



# LOST ANIMAL

## ספר הפרויקט

### מגשים

רן גרמן 304862733  
אלחנן סמואל 040316911

מנחה: אילן קירש

## תוכן

2	תיאור הפרויקט
3	תיאור פונקציונלי
3	מסך כניסה
4	מסך הרשמה
6	פרופיל
8	עריכת פרופיל
9	מסך החיפוש
10	דפדוף במודעות
5	יצירת מודעה חדשה
6	שליחת הודעות
11	סוכן חכם
11	התראה על הודעות
12	טכנולוגיות המערכת
12	Google maps auto complete API
13	BootStrap android
13	NODE.JS – מבוסס java script -לשימוש ב firebase cloud function
13	FCM -firebase colud messaing
14	תיאור בסיס הנתונים
14	סביבת עבודה
14	תיאור סכימת בסיס הנתונים
18	Machine learning - ולמה החלטנו שלא להוסיף מודול שתומך בכך
22	חלוקת עבודה:

## תיאור הפרויקט:

בחרנו ליצור אפליקציית אנדרואיד המסייעת לאנשים שאיבדו את בעל החיים שלהם.

האפליקציה באה לשרת הן אנשים שאיבדו את בעל החיים שלהם, והן כאלו שמצאו בעל חיים ולא יודעים למי הוא שייך.

מרבית האפליקציות בנושא מתמקדות בהתקן GPS-סלולרי לכלב, שלא רלוונטי עבור כלבים שאבדו או נמצאו.

כיום אין בשימוש אפליקציות דומות, והפלטפורמה הנפוצה בזירה הזאת היא הרשתות החברתיות (בראשן פייסבוק/וואטסאפ) בה ניתן לשתף מודעות שכאלה. החיסרון בפייסבוק הוא ריבוי קבוצות, וכדי לחבר בין מודעת כלב אבד, לבין מי שמחפש אותו יש צורך להצליב מידע בין קבוצות (מקרה שקרה לאחד המפתחים, ומכאן הרעיון ליצור את האפליקציה). האפליקציה עונה בדיוק על הצורך הזה – הצלבה בין מודעות חיפוש בעלי חיים ובין בעלי חיים שנמצאו בדגש על זמן הפרסום ומקום האובדן/מציאה. אפליקציה כלל עולמית שמשרתת משתמשים מקומית.

באפליקציה מאגר מידע, שהחלקים העיקריים בו הן רשימת בע"ח שנמצאו, ורשימת בעלי חיים שאבדו.

פרסום מודעה חדשה מתבצעת בקלות ומהירות.

בנוסף, יש אפשרות סינון מודעות לפי:

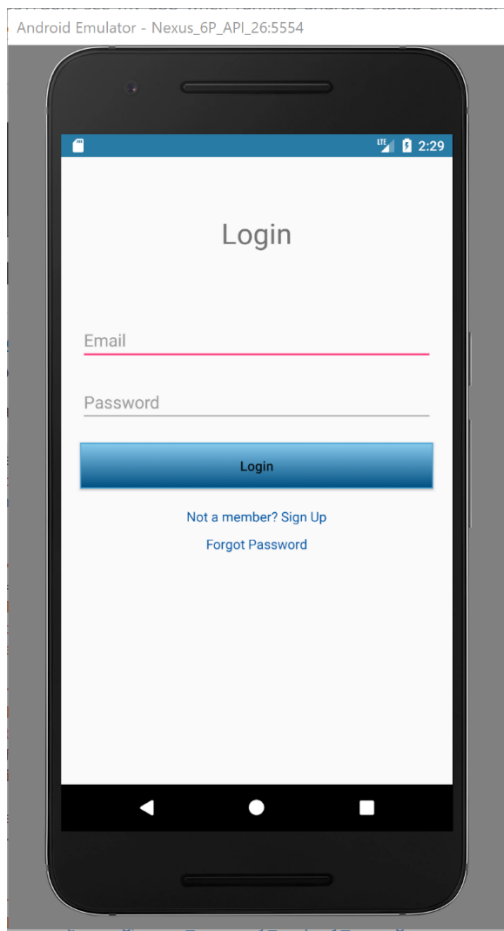
- חיפוש ברשימת בע"ח אבודים או בע"ח שנמצאו
- חיפוש לפי סוג בע"ח (כרגע התמקדנו לצורך פשטות חווית המשתמש בכלב/חתול/אחר)
- חיפוש לפי תאריך, חיפוש לפי שעה
- חיפוש לפי מיקום (מרכז החיפוש ורדיוס חיפוש בק"מ)

