



מוגש לבית הספר למדעי המחשב

המסלול האקדמי, המכללה למנהל

כחלק מהדרישות האקדמיות

לתואר .B.Sc במדעי המחשב

מיכאל רייזלמן 308273465

דודו למברברג 204028195

מנחה : ד"ר שי הורוביץ

2015-2016

תוכן עניינים

[1 תקציר 6](#_Toc457509185)

[2 הקדמה – עולם הבעיה 7](#_Toc457509186)

[2.1 מטרת הפרויקט 7](#_Toc457509187)

[3 מה נעשה בתחום 8](#_Toc457509188)

[3.1 מי המתחרים שלנו 8](#_Toc457509189)

[3.2 מדוע הפתרון שלנו יותר טוב? 8](#_Toc457509190)

[4 הצדקה אקדמאית 10](#_Toc457509191)

[4.1 השימוש באלגוריתם 10](#_Toc457509192)

[4.2 האלגוריתם 10](#_Toc457509193)

[4.2.1 הסבר כללי 10](#_Toc457509194)

[4.2.2 איך עובד האלגוריתם 11](#_Toc457509195)

[4.3 האלגוריתם בקוד 11](#_Toc457509196)

[4.3.1 שליפת נתונים אודות העיר 11](#_Toc457509197)

[4.3.2 ביצוע החיתוך עם K-means 11](#_Toc457509198)

[4.3.3 חלוקה לשטחים עם voronoi 12](#_Toc457509199)

[4.3.3 עדכון השטחים המשויכים לעיר ב-DB 13](#_Toc457509200)

[4.4 האלגוריתם במסכים 14](#_Toc457509201)

[5 גוף הפרויקט (DR) 14](#_Toc457509202)

[5.1 ארכיטקטורה 14](#_Toc457509203)

[5.1.1 שכבת הנתונים DataBase 14](#_Toc457509204)

[5.1.2 שכבת ה- server 14](#_Toc457509205)

[5.1.3 שכבת הclient 15](#_Toc457509206)

[תרשים מספר 2 – תרשים ארכיטקטורה 16](#_Toc457509207)

[5.2 מבנה נתונים 16](#_Toc457509208)

[תרשים מספר 3 - EER Diagram 18](#_Toc457509209)

[תרשים מספר 4 – Class Diagram 19](#_Toc457509210)

[6 אפיון המסכים 20](#_Toc457509211)

[6.1 מסכי המערכת 20](#_Toc457509212)

[7 קשיים במהלך הפרויקט 26](#_Toc457509233)

[8 נספח – SRS Document 26](#_Toc457509234)

[9 ביבליוגרפיה 27](#_Toc457509235)

# תקציר

Homade היא פלטפורמה הנותנת אפשרות לבשלנים לפרסם אוכל ביתי אותו הם מכינים למכירה. אנשים מהאזור יכולים לצפות במנות שבאזור, לקנות את המנות אשר מוצעות באותו רגע למכירה, ולאחר מכן לדרג אותן.

בנוסף המערכת מאפשרת שיחה אקטיבית בין הבשלן ללקוח, מקדמת בשלנים מוצלחים, ומציעה לבשלנים מתחילים מנות להכנה אשר צפוי שיצליחו באזור בו הם מתכננים למכור.

המערכת משתמשת באלגוריתמי data mining לצורך התאמת מנות מומלצות לפי אזורים שונים, וממליצה עליהם לבשלנים המתחילים באזור.

האתגרים שהמערכת מתמודדת איתם:

* פיתוח המסכים בצורה נוחה למשתמש.
* ביצוע יעיל של אלגוריתמי ההמלצות.
* הצגת המנות הרלוונטיות למשתמשים.
* חיפוש יעיל ומהיר בין המנות האפשריות.

הממשק הוויזואלי של המערכת נבנה ב AngularJs, bootstrap תוך כדי שימוש במספר ספריות המקלות על הפיתוח (chartjs, google maps & more).

המערכת ממומשת בטכנולוגיות :

צד שרת – ממשק Restful הכתוב ב-NodeJS.

Database – בסיס נתונים – mongo.

# 2 הקדמה – עולם הבעיה

כיום יש הרבה מאוד בשלנים חובבים שמחפשים דרך להרוויח כסף מהאוכל שהם מכינים. לצערם אין להם פלטפורמה שבה הם יכולים לשווק את המנות שלהם ולכן קשה להם נורא למצוא לקוחות.

מצד שני, קיימים הרבה אנשים שנהנים לאכול אוכל ביתי וכתוצאה מנסיבות מסוימות (חוסר בזמן, חוסר בידע) הם אינם יכולים להכין לעצמם כזה.

בפלטפורמה ישנה יכולת להתריע בפני הבשלנים איזה מנות הן הכי פופולריות באזור בו הם מבשלים וכן, להציג מנות פופולריות באזור עבור המשתמשים.

מטרת העל של הפלטפורמה היא לחבר בין בשלנים לבין לקוחות פוטנציאליים שלהם שיהיו מעוניינים באוכל אותו הם מבשלים.

2.1 מטרת הפרויקט

הפלטפורמה תספק פתרון הן לבשלנים שמתקשים למצוא לקוחות והן ללקוחות שמעוניינים לקנות אוכל ביתי. היא תציג את המנות בצורה נוחה למשתמשים ותציג את הדירוגים שלהם ואת הדירוגים של כל הזמנה שבוצעה ע"י משתמש כלשהו.

הפרויקט יאפשר להפוך את תחביב הבישול למקצוע של ממש עבור כל המעוניין בכך, ואולי גם להיות אחד הבשלנים המוצלחים באזור.

# 3 מה נעשה בתחום

כיום יש מספר דרכים בהם בשלנים יכולים למכור את האוכל שלהם. הבשלנים יכולים לפרסם מודעות ברחוב המפרסמות אותם. יכולים לפרסם מפה לאוזן, והמקצועיים יותר יכולים אף לפתוח מסעדת אוכל ביתי.

מצד שני, אנשים המתעניינים בצריכה של אוכל ביתי, יכולים לשמוע מאחרים על אוכל ביתי באזור שלהם, יכולים לחפש מסעדות המספקות את הדברים באזור שלהם וזהו פחות או יותר.

בנוסף רוב האוכל הביתי שנמכר היום, נמכר ע"י אנשים שעושים את זה במסגרת של מסעדות אוכל ביתי, לדעתנו הרעיון שלנו יעודד אנשים פרטיים להכין גם הם אוכל ביתי, בזמנם הפנוי (ולא במסגרת העבודה) וכך להשלים הכנסה.

