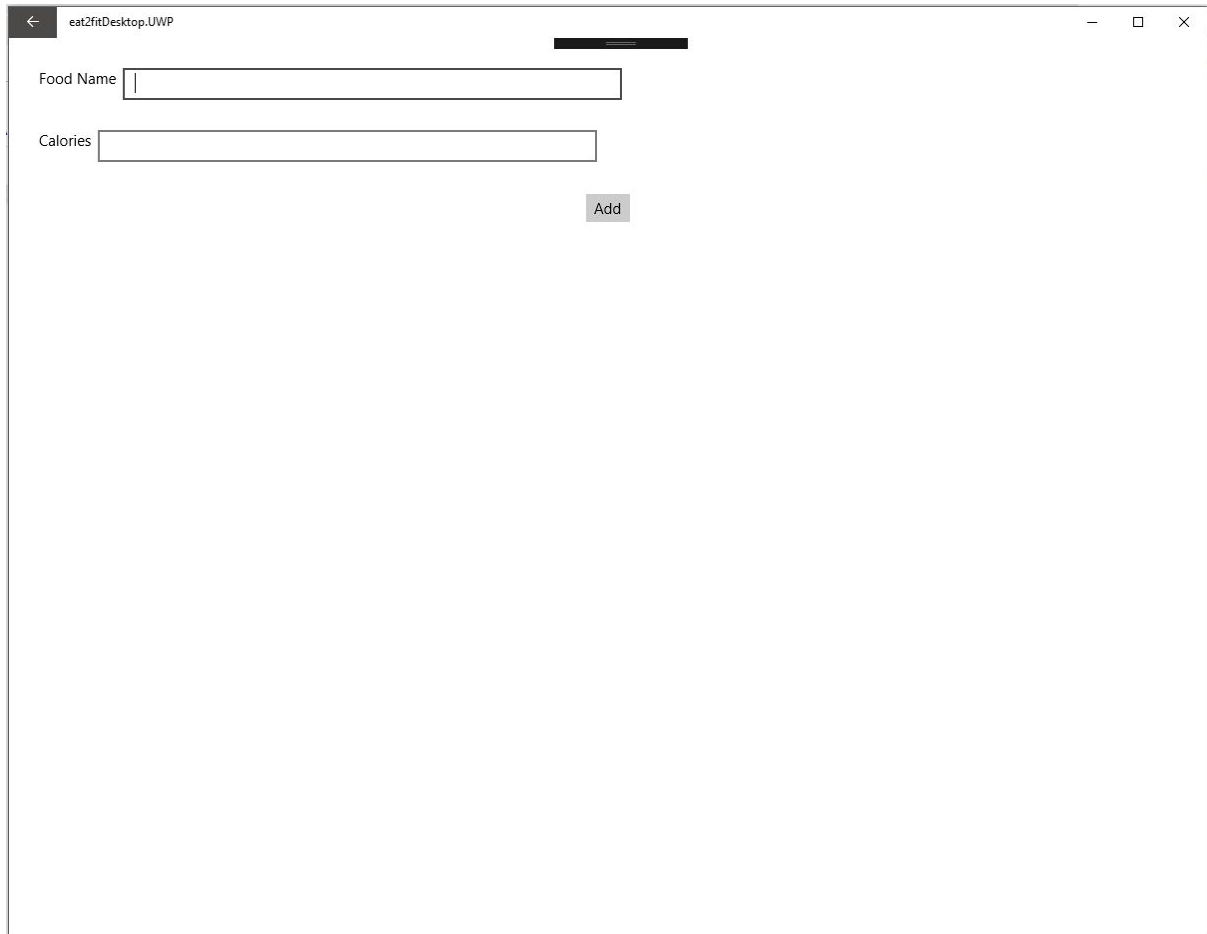
Meal Time: 10:10

Meal Calories: 120

Add Meal

חלון הוספת מאכל

לחלון זה ניתן להגיע על ידי לחיצה על "New Food" בחלון הוספת ארוחה.
לכל מאכל יש להזין את שמו ואת כמות הקלוריות בו.



The screenshot shows a Windows-style application window titled "eat2fitDesktop.UWP". It contains a form with two input fields: "Food Name" and "Calories". Below the "Calories" field is an "Add" button. The window has a standard Windows title bar with a back arrow, the application name, and minimize, maximize, and close buttons.

לאחר לחיצה על "Add" המאכל יתווסף לרשימת המאכלים.