בנוסף לחיפוש, הוספנו אפשרות לשמור את מאפייני החיפוש בשרת, ובכל פעם שבסיס הנתונים מתעדכן (למעשה בעל פעם שמפורסמת הודעה) מופעלת בשרת פונקציית סינון ואם פורסמה מודעה חדשה הרלוונטית למשתמש מסוים (בדגש על סוג בעל החיים ואזור החיפוש) אז המשתמש יקבל התראת PushNotification בתוך כמה שניות מפרסום המודעה.

בנוסף, הוספנו מערכת להתכתבות בין המשתמשים כאשר התכתבות מתחילה דרך ההודעה המפורסמת (פניה למפרסם ההודעה) - גם כן עם אופציית PushNotification.

## תיאור פונקציונלי

### מסך כניסה:



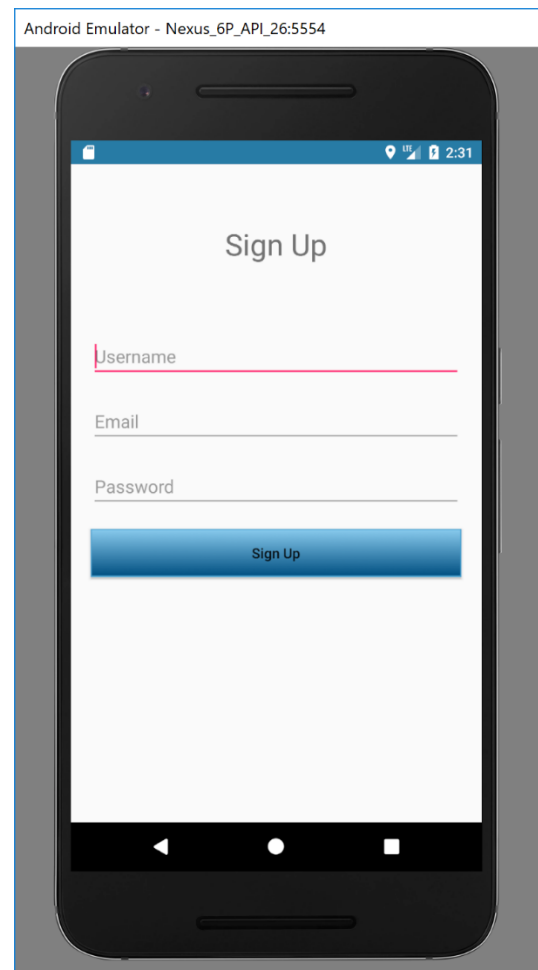
כאשר נכנסים לאפליקציה בפעם הראשונה מגיעים למסך זה.

במידה והמשתמש כבר רשום, עליו להזין את כתובת המייל והסיסמא, ומרגע זה ואילך, בכל פעם שהמשתמש יכנס לאפליקציה הוא יגיע אוטומטית למסך הבית (בלי צורך להזין סיסמא כל פעם מחדש).

אם המשתמש רשום אך לא זוכר את הסיסמה, הוא יכול ללחוץ על "שכחתי סיסמא" ואז ישלח לכתובת המייל שלו קישור לאיפוס ססמה.

משתמש חדש יכול ללחוץ על "הירשם" ובכך לעבור למסך ההרשמה.