## מי המתחרים שלנו

בנוסף יש מספר פתרונות טכנולוגיים הקיימים לבעיה:

1. שם: יאמי - Yummi

כתובת: <https://www.yummi.co.il/>

אתר להזמנת אוכל ביתי משפים שונים ברחבי הארץ.

חסרונות: אתר אזורי (פעיל רק בישראל), האתר כופה הזמנה מראש – בעוד אצלנו ההזמנה היא מיידית ומתקבלת אצל השף באותו הרגע.

1. שם: foodieshares

כתובת: <http://foodieshares.com/>

אפליקציה להזמנת אוכל משפים מקומיים בארה"ב.

חסרונות: פעילה במספר מצומצם של מקומות, הגבלה בתנאי הקבלה לשפים – לא כל אחד יכול להיות שף.

## מדוע הפתרון שלנו יותר טוב?

1. אנו נותנים פתרון כלל אזורי בניגוד לשאר האפליקציות הזמינות במספר אזורים מצומצם.
2. קיים אצלנו אלגוריתם המלצות הממליץ לשפים מה להכין באזורים מסוימים על מנת למקסם את הרווחים שלהם, ולהתאים אותם לקהל המקומי.
3. אין אצלנו הגבלה לשפים – כל אחד יכול להיות שף באמצעות ביצוע מספר פעולות פשוטות.
4. הפלטפורמה שלנו מציעה דרכים נוחות לשיחה בין הבשלנים לבין הלקוחות שלהם. היא מאפשרת להם לבקש בקשות, לשוחח עם הבשלן על כל דבר שהם חפצים בנוגע למנה שלהם ו/או על האיסוף.
5. קיימת מערכת דירוג המאפשרת לקדם את השפים המוצלחים ולתת ביקורת לאחרים כדי שיוכלו להשתפר.
6. במערכת מוצגות תוצאות על גבי מפה כדי שאנשים יוכלו להתמצא בקלות לגבי היכן נמצא האוכל באזור.
7. במערכת ניתן לעקוב בנוחות אחרי סטטוס ההזמנה ולהתעדכן לגבי כל שינוי שכזה.

# 4 הצדקה אקדמאית

הפרויקט משתמש באלגוריתם למידה לצורך מתן המלצות עבור מנות מומלצות להכנה לשפים.

\*בהמשך נראה פירוט מעמיק

4.1 השימוש באלגוריתם

האלגוריתם מזהה בכל פעם שנעשית הזמנה במערכת. כאשר נזהה כי נעשה שינוי משמעותי במספר ההזמנות מאיזור מסוים נפעיל את האלגוריתם K-means שימצא לנו את מרכזי הפעילות בעיר בה נעשה השינוי ולאחר מכן לחלק את העיר לאזורים לפי המרכזים הללו בעזרת שיטת החילוק של voronei. בכל אזור כעת אנחנו מחשבים את הטאגים הפופולריים ביותר ומציגים אותם למשתמש.

4.2 האלגוריתם

## 4.2.1 הסבר כללי

**האלגוריתם K-means -**

האלגוריתם הינו אלגוריתם למידה שמסווג נקודות למספר מרכזים ידוע מראש כאשר כל נקודה מסווגת למרכז הקרוב ביותר אליה. האלגוריתם בוחר רנדומלית מספר מרכזים(ע"פ ה-k שנשלח כפרמטר), ואז הוא מסווג כל נקודה לפי המרכז הקרוב אליה ובעצם יוצר קבוצה של נקודות עבור כל מרכז. כעת הוא לוקח את כל הקבוצות ומחשבת את המרכזים של הקבוצות שנוצרו ע"פ הנקודות ששויכו לכל קבוצה. כעת הוא מפעיל שוב את החלוקה של הנקודות למרכזים החדשים וכך חוזרת חלילה עד אשר המרכזים נשארים קבועים לאחר שתי חלוקה מחודשת.

**שיטת החילוק של - voronoi**

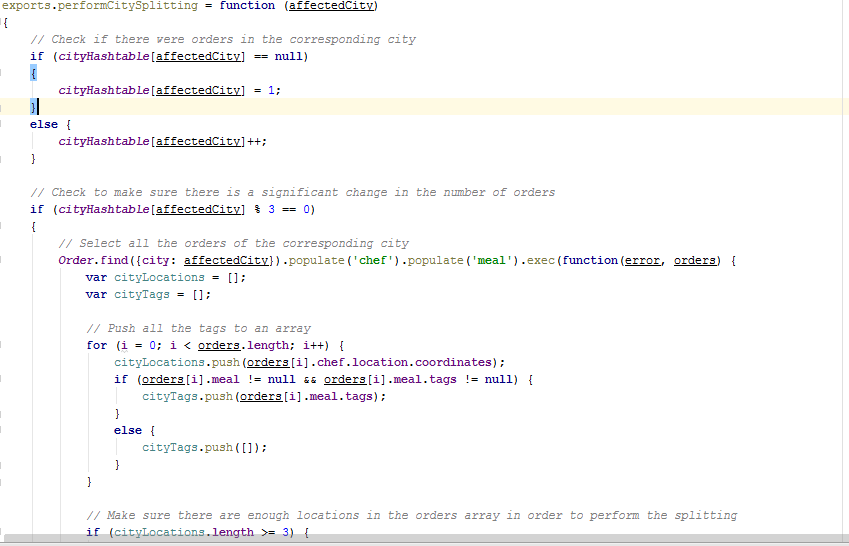
זוהי שיטת חילוק של שטחים שבה כל נקודה בתוך גוף תוחם מיוחסת לשטח מסוים ע"פ הקרבה שלה לאחת הנקודות (המרכזים מה-kmeans).

## 4.2.2 איך עובד האלגוריתם

האלגוריתם מקבל כקלט את הנקודות שבהם בוצעו ההזמנות בעיר מסוימת. האלגוריתם כעת מפעיל את K-means ומוצא את המרכזים העירוניים. כעת משתמש האלגוריתם ב-voronoi כאשר המרובע החוסם הוא העיר ואילו הנקודות המרכזיות הן המרכזים שנוצרו בתהליך הK-means. מה שכעת הוא עושה הוא לקחת כל הזמנה ולשלוף ממנה את הטאגים הרלוונטיים אליה, ולשייך את הטאגים האלה לאזור הרלוונטי על פי שייכות ההזמנה לאזור. כעת נעשית ספירה של מספר ההופעות לכל טאג, וסיווג הטאגים הפופולריים בכל אזור ושמירתם.

