



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 3](#_Toc427238037)

[2 הקדמה – עולם הבעיה 3](#_Toc427238038)

[2.1 מטרת הפרויקט 4](#_Toc427238039)

[3 מה נעשה בתחום 4](#_Toc427238040)

[3.1 מי המתחרים שלנו 5](#_Toc427238041)

[3.2 למה אנחנו יותר טובים 6](#_Toc427238042)

[4 הצדקה אקדמאית 7](#_Toc427238043)

[4.1 השימוש באלגוריתם 7](#_Toc427238044)

[4.2 האלגוריתם 7](#_Toc427238045)

[4.2.1 הסבר כללי 7](#_Toc427238046)

[4.2.2 איך עובד האלגוריתם 7](#_Toc427238047)

[4.2.3 דוגמא 8](#_Toc427238048)

[תרשים מספר 1 – דוגמא לRandom forest 8](#_Toc427238049)

[4.3 האלגוריתם בקוד 9](#_Toc427238050)

[4.3.1 מחלקת אימון 9](#_Toc427238051)

[4.3.2 מחלקת חיזוי 10](#_Toc427238052)

[4.3.3 מחלקת בדיקה 11](#_Toc427238053)

[4.4 האלגוריתם במסכים 12](#_Toc427238054)

[5 גוף הפרויקט (DR) 13](#_Toc427238055)

[5.1 ארכיטקטורה 13](#_Toc427238056)

[5.1.1 שכבת הנתונים DataBase 13](#_Toc427238057)

[5.1.2 שכבת ה- server 13](#_Toc427238058)

[5.1.3 שכבת הclient 13](#_Toc427238059)

[תרשים מספר 2 – תרשים ארכיטקטורה 14](#_Toc427238060)

[5.2 מבנה נתונים 14](#_Toc427238061)

[תרשים מספר 3 - EER Diagram 16](#_Toc427238062)

[תרשים מספר 4 – Class Diagram 17](#_Toc427238063)

[6 אפיון המסכים 20](#_Toc427238064)

[6.1 מסכי המערכת 20](#_Toc427238065)

[תרשים מספר 5 - עץ המסכים 20](#_Toc427238066)

[6.1.1 Home Page 20](#_Toc427238067)

[6.1.2 Login 21](#_Toc427238068)

[6.1.3 Forum 22](#_Toc427238069)

[6.1.4 Add Message 22](#_Toc427238070)

[6.1.5 Calendar 23](#_Toc427238071)

[6.1.6 Committee Status...................................................... 23](#_Toc427238072)

[6.1.7 Users Information 24](#_Toc427238073)

[6.1.8 Suppliers Information 24](#_Toc427238074)

[6.1.9 Faults Information 25](#_Toc427238075)

[6.1.10 About As 25](#_Toc427238076)

[7 קשיים במהלך הפרויקט 26](#_Toc427238077)

[8 נספח – SRS Document 28](#_Toc427238078)

[9 ביבליוגרפיה 28](#_Toc427238079)

# 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**

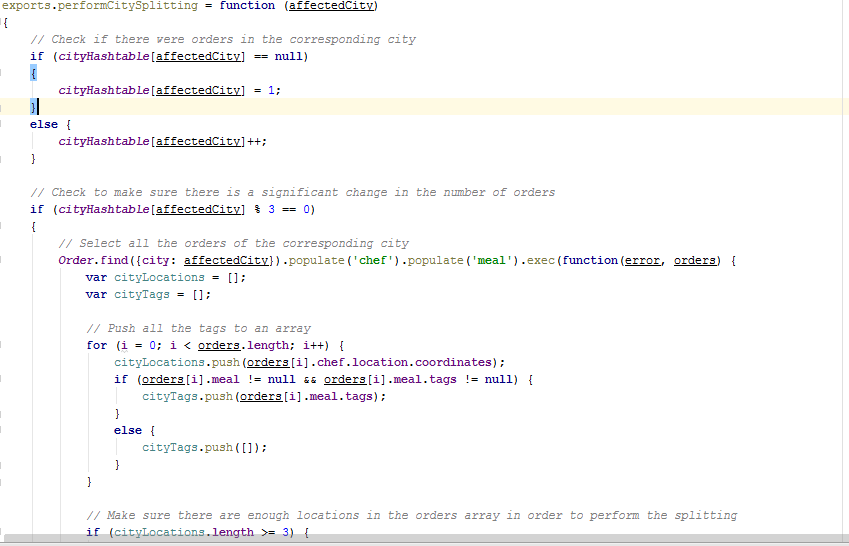
זוהי שיטת חילוק של שטחים שבה כל נקודה בתוך גוף תוחם מיוחסת לשטח מסוים ע"פ הקרבה שלה לאחת הנקודות (המרכזים מה-kmeans).