השוואה בין התפריט לבין הצריכה בפועל

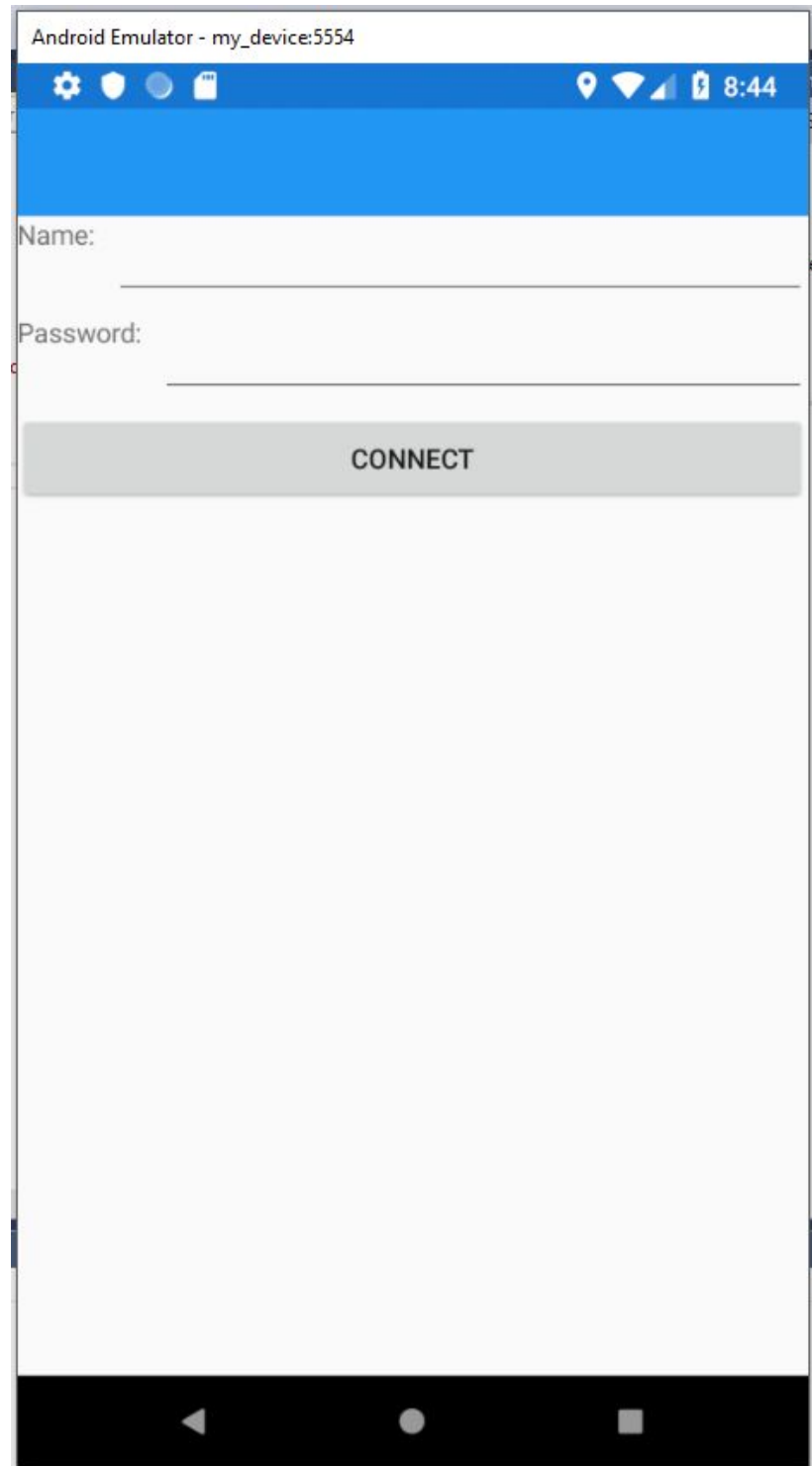
כאמור, במסך הראשי ניתן לראות את התפריט המוצע למול הצריכה בפועל. מסך זה צולם לאחר הוספת הארוחה מממשק הלקוח כפי שנראה בהוראות ההפעלה בעמודים הבאים.

The screenshot displays the 'eat2fitDesktop.UWP' application window. At the top, there's a 'Change Customer:' label followed by a dropdown menu showing 'Moran Carmy' and a 'New Customer' button. Below this, 'Selected Customer:' is followed by the text 'Moran Carmy, Age: 27'. The main area is divided into two columns: 'Actual Costumer Intake' and 'Suggested Diet For Customer'. The 'Actual' column shows 'Meal Time: 10:15' and 'Meal Calories: 116'. The 'Suggested' column shows 'Meal Time: 10:10' and 'Meal Calories: 120'. At the bottom right, there is an 'Add Meal' button.