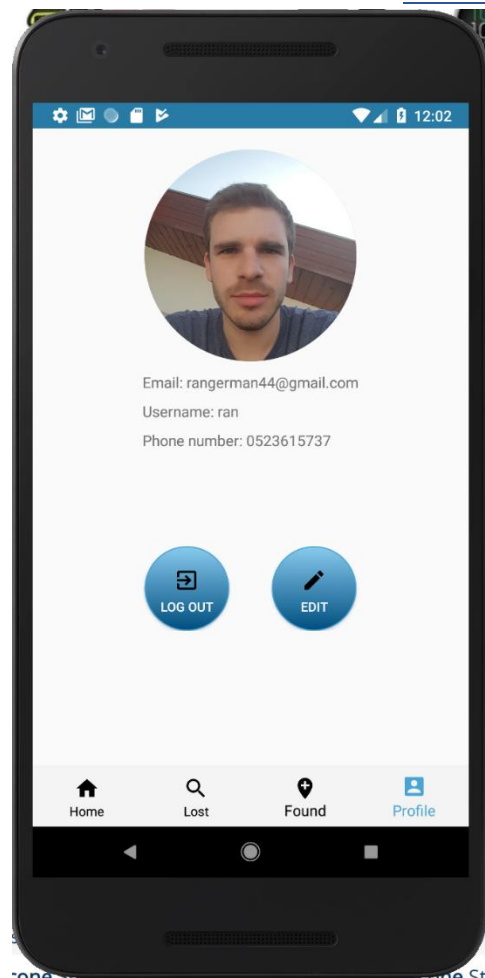
## מסך הרשמה:



במסך זה משתמש חדש נרשם למערכת, כאשר הפרטים שיש למלא הם: שם משתמש, אימייל וסיסמא.

לאחר לחיצה על כפתור "הרשם" הנתונים שהוכנסו ייבדקו (כתובת אימייל חוקית וייחודית, כך שלא ניתן להירשם פעמיים מאותה כתובת, סיסמא בעלת לפחות 6 תווים, לא נשאר אף שדה ריק), במידה ומשהו נמצא לא תקין, יסומן השדה המתאים ותוצג הודעה שמודיעה מה לא בסדר. במידה והכל תקין, תוצג הודעת אישור והמשתמש יועבר למסך הראשי.

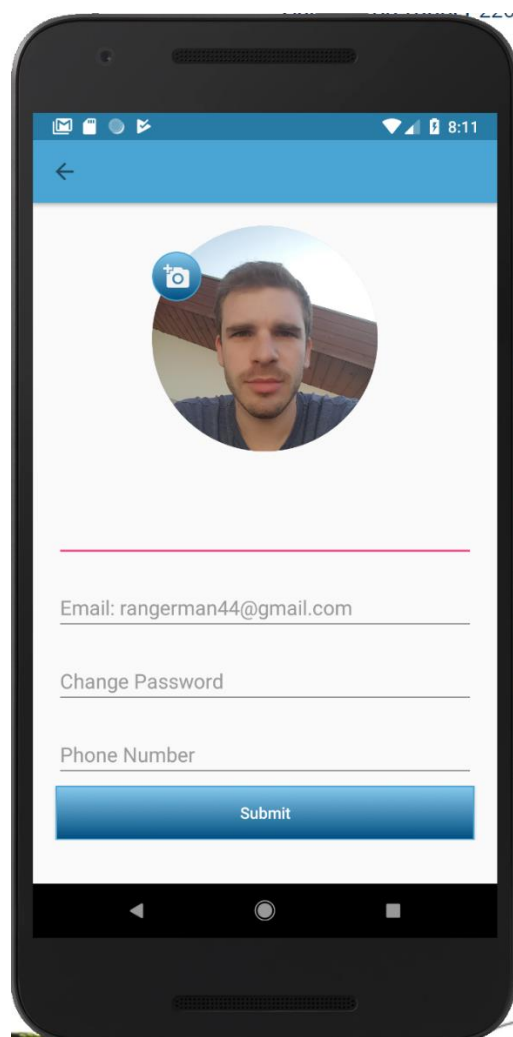
פרופיל:



במסך הפרופיל רואים את כל פרטי המשתמש, וישנם שני כפתורים – ערכית פרופיל והתנתקות. במידה והמשתמש בחר להתנתק, יפתח חלון קטן שיבדוק שאכן המשתמש רוצה להתנתק ולא לחץ על הכפתור בטעות. לחיצה על כפתור זה תנתק את המשתמש מאפליקציה ותעביר אותו למסך ההתחברות.