## 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

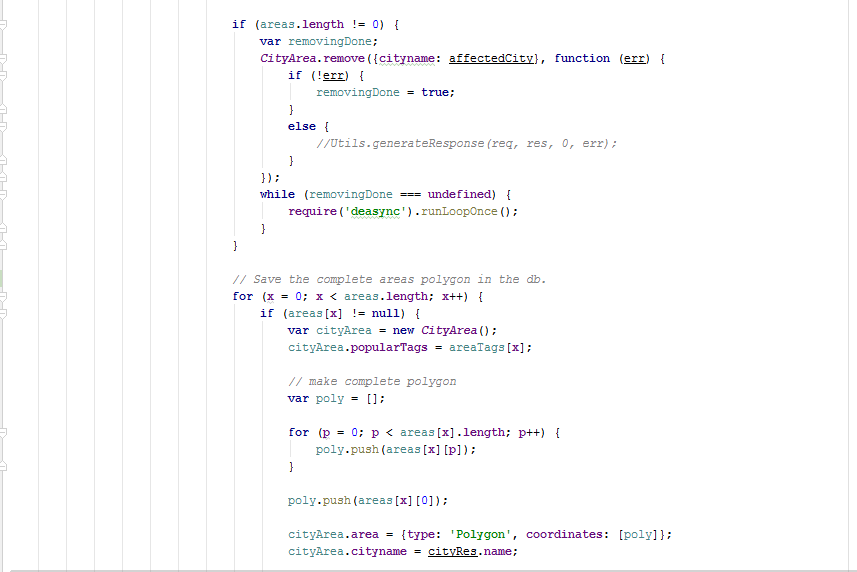
4.3.2 ביצוע החיתוך עם K-means

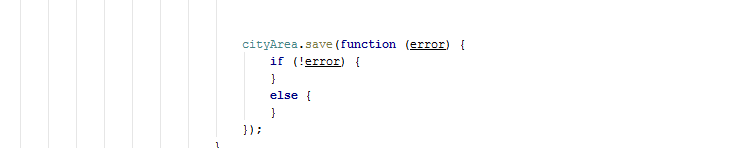


4.3.3 חלוקה לשטחים עם voronoi

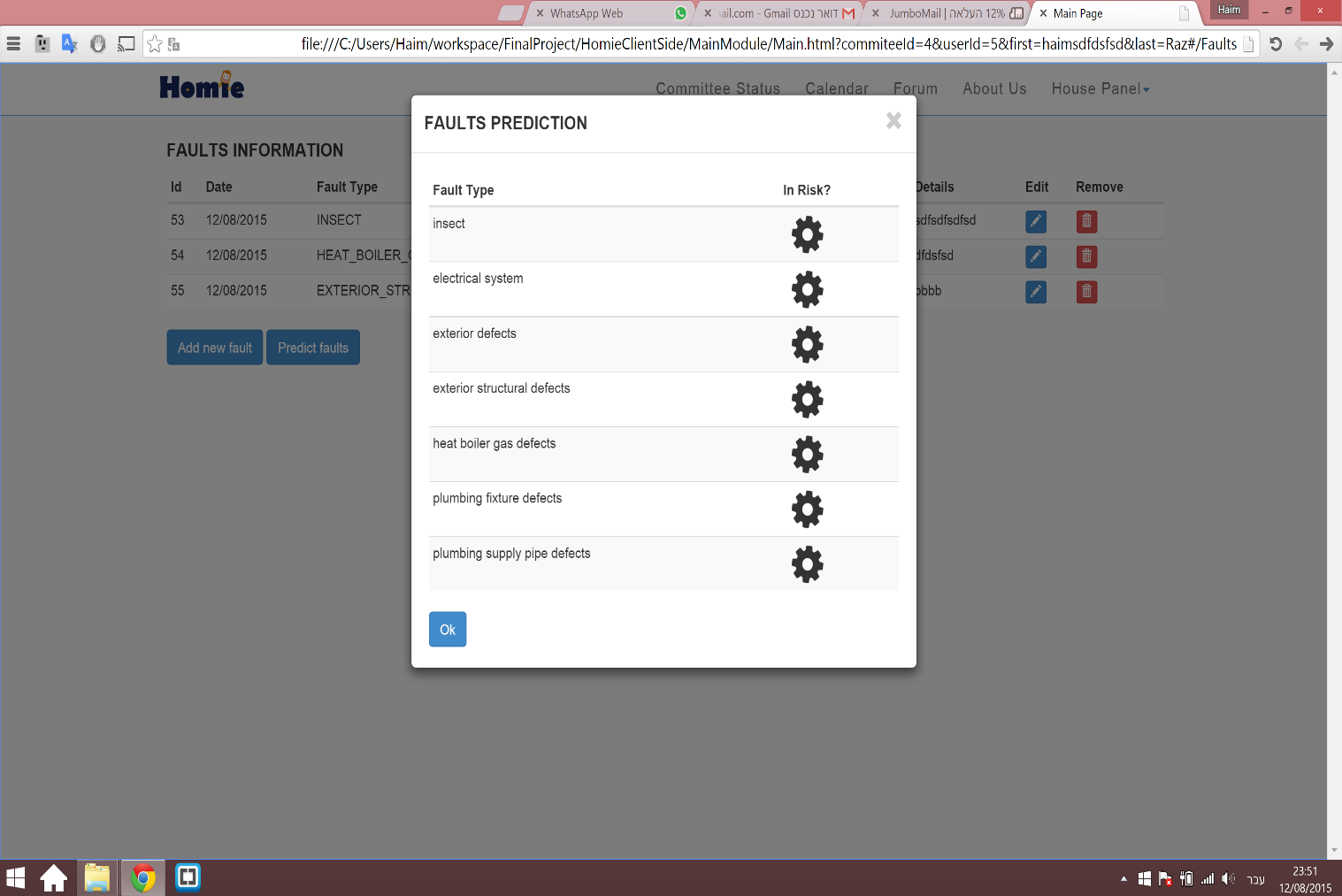


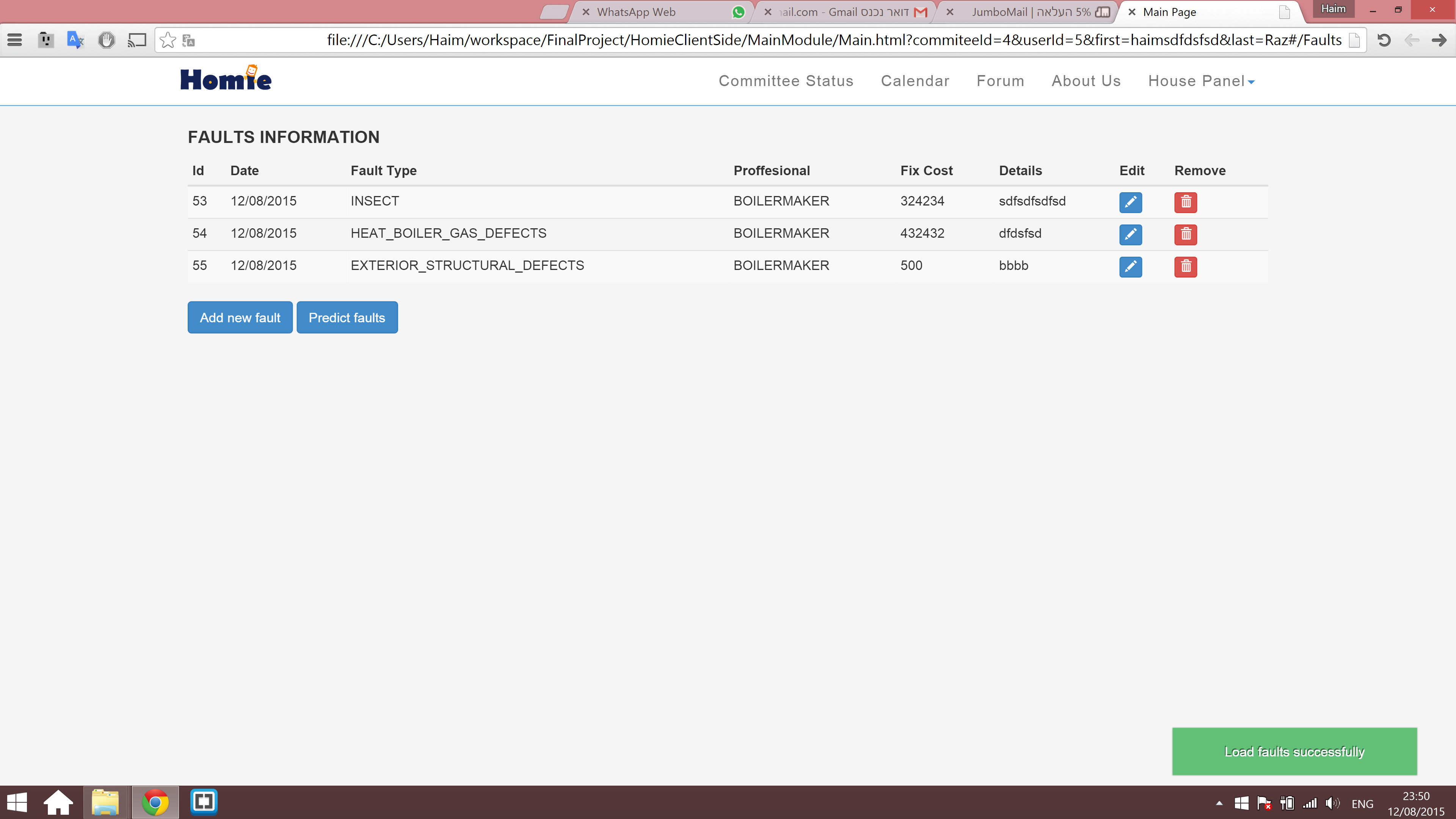
4.3.3 עדכון השטחים המשויכים לעיר ב-DB





* 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 הינו בmysql האפליקציה שומרת את כל הנתונים אודות המשתמש , האירועים המתנהלים, הודעות הפורום, וכמובן כל היסטורית התקלות של כל בניין.

## 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 להצגה.

בנוסף השכבה אחראית על הוספה והסרה של נתנים מהDB

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

עוד מתפקיד הserver הוא לאמת את הנתונים המתקבלים )האם הם תקינים ואינם קוד זדוני).

## 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 וקל לתפעול,

בנוסך במערכת שלנו שמנו דגש על ה- UI להיות ""responsive ולכן על האלמנטים המצוירים על המסך חייבים לתמוך בכל גודל מסך (בין אם זה מחשב , טאבלט , או כל סוג של סמארטפון).