## 

4.3 האלגוריתם בקוד

4.3.1 שליפת נתונים אודות העיר

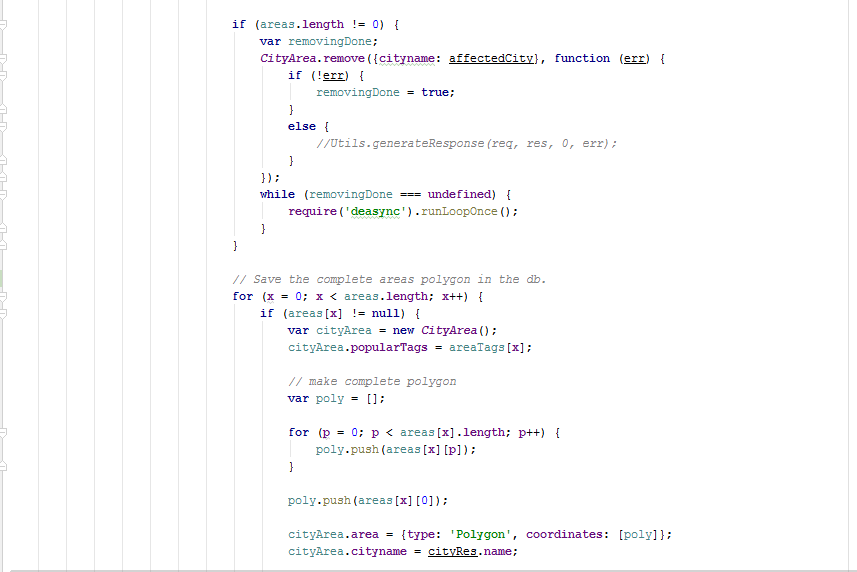
4.3.2 ביצוע החיתוך עם K-means



4.3.3 חלוקה לשטחים עם voronoi

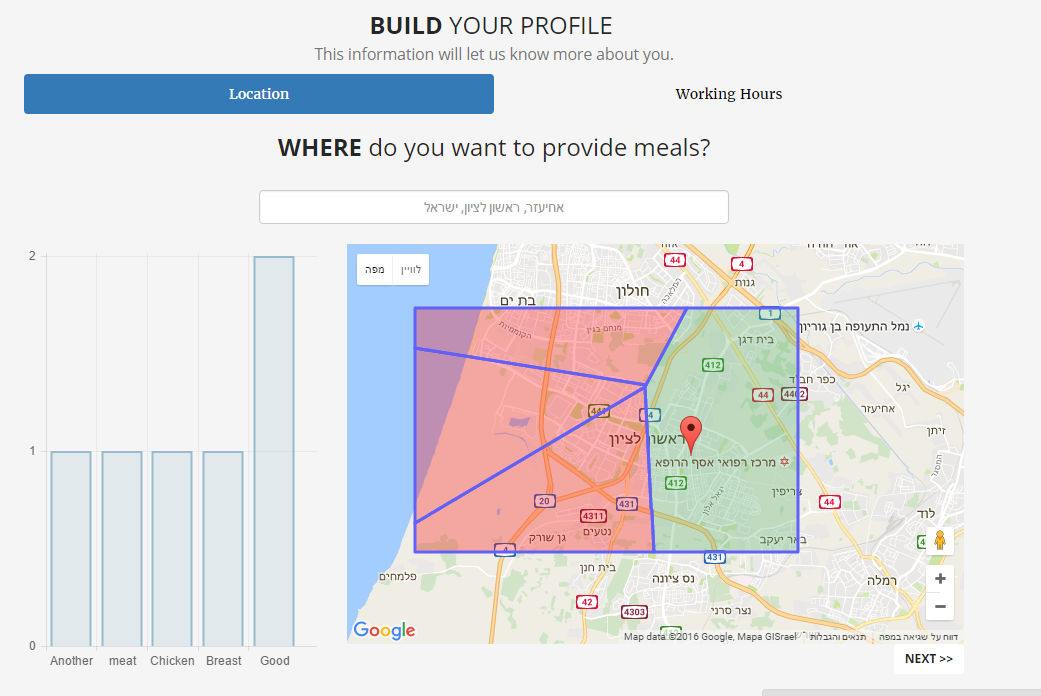


4.3.3 עדכון השטחים המשויכים לעיר ב-DB





* 1. האלגוריתם במסכים



# 5 גוף הפרויקט (DR)

5.1 ארכיטקטורה

האפליקציה, הינה אפליקציית web

ארכיטקטורת המערכת בנויה מכמה שכבות:

1. שכבת הנתונים – DB
2. שכבת ה- server
3. שכבת ה - client

## 5.1.1 שכבת הנתונים DataBase

שרת בסיס הנתונים mongoDB השומר את הנתונים כ-jsonים. האפליקציה שומרת את כל הנתונים אודות המשתמש, נתוני השפים המנות, ההתכתבויות וכו'.

## 5.1.2 שכבת ה- server

השכבה מקשרת בין שכבת הנתונים (DB) לשכבת הUI (Client) השכבה הזאת אחראית לשלוף את כל הנתונים מהDB ולהפעיל מניפולציות שונות על הנתונים על מנת שיהיו ניתנים לשימוש ע"י ה-client.

בנוסף השכבה אחראית על הוספה, עדכון והסרה של נתנים מהDB.

הפעולות התקבלו ב client והserver ידע לבצע את הפעולה הנדרשת מול הDB.

## 5.1.3 שכבת הclient

שכבת הclient אחראית על בדיקת הנתונים המוזמנים על ידי המשתמש (תאריכים תקינים, שמות תקינים וכו').

בנוסף על הclient להציג את הנתונים בצורה כמה שיותר ברורה למשתמש וקלה לתפעול, בנוסף. הקליינט יתקשר מול השרת כדי להכניס נתונים ולשמור שינויים שנעשו.

בגלל שהקליינט כתוב ב-angular חלק גדול מהלוגיקה מתבצעת בקליינט ולא בשרת בניגוד לאפליקציות web קלאסיות(PHP, ASP.Net וכו').