Actual Costumer Intake	Suggested Diet For Customer
Meal Time: 10:15	Meal Time: 10:10
Meal Calories: 116	Meal Calories: 120

הוראות הפעלה ממשק לקוח

מסך התחברות



במסך זה יש להכניס שם משתמש וסיסמא כפי שסופקו על ידי התזונאי המטפל.

מסך ראשי

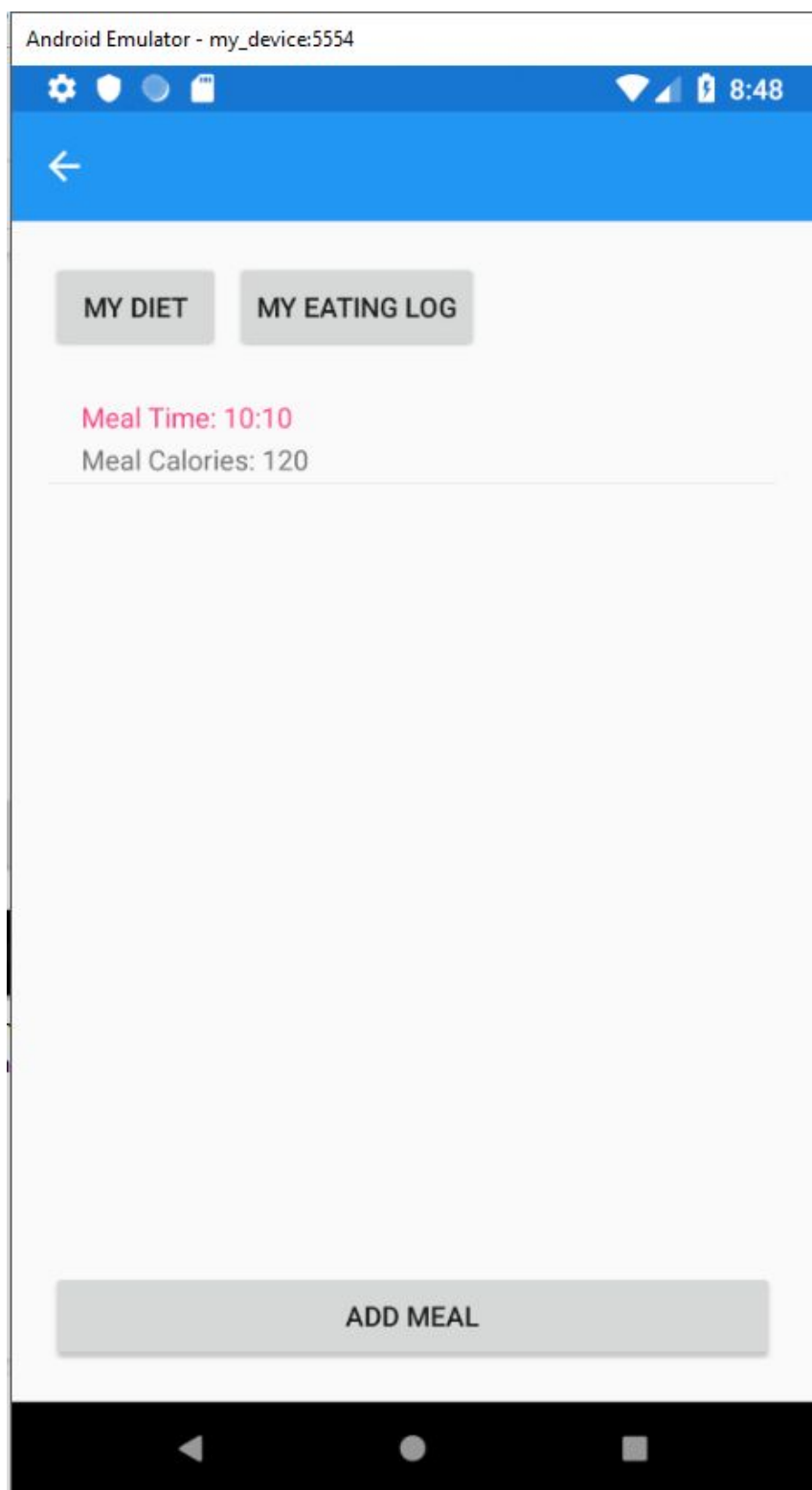


במסך זה ניתן להציג את התפריט שלי כפי שהוזן על ידי התזונאי ואת יומן האכילה שלי כפי שהזנתי.