### תרשים מספר 2 – תרשים ארכיטקטורה

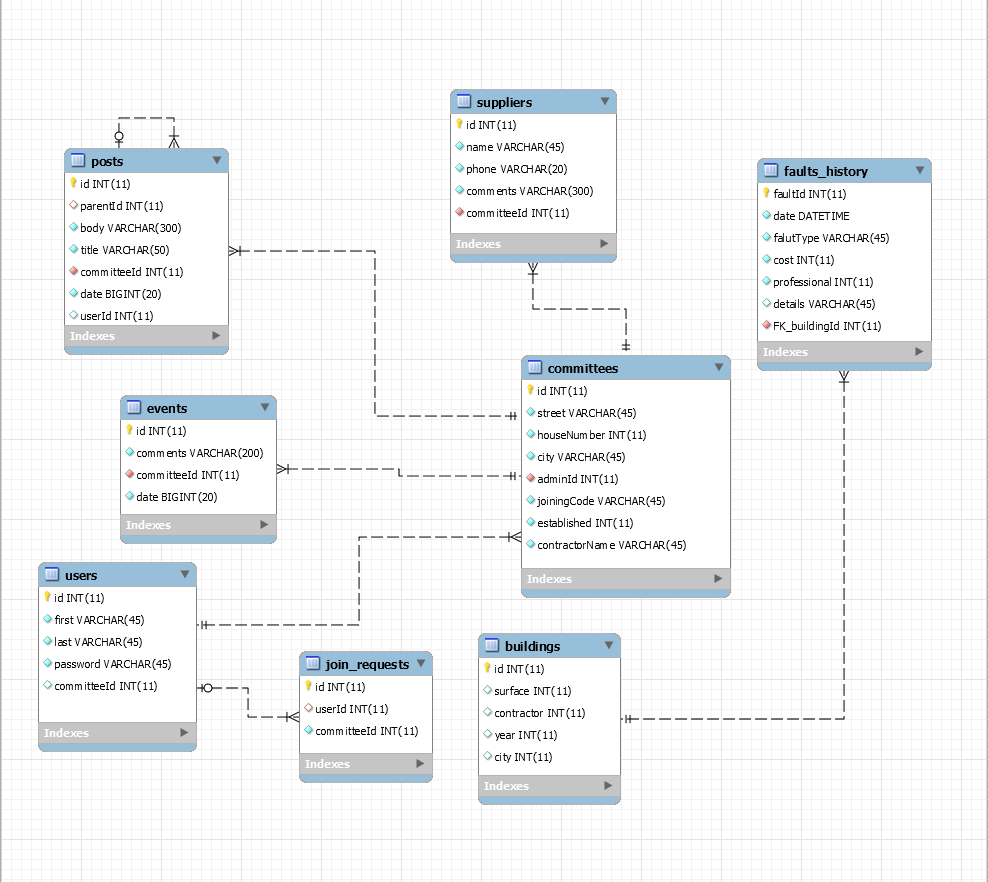


5.2 מבנה נתונים

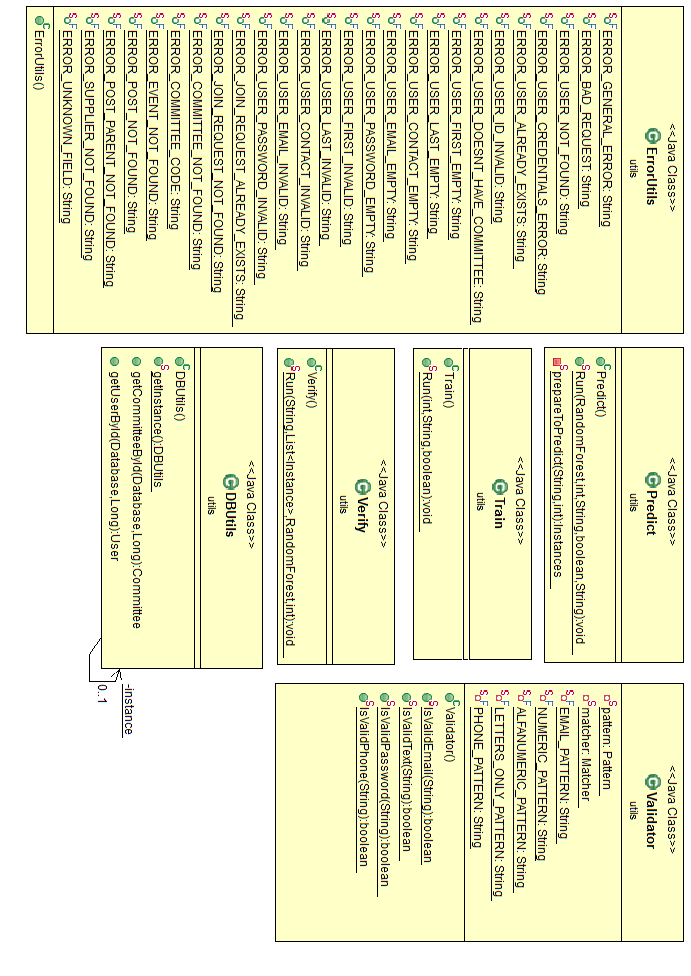
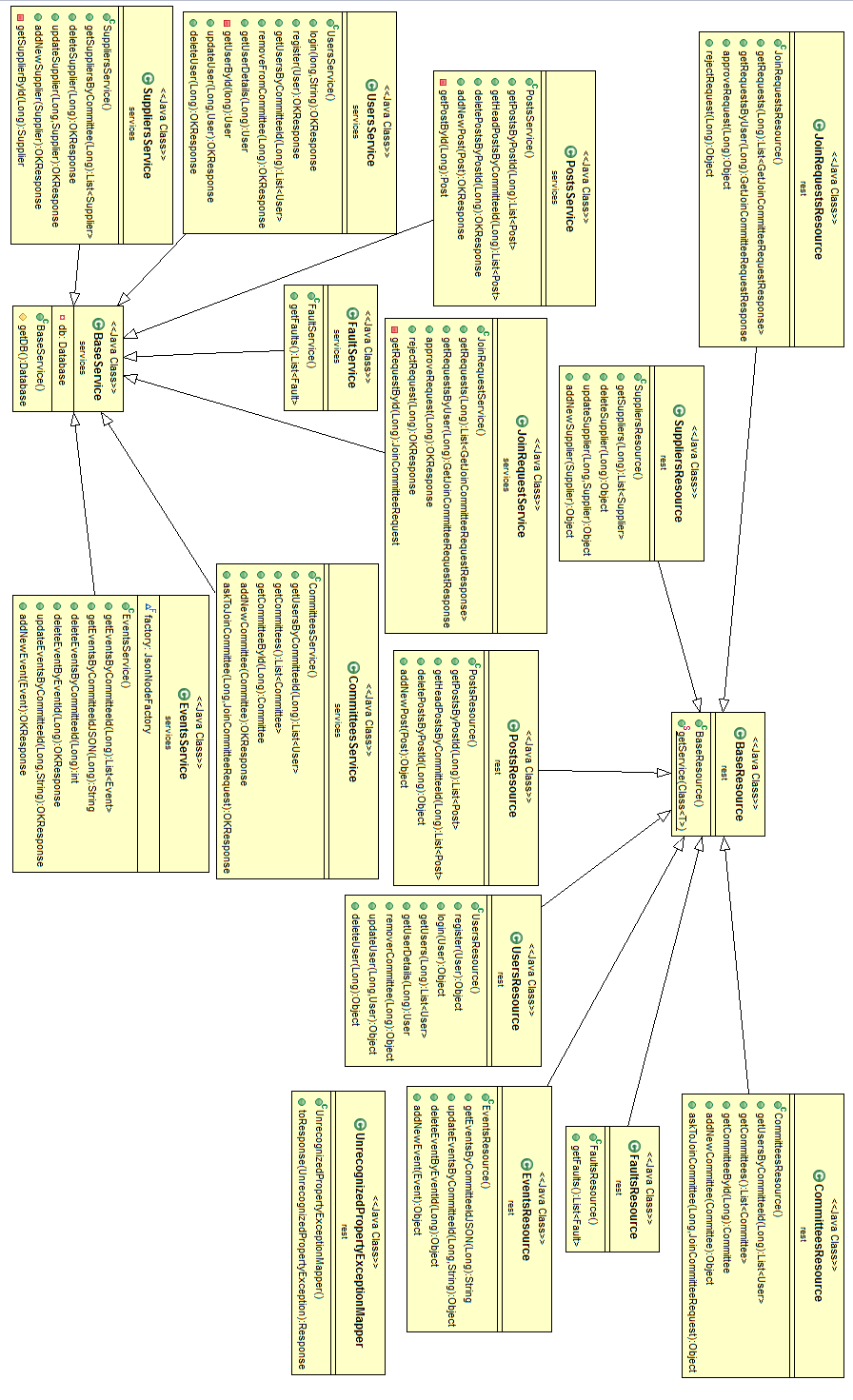
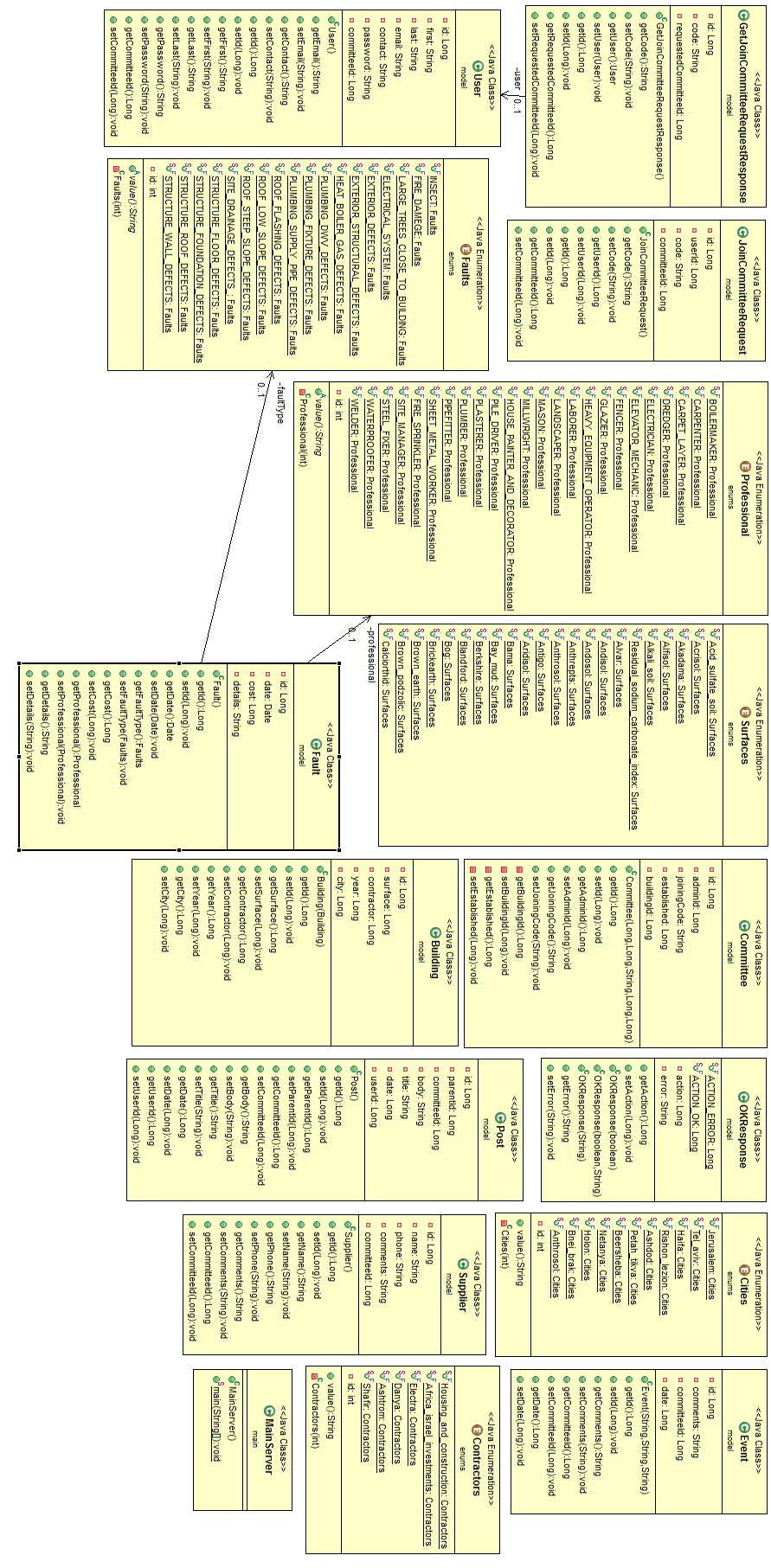
בסיס הנתונים מכיל את הטבלאות הבאות:

* Users – טבלת משתמשים, הטבלה מכילה את כל משתמשי המערכת ובנוסף את הנותנים עבור על משתמש.
* Join-Request – טבלת בקשות, הטבלה מכילה את המידע עבור כל משתמש לאיזה וועד הוא משתייך, הטבלה נועדה כדי לאמת האם המשתמש באמת שייך לוועד על מנת למנוע התחזות לחלק מהבניין.
* Committees – טבלת הוועדים, הטבלה מכילה את כל הוועדים הקיימים במערכת ובנוסף את המידע עבור כל וועד.
* Events – טבלת האירועים, הטבלה מכילה את כל האירועים מלוח השנה, ובנוסף את כל המידע עבור כל אירוע. כל אירוע מקושר לוועד המתאים וכך כל וועד צופה רק באירועים שלו (מקושר לטבלת Committees).
* Posts – טבלת ההודעות בפורום, הטבלה מכילה את כל ההודעות בפורום ובנוסף את כל המידע עבור כל הודעה בפורום. כל הודעה מקושרת לוועד המתאים וכך על וועד צופה רק בהודעות שלו בפורום (מקושר לטבלת Committees).
* Suppliers – טבלת אנשי המקצוע, הטבלה מכילה את כל אנשי המקצוע במערכת ובנוסף את כל הפרטים עבור כל איש מקצוע. כל איש מקצוע מקושר לוועד בו של הבניין בו הוא עובד, וכך ניתן לדעת איזה אנשי מקצוע וועד הבית מעסיק עבור הבניין.
* Buildings – טבלת הבניינים, הטבלה מכילה את כל הבניינים במערכת ובנוסף את הנתונים עבור כל בניין
* Faults history- טבלת התקלות, הטבלה מכילה את פירוט התקלות עבור כל בניין (מקושר לטבלת Buildings).

### תרשים מספר 3 - EER Diagram



### תרשים מספר 4 – Class Diagram



# 6 אפיון המסכים

6.1 מסכי המערכת

### תרשים מספר 5 - עץ המסכים

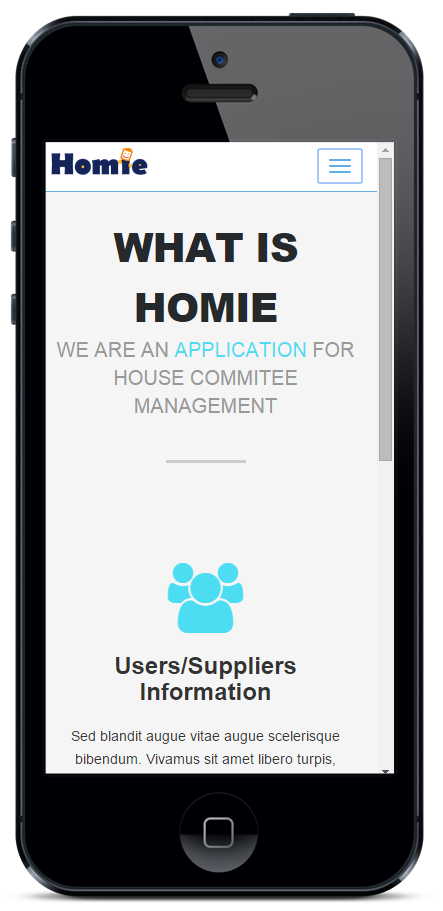
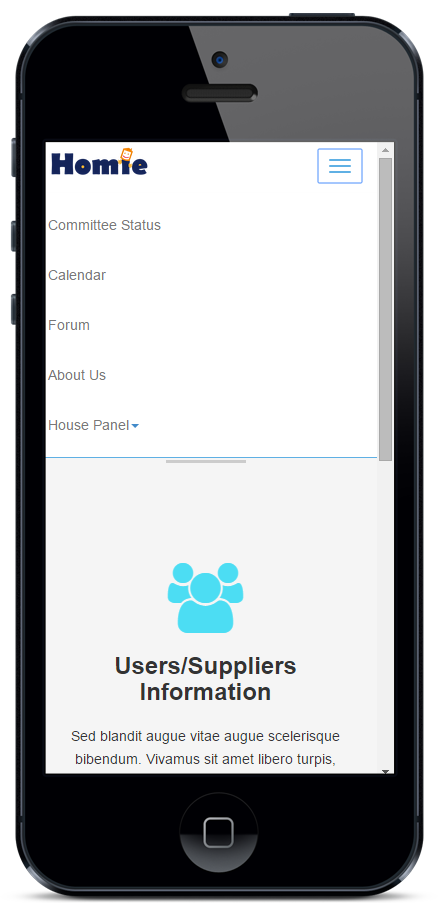
גישה