### תרשים מספר 2 – תרשים ארכיטקטורה

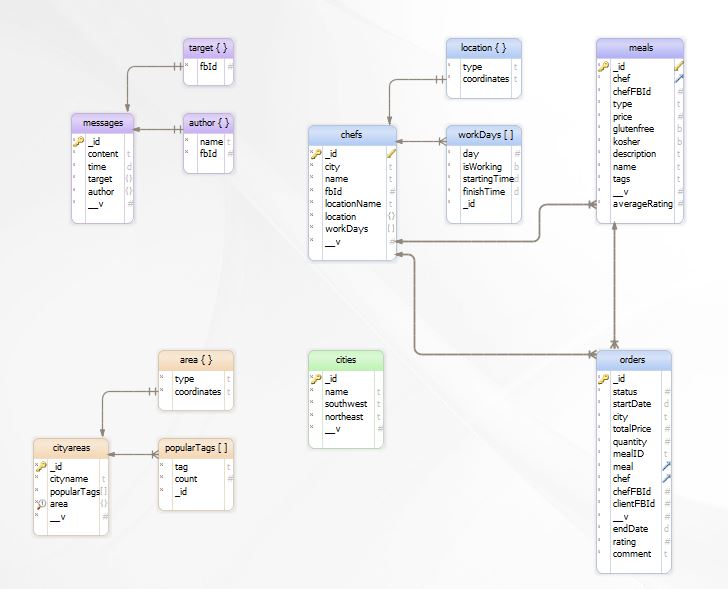


5.2 מבנה נתונים

בסיס הנתונים מכיל את הטבלאות הבאות:

* Chefs - מכיל פרטים אודות הבשלנים המערכת. בטבלה נשמרים נתונים אודות הכתובת והמיקום של השף (לצורך שימוש באלגוריתם אח"כ). בנוסף נשמרים נתונים אודות משתמש הפייסבוק של הבשלן ונתונים אודות הזמינות של הבשלן (באיזה ימים ושעות הוא עובד).
* Cities - נתונים אודות ערים שבהם הבשלנים שלנו זמינים. עבור כל בשלן שמתווסף אנו בודקים האם העיר שלו קיימת כבר במסד הנתונים ואם לא, אנחנו בודקים מול ה-api של גוגל את הריבוע שחוסם את העיר ושומרים אותו בטבלה.
* CityAreas - שומר את תוצאות החלוקה של האלגוריתם. בין היתר שומר נתונים אודות השטחים בכל עיר(השטח הגיאוגרפי), ואת הטאגים הפופולריים באזור וסטטיסטיקות עליהם.
* Meals - מכילה נתונים על המנות במערכת, על כל מנה נשמרים פרטים כלליים עליה, פרטים על השף שלה, טאגים שקושרו אליה ועוד.
* Messages - מכילה את כל ההודעות הנשלחות בין בשלנים ללקוחות, נשמרים פרטים אודות השולח, המקבל, התוכן ובזמן השליחה.
* Orders - מכילה פרטים אודות הזמנות. עבור כל הזמנה נשמרים פרטי המנה בהזמנה הכמות שהוזמנה, הסכום ששולם וסטטוס ההזמנה הנוכחי.