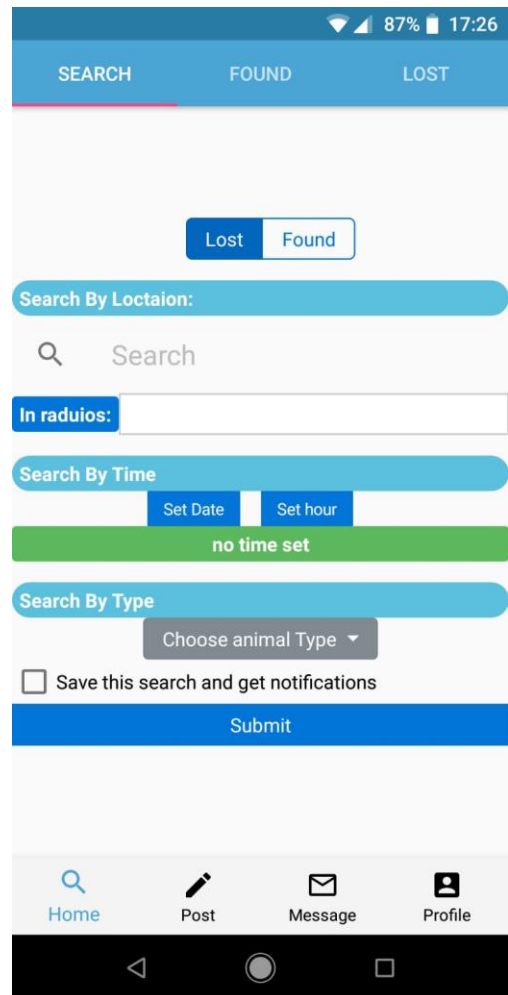
במידה והמשתמש לחץ על כפתור העריכה, הוא יועבר למסך עריכת הפרופיל.

עריכת פרופיל:



במסך זה יכול המשתמש לשנות את פרטיו (אימייל, סיסמא, וכו') וכמו כן פה ניתן להוסיף תמונת משתמש.