לתפריט

בכל חלון

באתר

## 6.1.1 Home Page

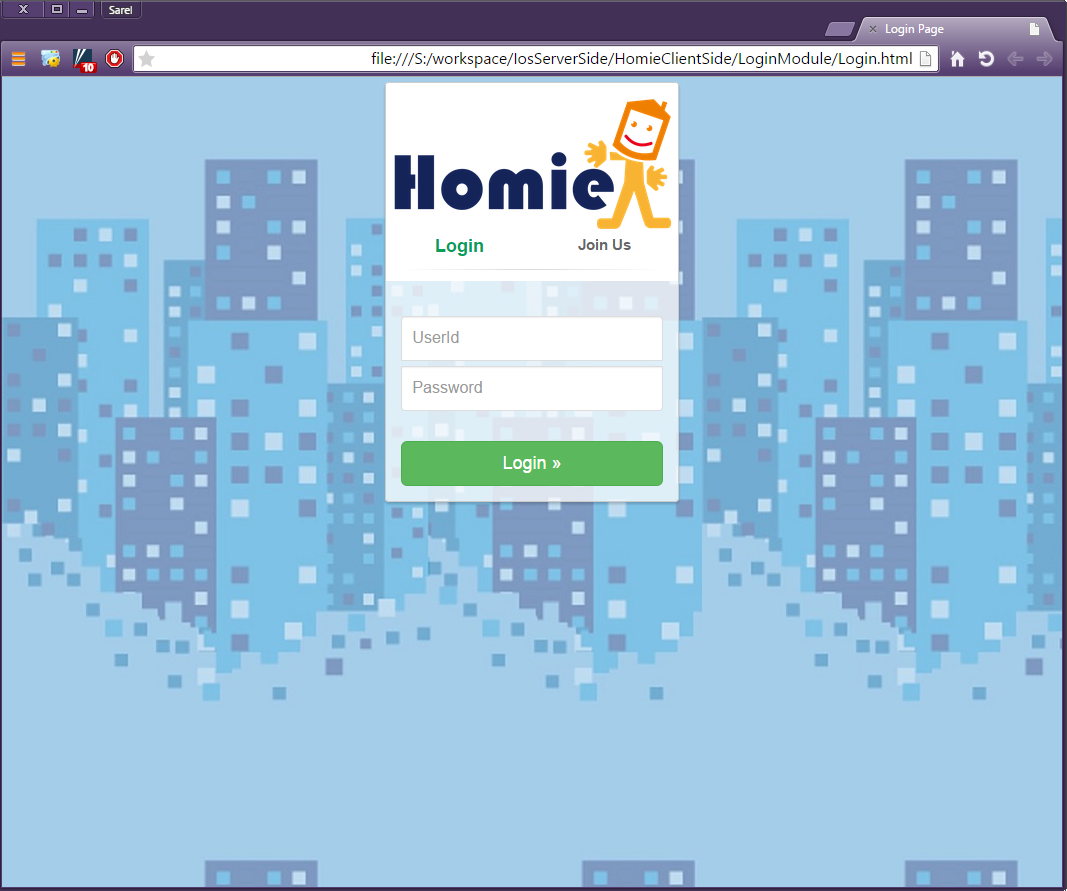


גישה לתפריט בכל חלון

באפליקציה

## 6.1.2 Login

הכנסת שם משתמש וסיסמא



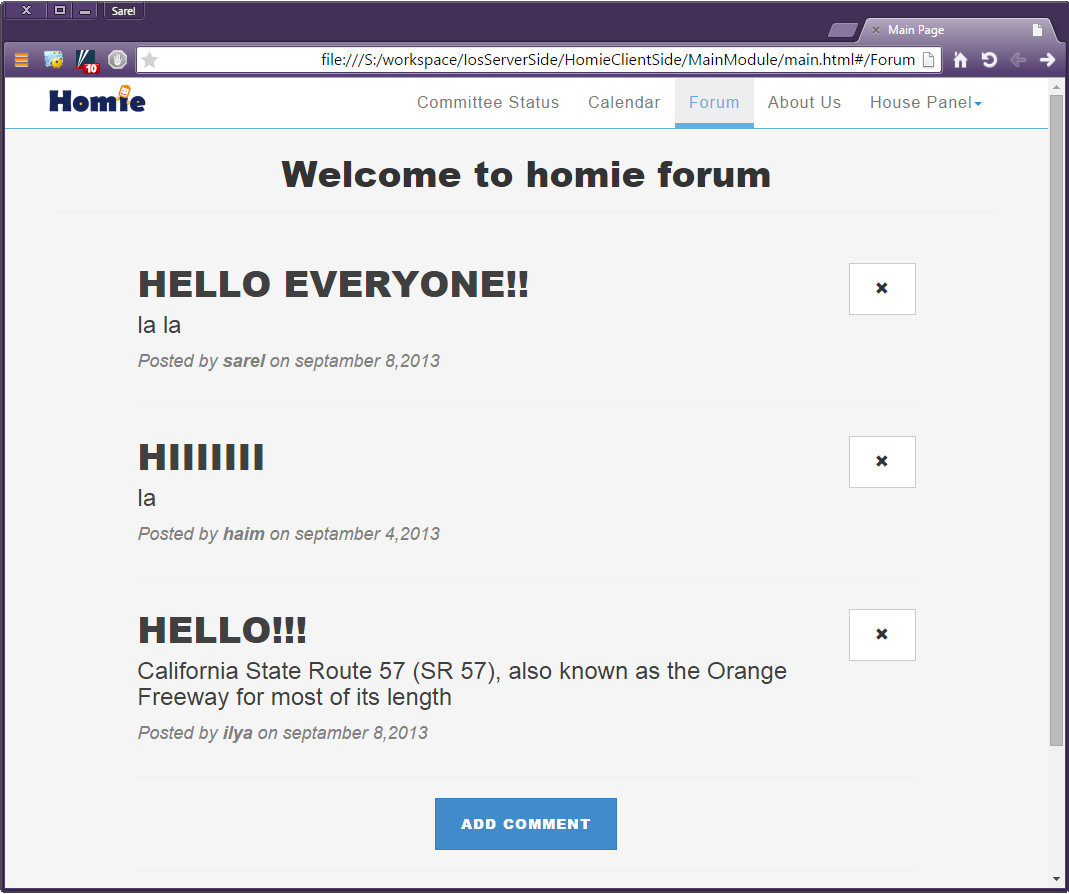
\*\*\*\*\*\*\*

hadas

\*\*\*\*\*\*\*