### תרשים מספר 3 - EER Diagram



### תרשים מספר 4 – Class Diagram

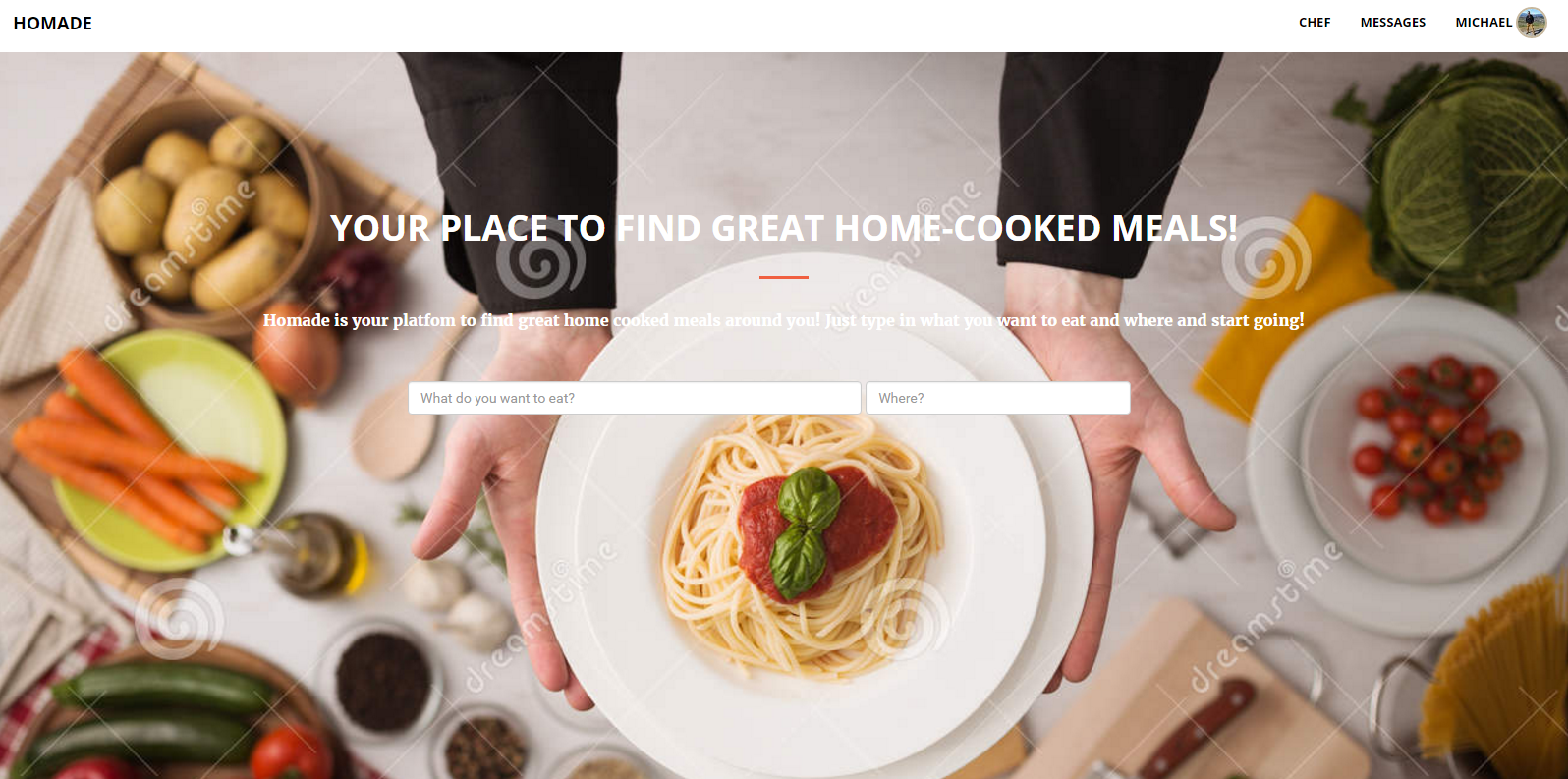


# 6 אפיון המסכים

6.1 מסכי המערכת

### תרשים מספר 5 - עץ המסכים

## 6.1.1 Home Page



## 

## 6.1.2 Login with facebook

הכנסת שם משתמש וסיסמא

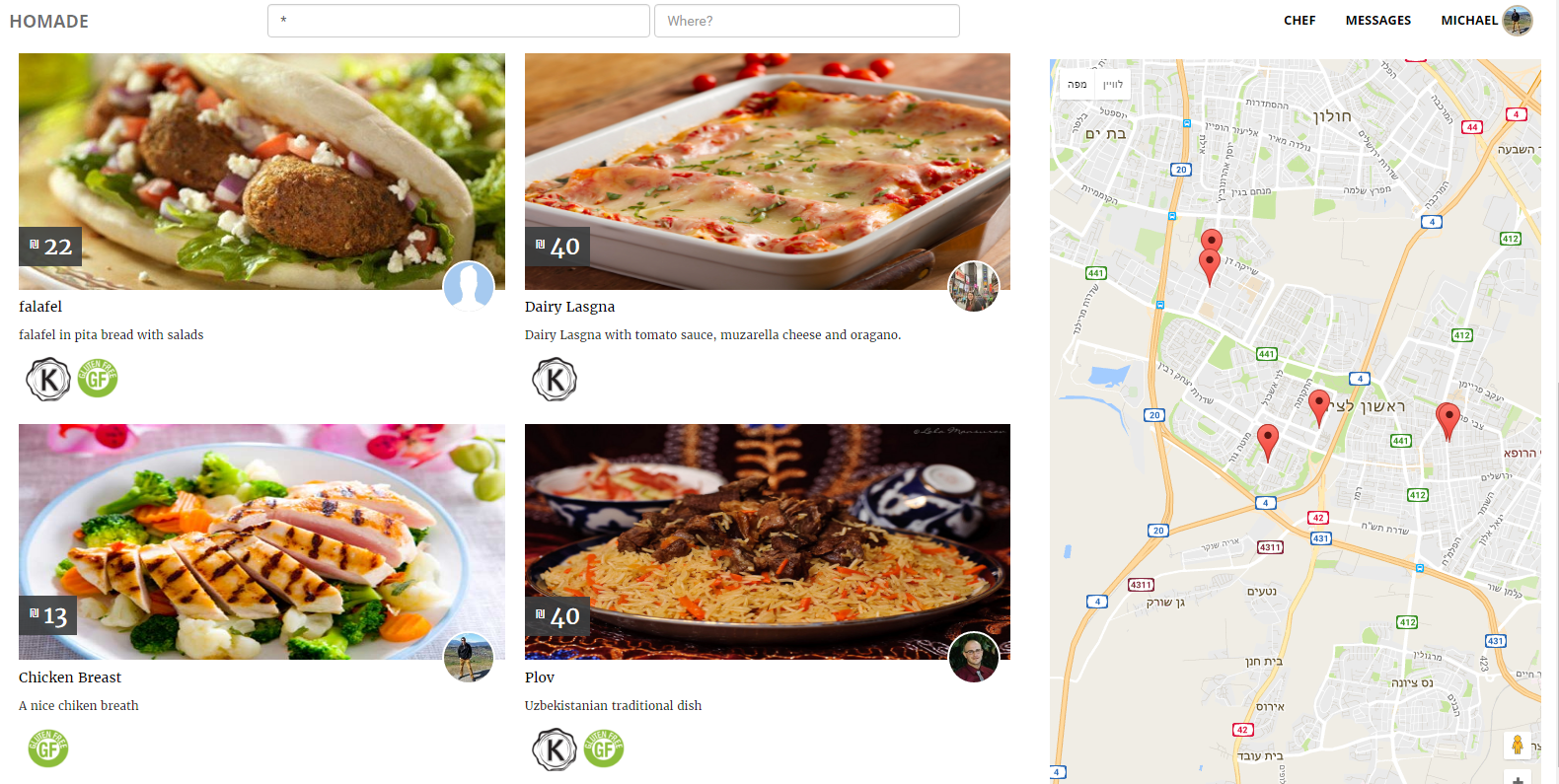
## 

הכנסת שם משתמש וסיסמא

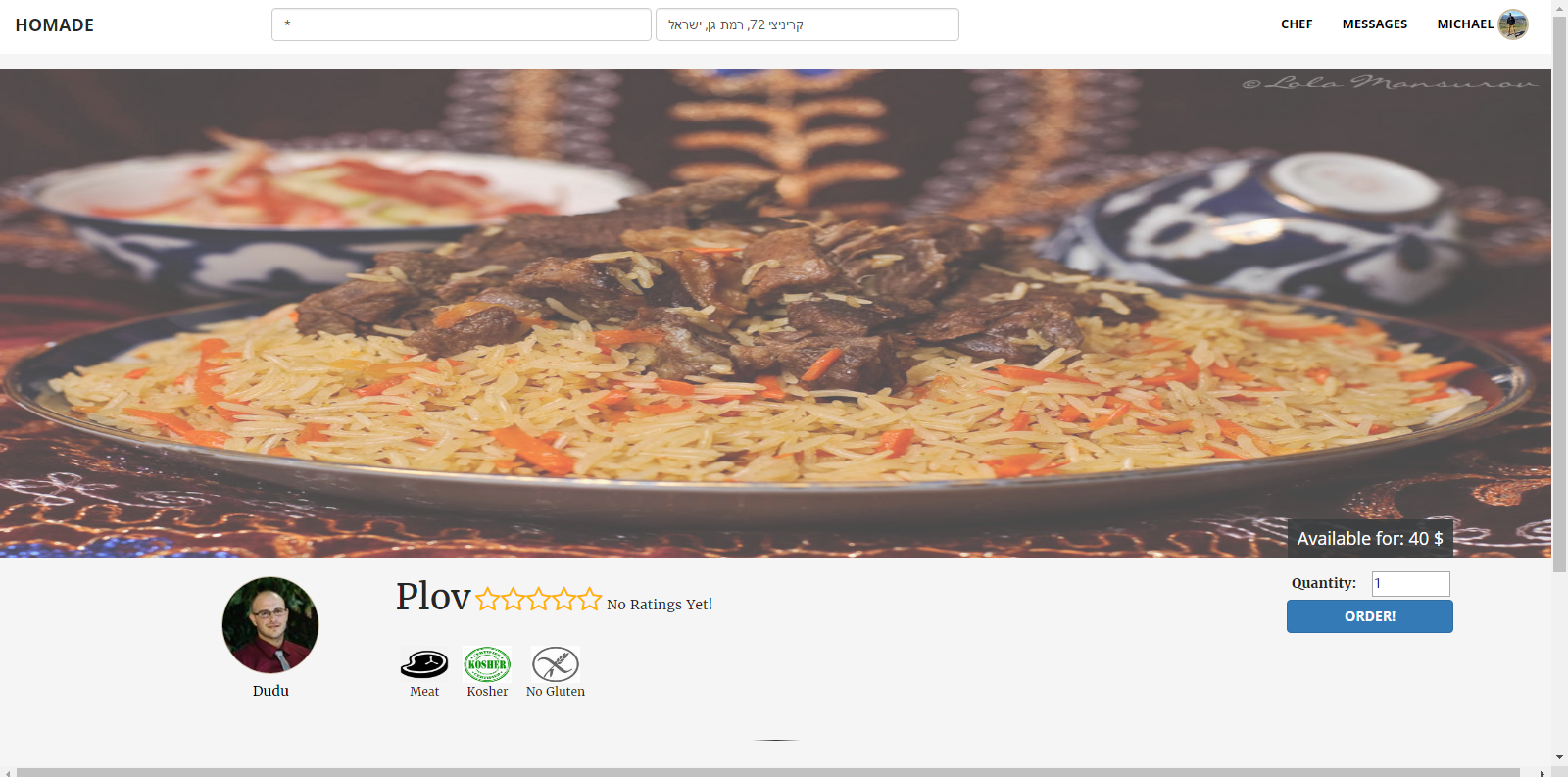
## 6.1.3 Messages

## 

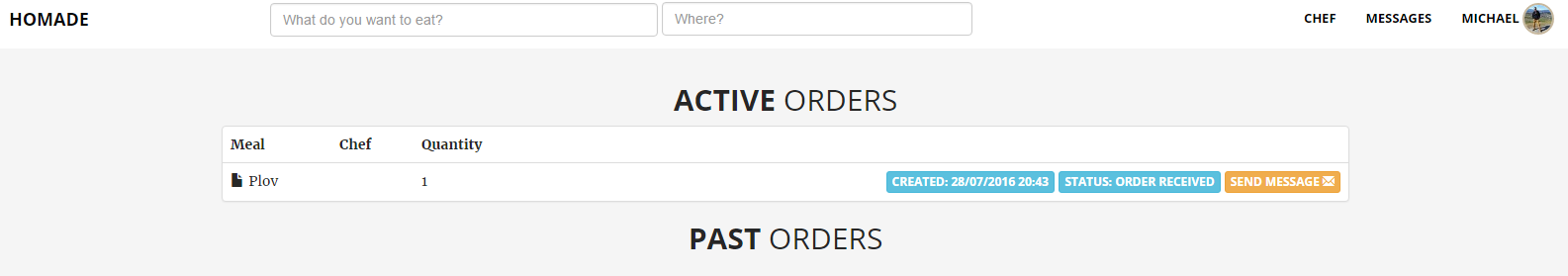
## 6.1.4 Search



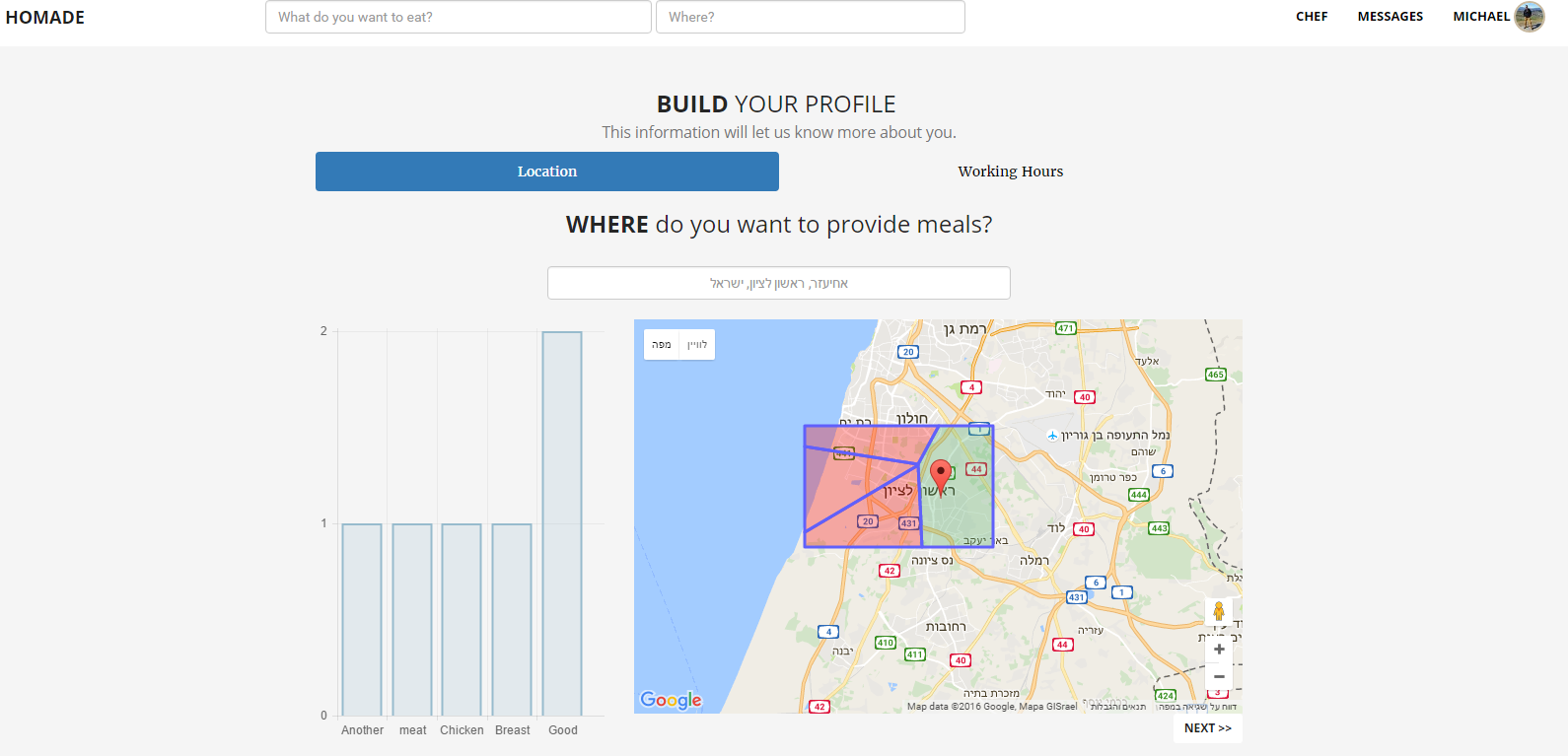
## 6.1.4 Order meal



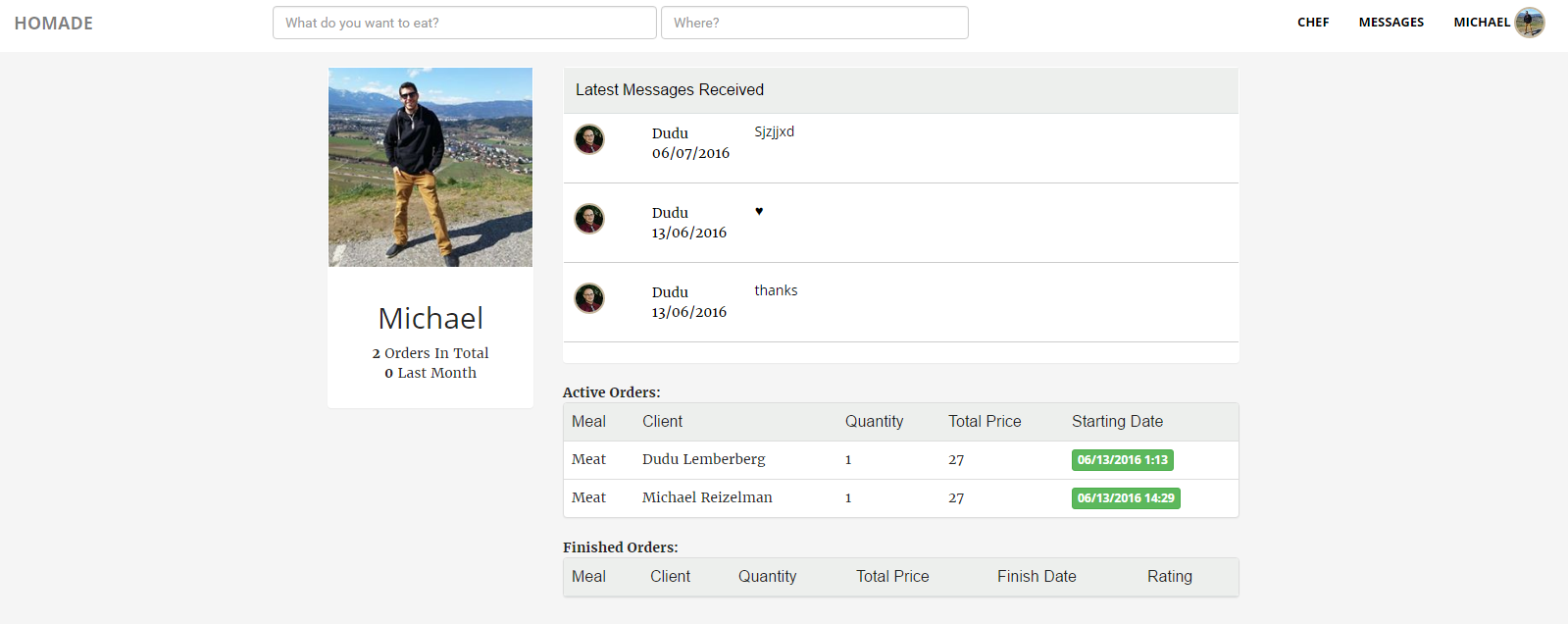