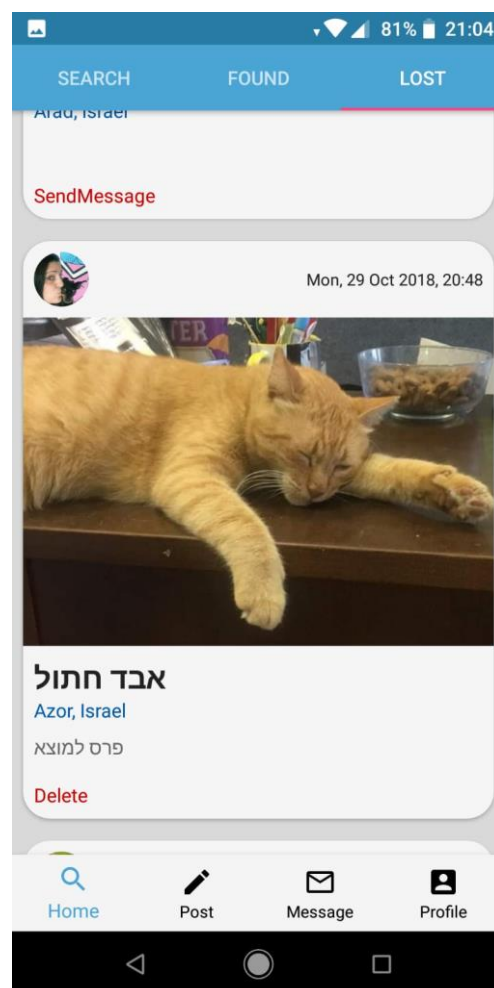
מסך החיפוש



לאחר שמתמש נרשם/נכנס לאפליקציה הוא יגיע למסך זה. זהו המסך הראשי של האפליקציה (HOME) ונכנסים אליו אם המשתמש כבר רשום ומחובר. בראש המסך שלוש לשוניות (טאבים) וניתן לעבור ישירות לדפדוף במאגר המידע על ידי הקשה על LOST או FOUND והלשונית הראשית היא לשונית החיפוש (SEARCH). על המשתמש לבחור לחפש בFOUND/LOST ויתר הפרמטרים של החיפוש הם אופציונליים, וכאשר מוכנס פרמטר אז הוא מחייב בחיפוש: תאריך – תופענה מודעות שפורסמו החל מתאריך זה ואילך, כנ"ל שעה. מקום חיפוש ורדיוס – תופענה מודעות התואמות למקום הנבחר ברדיוס הנבחר (ק"מ). בסביבת האפליקציה השתמשנו ביכולת המובנית של CLASS LOCATION של JAVA ב-ANDROID, המרנו את המידע מהשרת (שמאוחסן כ LONGTITUDE/LATITUDE טקסטואלית) ובצענו את החישוב באמצעות הפונקציה המובנית במחלקה.

על הצ'קבוקס Save this search and get notifications נפרט בסעיף **סוכן חכם**.





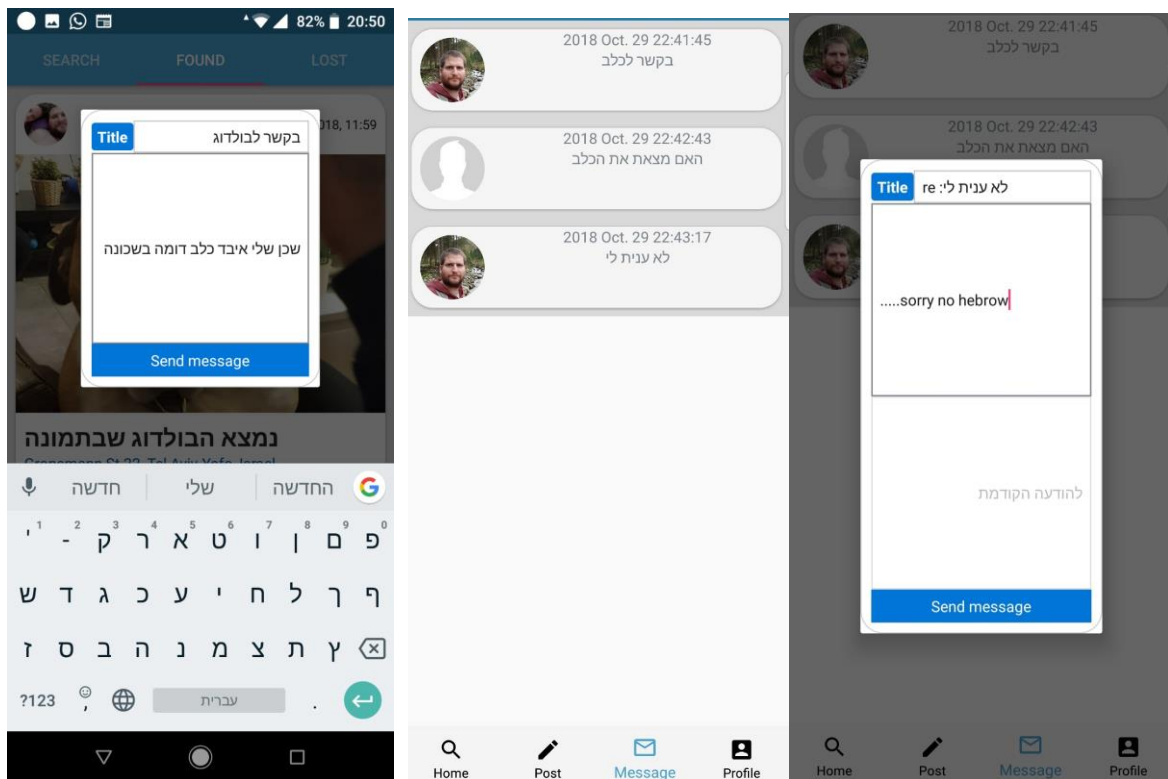
אפשר להגיע למסך דפדוף בתוצאות על ידי חיפוש עם פרמטרים או על ידי הקשה על הלשונית בראש המסך.