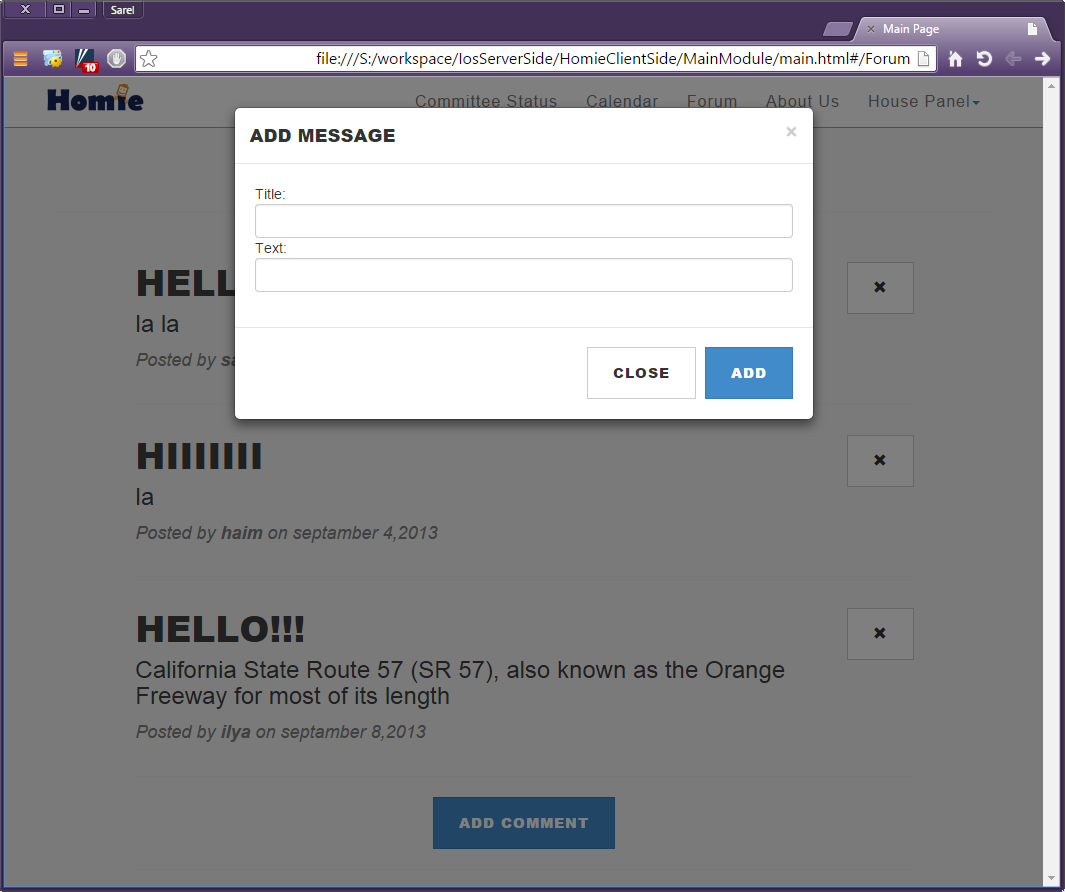
hadas

## 6.1.3 Forum

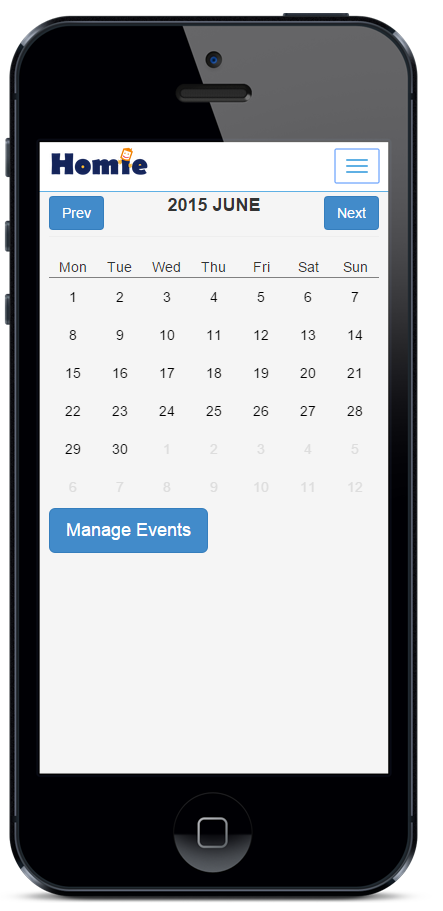


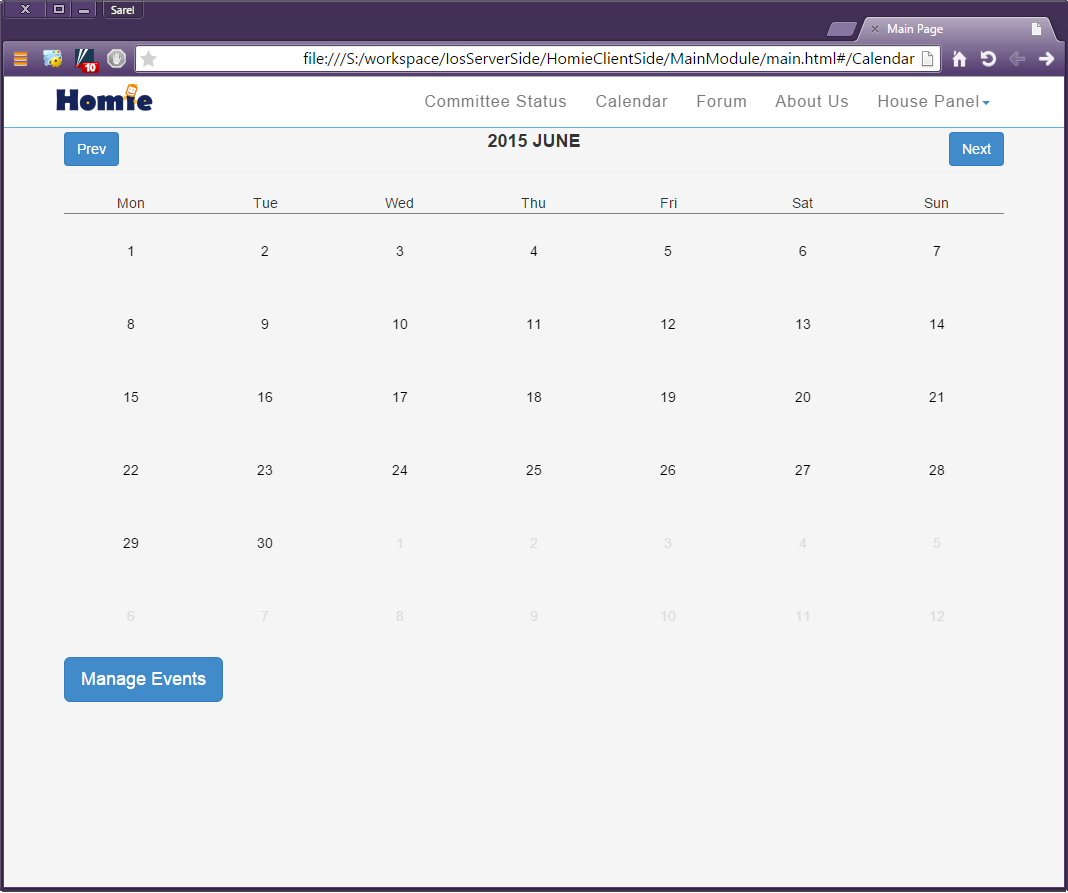
מחיקה

## 6.1.4 Add Message

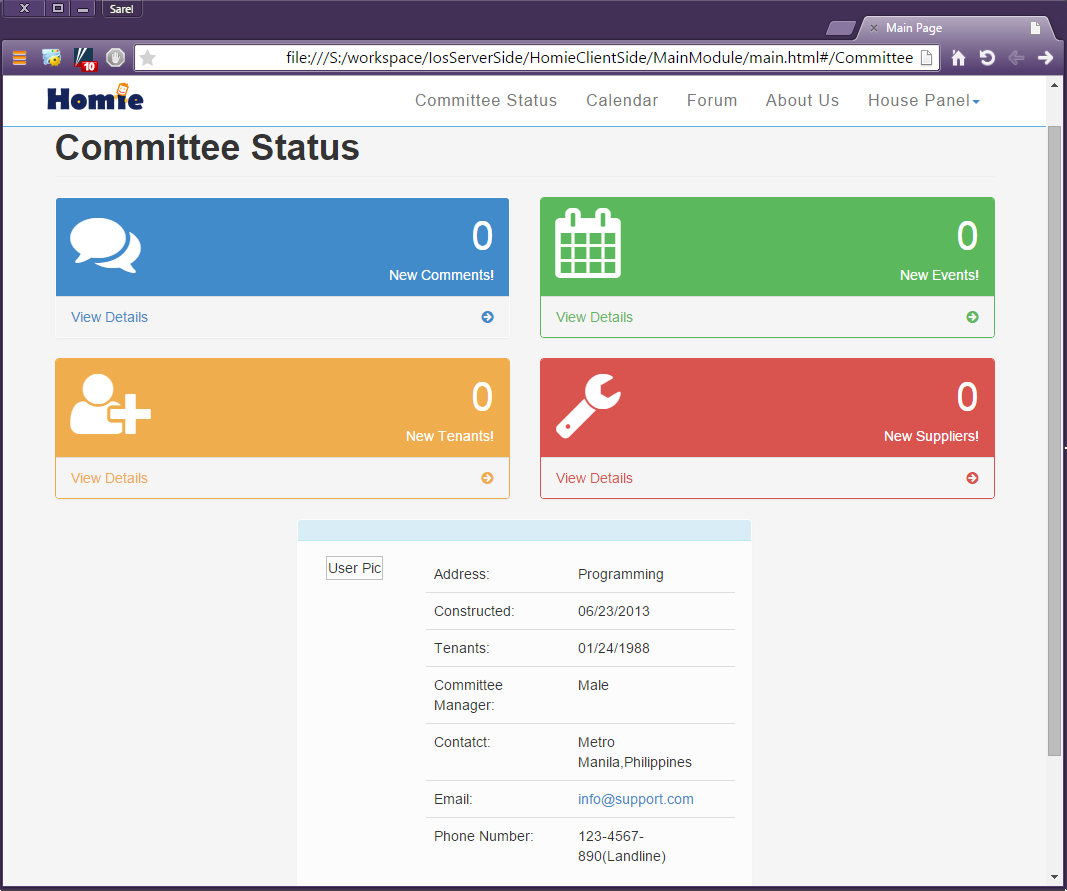
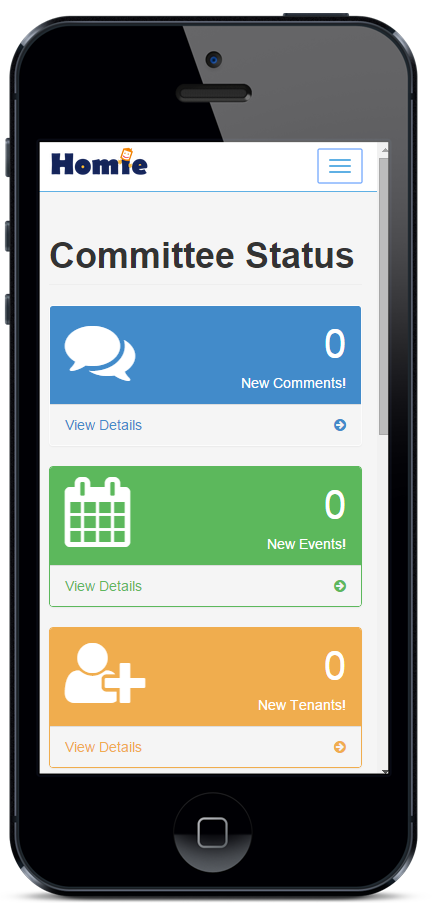