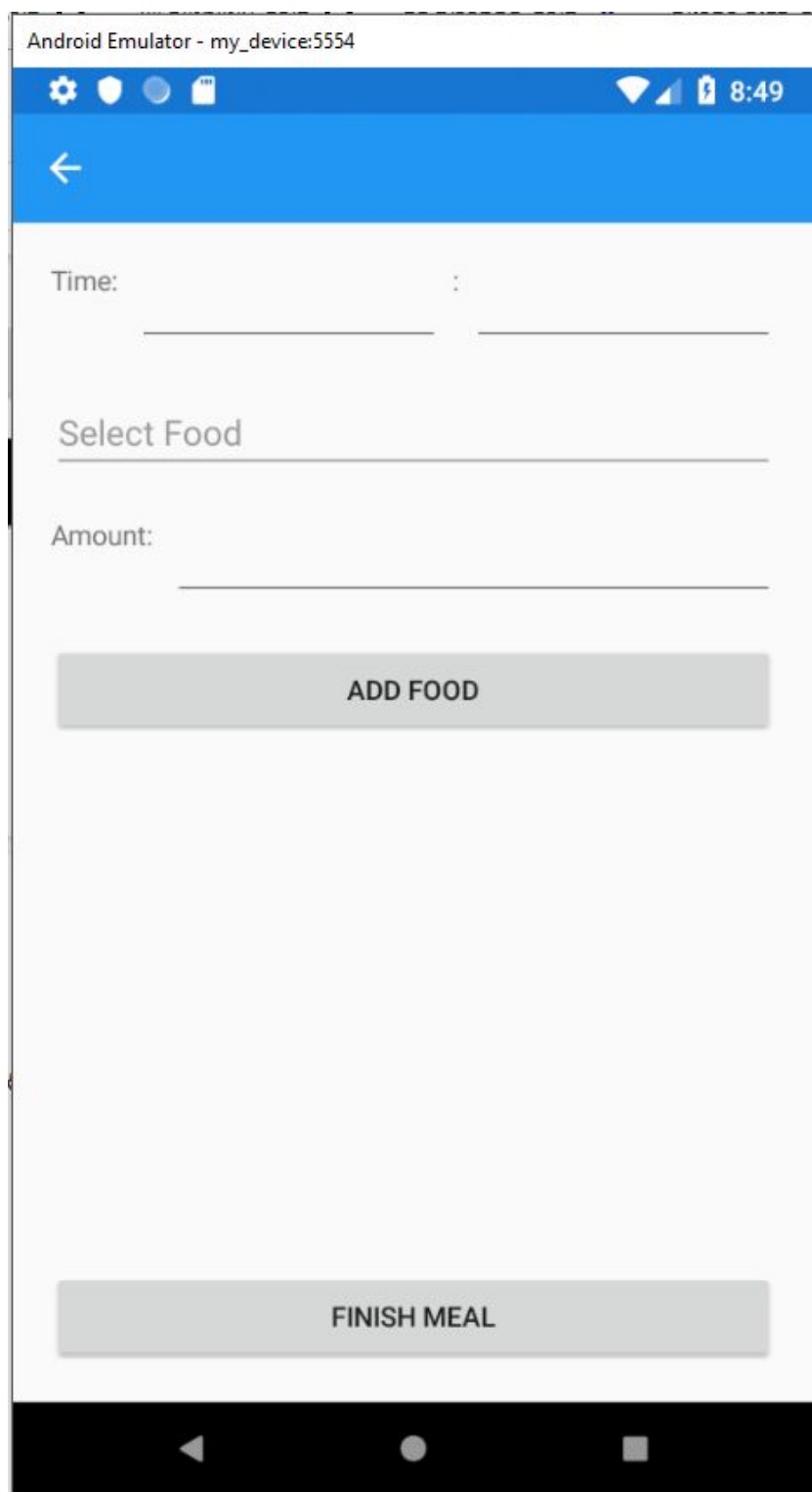
כמו כן ניתן לעבור לחלון הוספת ארוחה ולתעד ארוחה שנאכלה בפועל.