## 6.1.5 Client Orders



## 6.1.6 Become chef / Edit details



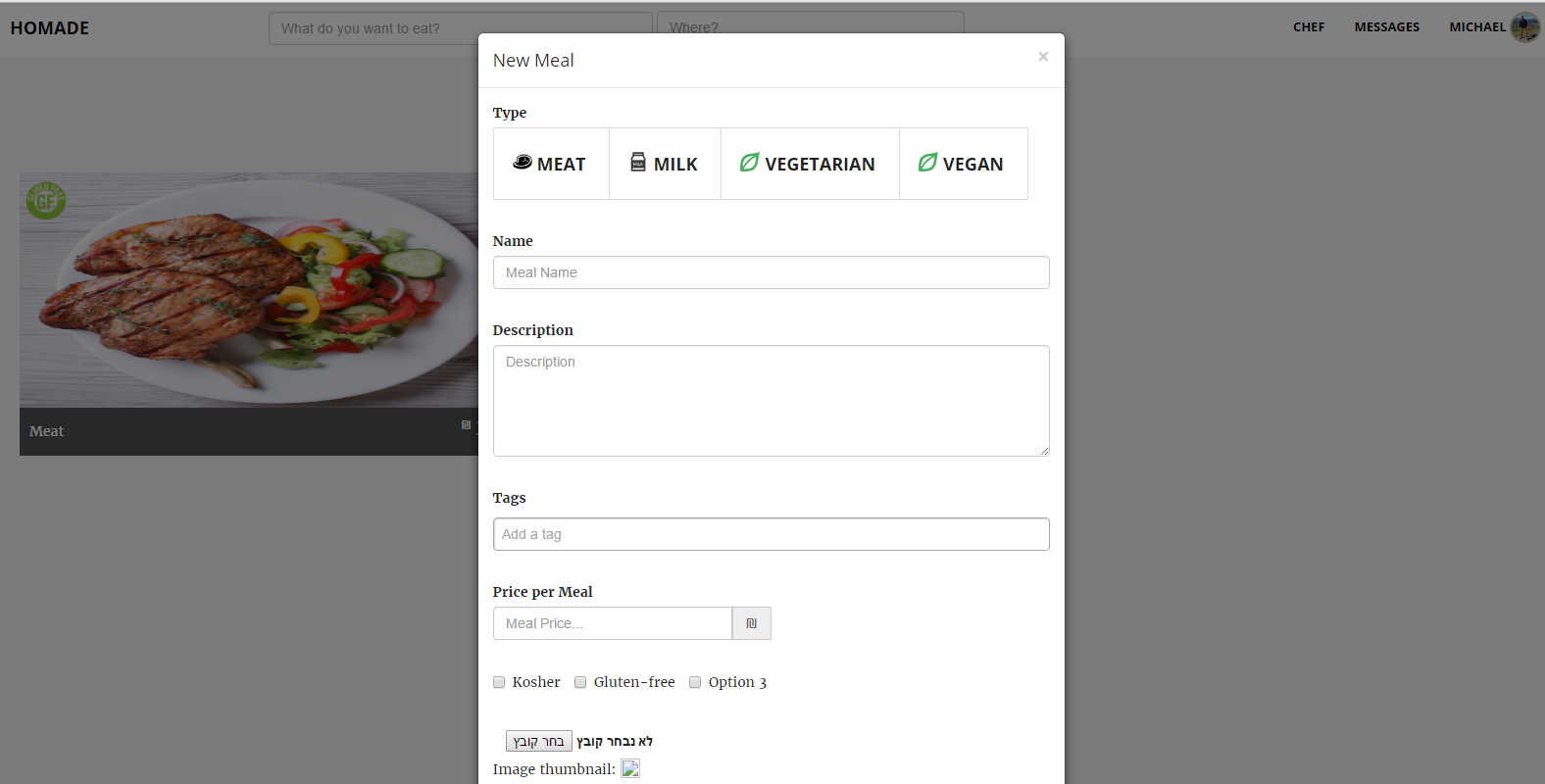
## 6.1.7 Chef Dashboard



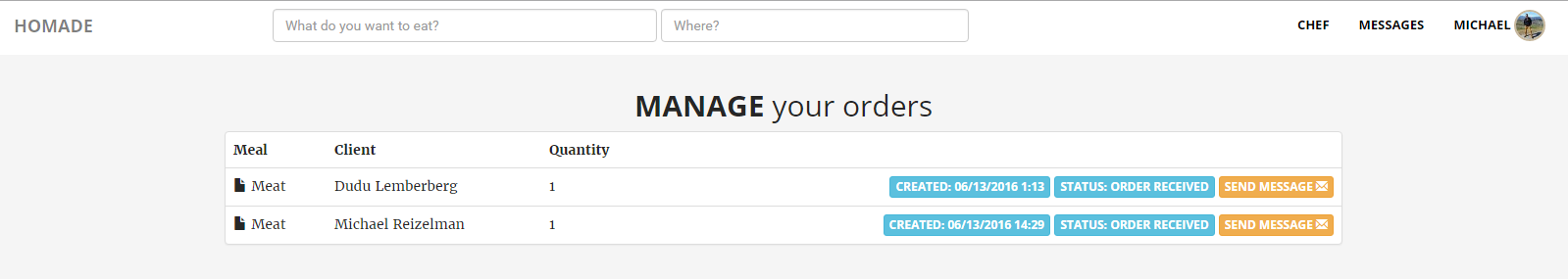
## 6.1.7 My meals

## 

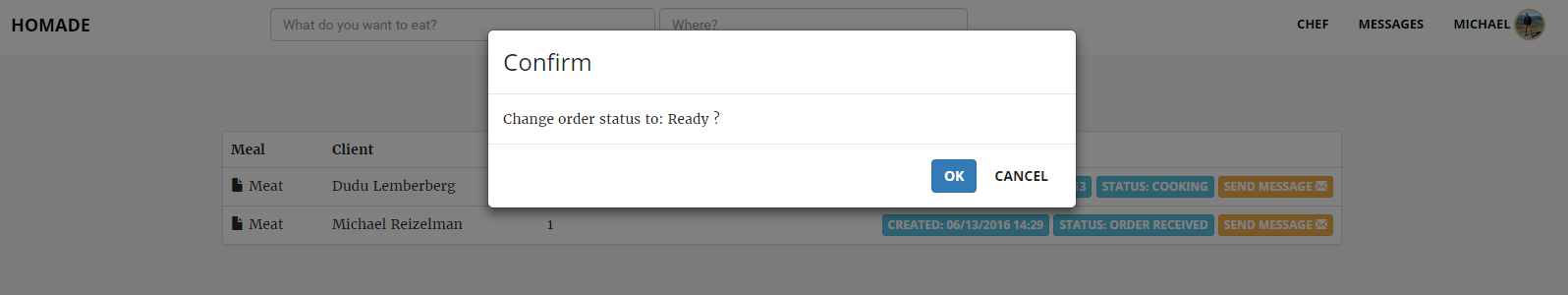
## 6.1.8 New meal



## 6.1.9 Chef orders



## 6.1.10 Chef orders – change status



## 6.1.11 Chef orders – send message

## 

## 6.1.12 Chef orders – send message

## 

# 7 קשיים במהלך הפרויקט

במהלך הפרויקט נתקלנו במספר קשיים:

1. כתיבת האלגוריתם – כאשר התחלנו לעבוד על הפרויקט "homade", מיד חשבנו כיצד ניתן יהיה לממש מנגנון שיפיק המלצות רלוונטיות עבור כל שף. לאחר מחשבה מסוימת החלטנו להשתמש באלגוריתם למידה חישובית. מייד נזכרנו בקורס כריית התונים שעברנו, והנחנו כי k-means יוכל לשרת את המטרות שלנו. הבעיה הייתה איך נוכל למצוא דרך לחלק את התוצאה של k-means לשטחים. לאחר חיפושים ומחקר קצר, מצאנו כתבה באינטרנט על שיטת voronoi לחלוקה – והבנו שהיא מתאימה בדיקות לצרכים שלנו. החל מאותו רגע נשאר רק לממש את האלגוריתם ולבדוק שהוא אכן עובד כמצופה.
2. שימוש בספריות חיצונית שאינן מיועדות לעבודה עם angular – בעת מימוש צד הקליינט בחרנו להשתמש במספר ספריות שונות שלא הייתה להן תמיכה מלאה לעבודה בסביבת angular (chart.js). נאלצנו לעתים להשתמש בפתרונות יצירתיים כדי לפתור את הבעיות שנתקלנו בהם - לרובן מצאנו פתרונות ב-stackoverflow ואתרים דומים. את שאר הבעיות הצלחנו לפתור לאחר מאמצים מסוימים.

# 8 נספח – SRS Document

מצורף בסוף התיק פרויקט.

# 9 ביבליוגרפיה

<https://en.wikipedia.org> – ויקיפדיה

<http://stackoverflow.com> – Stackoverflow

שימוש בספריות Open Source :

<https://angularjs.org/> - ספריית AngularJS