## 6.1.5 Calendar

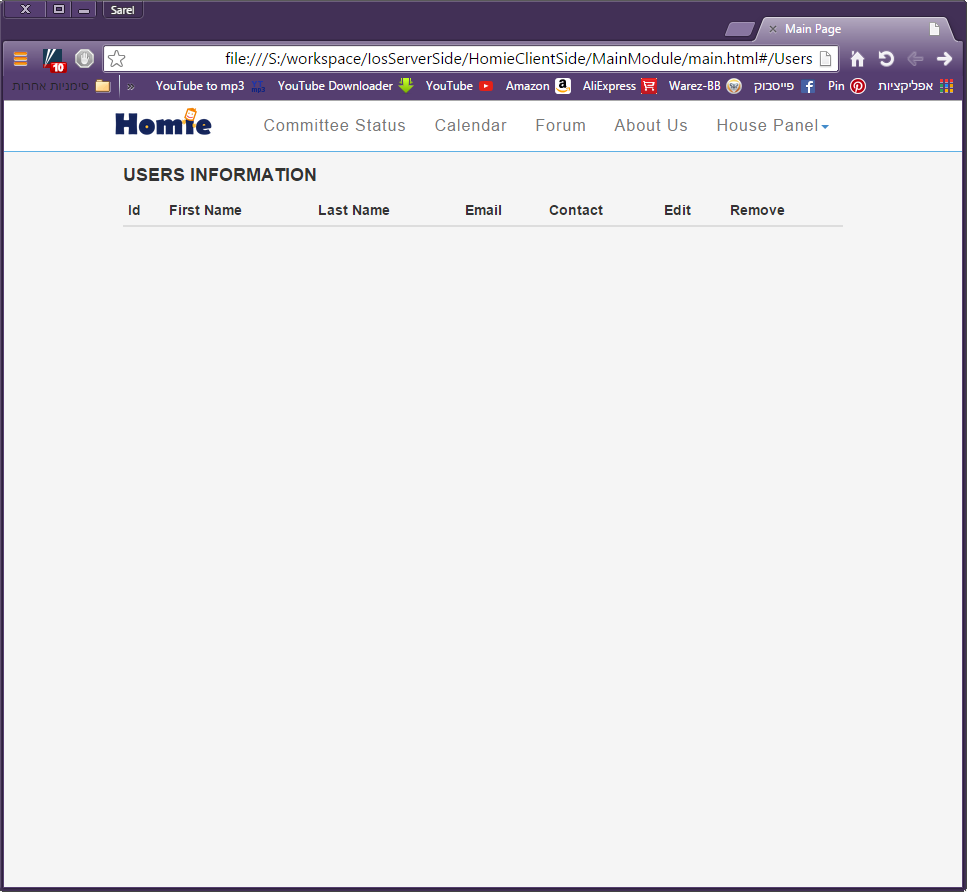




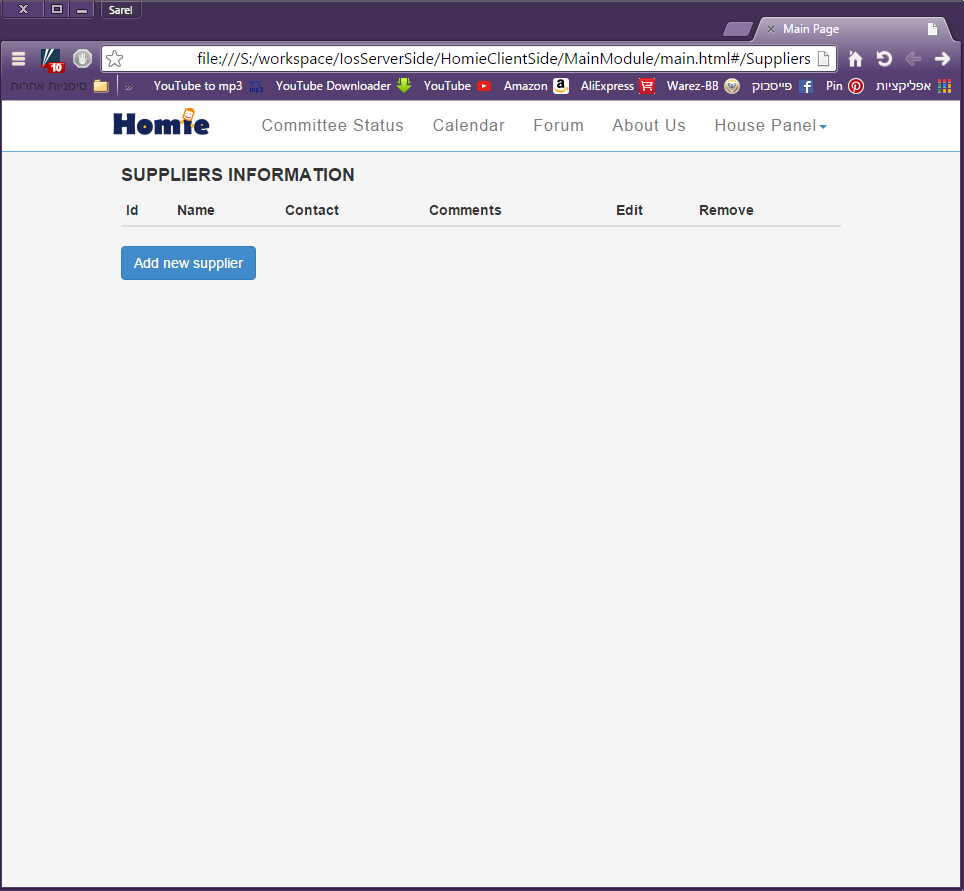
## 6.1.6 Committee Status



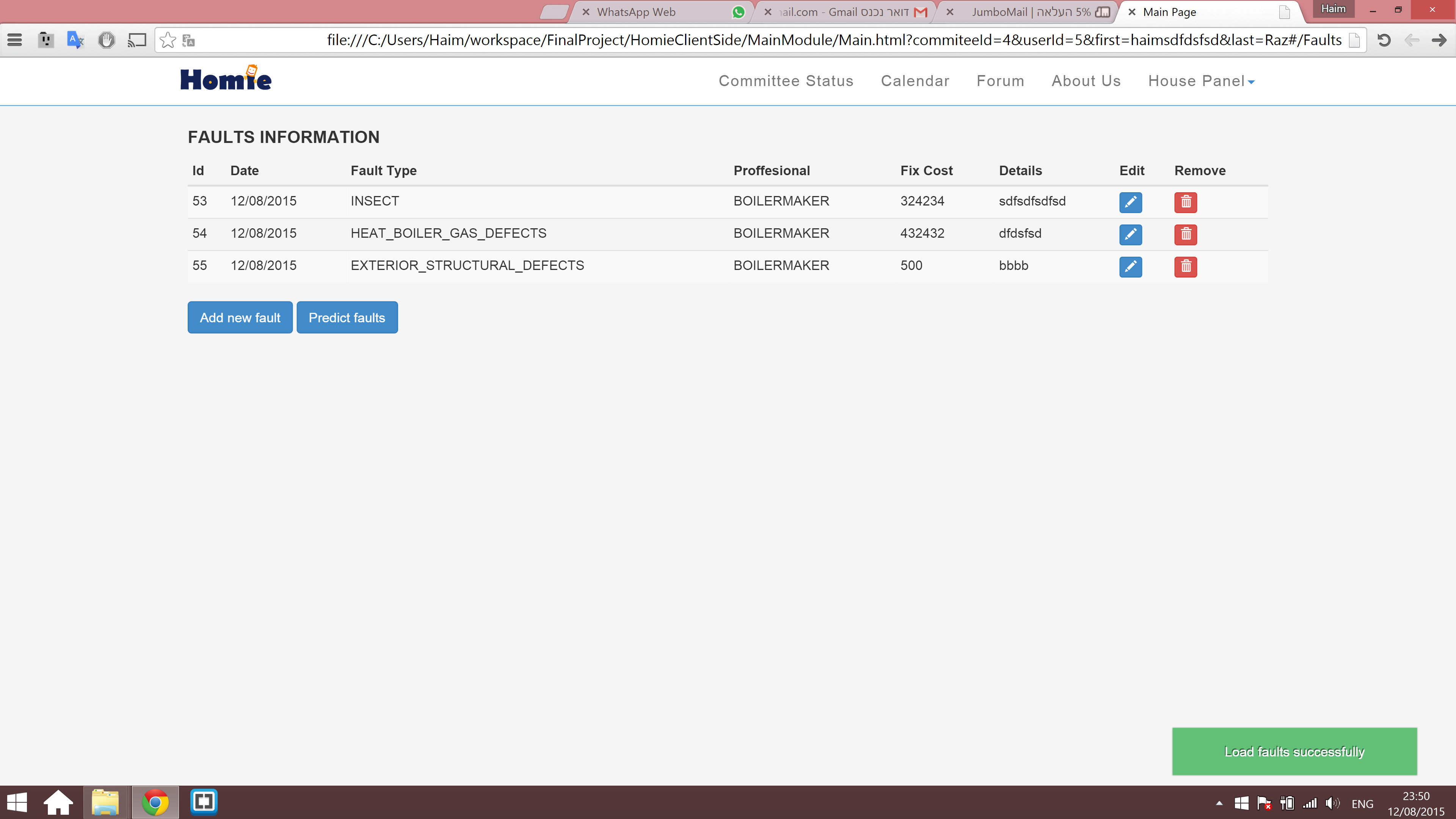
## 6.1.7 Users Information



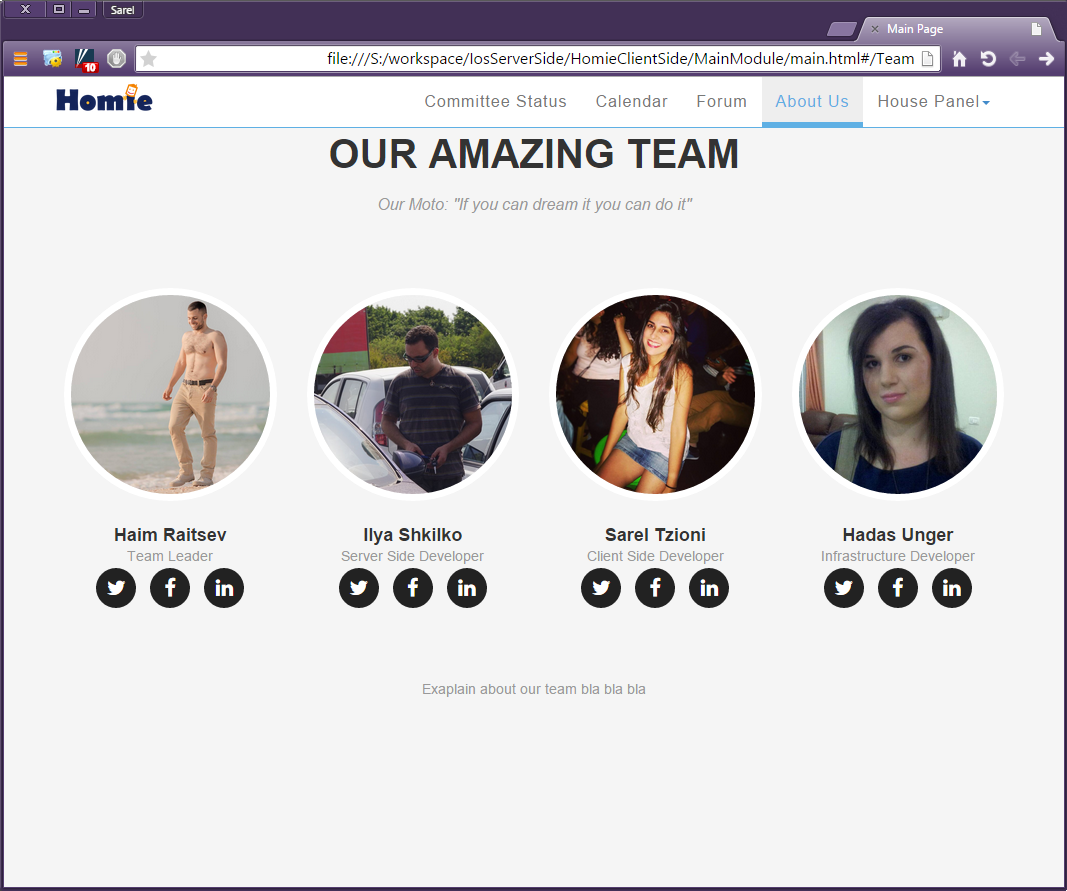
## 6.1.8 Suppliers Information



6.1.9 Faults Information



## 6.1.10 About As



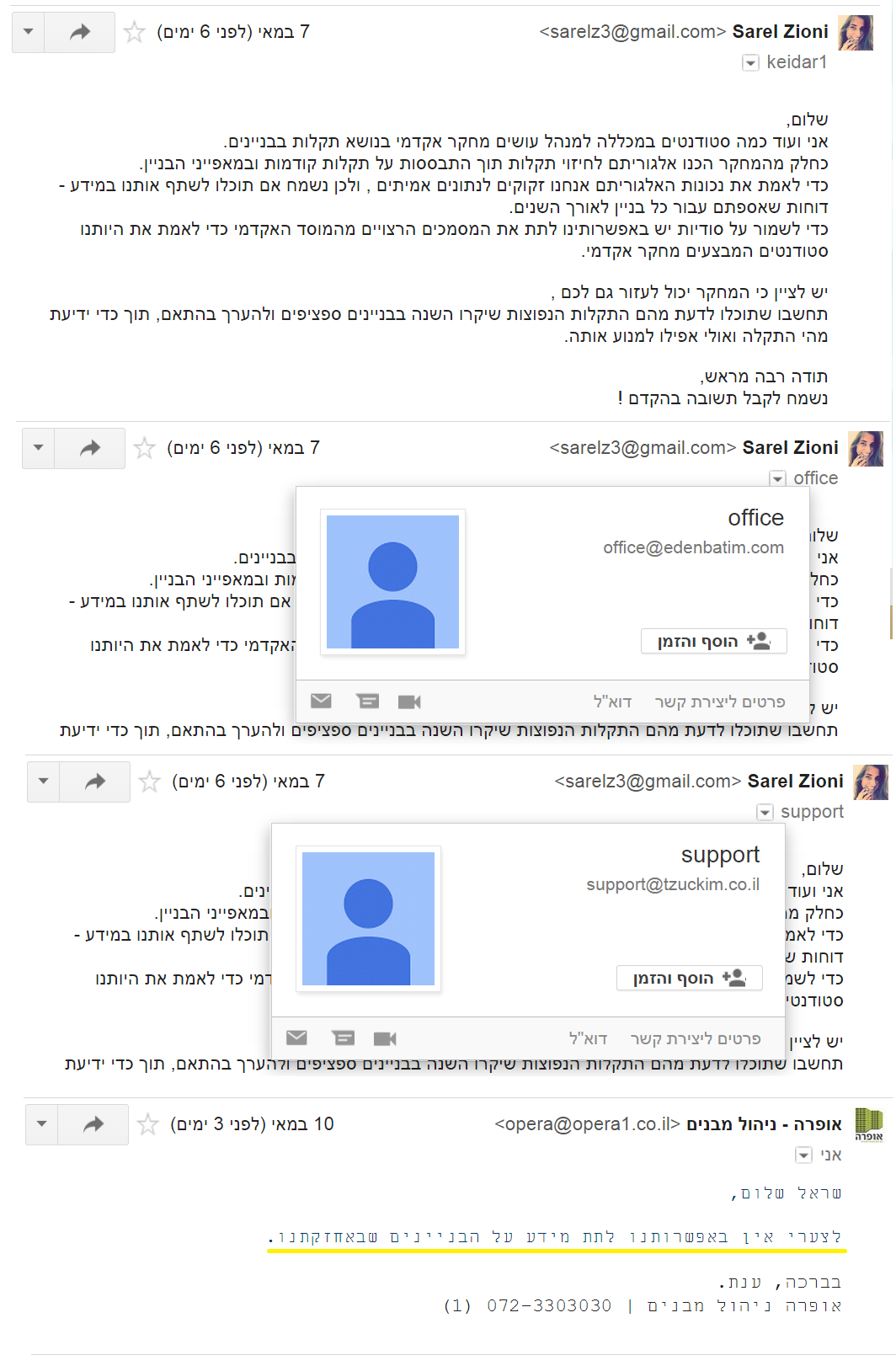
# 7 קשיים במהלך הפרויקט

במהלך הפרויקט נתקלנו במספר קשיים

1. כתיבת הפרויקט במערכת IOS – כאשר התחלנו לעבוד על הפרויקט "homie" לאחר שבסמסטר הראשון למדנו את הקורס IOS התלהבנו מהטכנולוגיה החדשה וחשבנו שהיא תהיה מתאימה, בתחילת סמסטר ב עלו המון קשיים במהלך העבודה עם מערכת IOS כשבראש הבעיות, הבעיה העיקרית הייתה שרצינו להגיע לכמה שיותר משתמשים ודרך ה- appStore הגענו לנתח יחסית קטן, מעבר לכך ככול שהתקדמנו עם הפרויקט הבנו כי חייבת להיות גם אפליקציה אינטרנטית המאפשרת למשתמש לגשת לhomie דרך הדפדפן. ולכן לאחר התלבטויות רבות החלפנו את שכבת המשתמש במערכת (UI) לטכנולוגית Web כמפורט בתקציר