התפריט המוצע

בלחיצה על "MY DIET" ניתן לראות את התפריט כפי שהוזן בתוכנת המחשב על ידי התזנואי המטפל



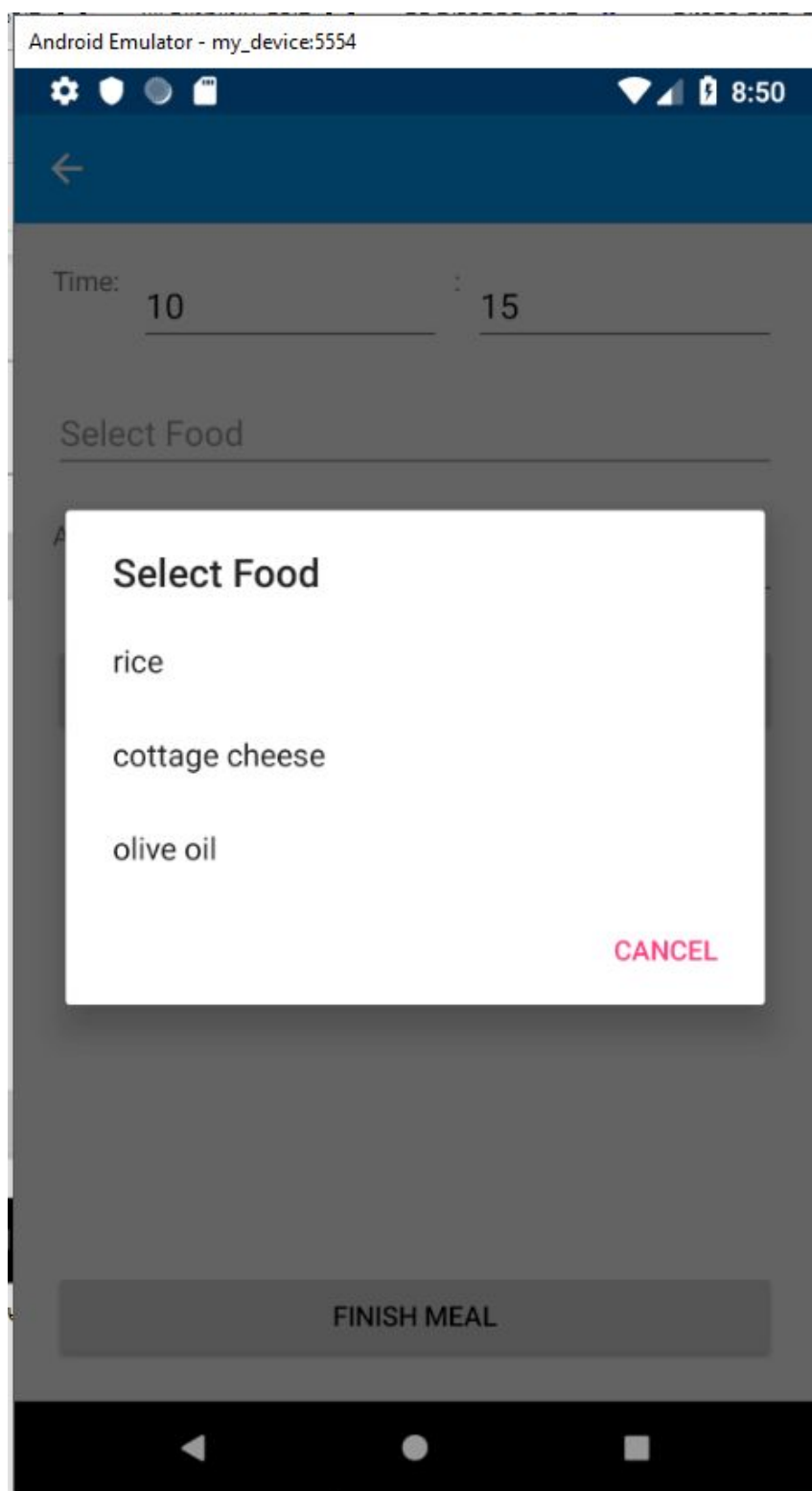
חלון הוספת ארוחה



The screenshot shows an Android emulator window titled "Android Emulator - my_device:5554". The status bar at the top displays icons for settings, a shield, a blue circle, and a battery icon, along with the time 8:49. The app interface has a blue header bar with a white back arrow. Below the header, the form contains the following elements: a "Time:" label followed by two input fields separated by a colon; a "Select Food" label followed by a text input field; an "Amount:" label followed by a text input field; a grey button labeled "ADD FOOD"; and a grey button labeled "FINISH MEAL" at the bottom. The bottom of the emulator shows the standard Android navigation bar with back, home, and recent apps buttons.

יש להוסיף את שעת האכילה
לבחור במאכלים שנאכלו ולהוסיף את הכמויות

בחירת מאכל



לאחר הוספת המאכלים ניתן ללחוץ "FINISH MEAL" והארוחה תעבור ליומן האכילה

Android Emulator - my_device:5554

8:51

←

Time: 10 : 15

cottage cheese

Amount: 20

ADD FOOD

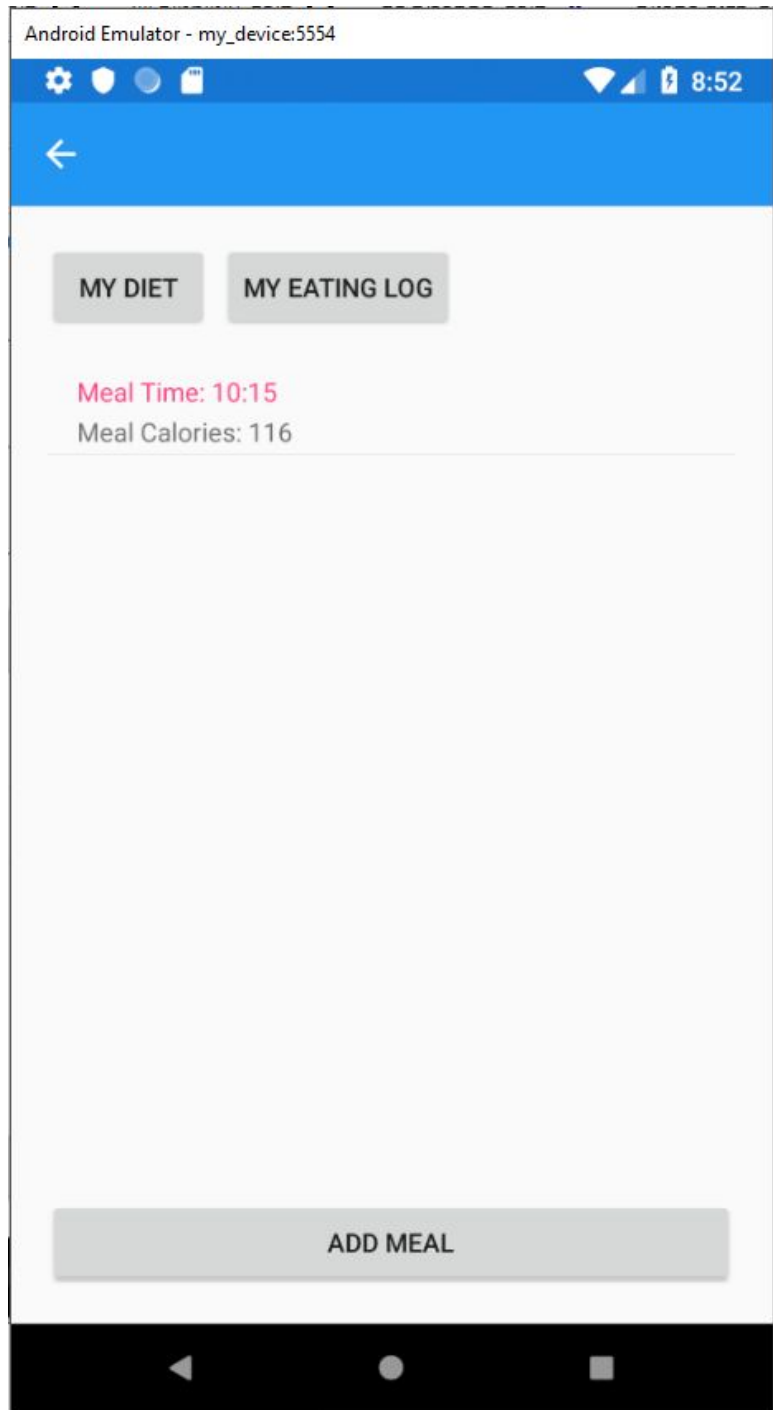
rice
Calories: 120 Amount: 80

cottage cheese
Calories: 100 Amount: 20

FINISH MEAL

יומן האכילה

בלחיצה על "MY EATING LOG" ניתן להציג את יומן האכילה שלי כפי שנאכל בפועל



יומן האכילה מוצג באופן דומה לתפריט המוצע, כך שניתן להשוות ביניהם בקלות