2. איסוף נתונים לData Mining – כאשר החלטנו לבדוק תקלות של בנינים היה ברצוננו לקחת נתונים אמתיים, על מנת לעשות זאת שלחנו לכמה חברות תחזוקה מייל לגבי נתונים אשר הם יכולים לשתף אותנו בהם אודות תקלות שהם פתרו בבנייינים

לצערנו הרב לא הצלחנו להשיג נתונים בדרך הזו, ולבסוף עברנו כמה בניינים ושאלנו את ועדי הבית אודות תקלות שהתרחשו בבניין בנוסף לכך חקרנו באינטרנט אודות תקלות וכך הזנו נתונים מבוססי נתוני אמת.

\*מצורף: תיעוד המיילים



1. בחירת האלגוריתם לData mining – לאחר שאספנו את הנתונים הנדרשים על ידי ניסויי ותהייה של כמה אלגוריתמים החלטנו להשתמש באלגוריתם שהחזיר את תוצאות הדיוק הטובות ביותר

להלן האלגוריתמים שניסינו :

* J48
* BayesNet
* SVMreg

לבסוף החלטנו על האלגוריתם : Random forest

# 8 נספח – SRS Document

מצורף בסוף התיק פרויקט.

# 9 ביבליוגרפיה

<https://en.wikipedia.org> – ויקיפדיה

<https://msdn.microsoft.com> – msdn

<http://stackoverflow.com> – Stackoverflow

<https://www.youtube.com> – youtube

שימוש בספריות Open Source :

<http://www.cs.waikato.ac.nz/ml/weka/> - ספריית weka

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