כל מודעה ברשימה מכילה את תאריך היצירה שלה, שם המשתמש של מי שיצר אותה ותמונה שלו (אם העלה), תמונה של בעל החיים, כותרת תמציתית, פרטים על בעל החיים, ומיקום – ברירת המחדל היא מיקום נוכחי שהאפליקציה יודעת למצוא לבד, אך ניתן לבחור להכניס מיקום ידנית, ואז יופיע שם המיקום.

במידה והמשתמש שנמצא באפליקציה מסתכל על מודעה שהוא העלה, הוא רואה כפתור מחיקה של המודעה.

ליתר המודעות מופיעה אפשרות לשליחת הודעה. שמופיעה בעמוד הבא:

## שליחת הודעות



מסך ראשון משמאל:

שליחת הודעה ליוצר המודעה:  
נוסף דיאלוג על גבי המסך לשליחת הודעה, הרקע מעומעם.  
המשתמש יכול להזין כותרת ואת תוכן ההודעה ולשלוח.

מסך אמצעי:  
מסך הודעות

מסך ימני:  
שליחת הודעת תגובה להודעה. טקסט ההודעות הקודמות יופיע בחלק התחתון של הדיאלוג.

כל משתמש המקבל ההודעה יקבל התראה אוטומטית מהמערכת (PushNotification).

מסך זה אחראי ליצירת מודעות חדשות,

בראש המסך ניתן להוסיף תמונה (יש אפשרות מתוך הגלריה או לצלם באמצעות המצלמה).

לאחר מכן יש לבחור מיקום, כאשר ברירת המחדל היא מיקום נוכחי, אך ניתן להזין ידנית.

שדה סוג בעל החיים נועד כדי להפוך את החיפוש ליותר יעיל, כלומר אם אבד לי תוכי, אין סיבה שאעבור על עשרות/מאות מודעות של כלבים.

ניתן להוסיף פרטים בטקסט חופשי.

בהתאם לבחירה LOST/FOUND מופיעים הפרטים הרלוונטיים (שם החיה לאובדן ומסוכנות, צ'קבוקס "יש לי אותו" לחיה שנמצאה)

לאחר שהמשתמש לוחץ על כפתור השלח, המודעה תתווסף אוטומטית לרשימת הרלוונטית, ותוכל להימצא בחיפושים. כאמור הסוכן החכם מופעל בכל עדכון של DB.

## סוכן חכם

כאשר ניסנו לחשוב על שימושים באפליקציה, הבנו כי יש משמעות רבה לעדכונים חיים על הקורה בה. לא מספיק שאחפש בהודעות, אלא במידה ואיבדתי חיה ישנה חשיבות מכרעת בעבורי לדעת על חיות שנמצאו באזורי.

כמובן, שכאזרח טוב, אוכל להגדיר גם חיפושים באזור בהם אני מתגורר על מנת לפקוח עין. ניסנו לחשוב כיצד הדרך הנכונה ביותר לענות על הגדרות אלו והחלטנו כי האפליקציה צריכה לדחוף את הנתונים במצבים מסוימים.

ההחלטה הראשונה שלנו הייתה לייצר "סוכן חיפוש חכם".

**הרעיון:** מעתה לא רק מבצעים חיפוש בכל כניסה למערכת אלא מאפשרים שמירה של הנתונים אל תוך הDB, ובכל פעם שבו תפורסם הודעה שעונה על הגדרות החיפוש השלי, תוקפץ באופן אוטומטי הודעה למשתמשים אשר הגדירו סוכן חיפוש חכם.

**טכנולוגיות לביצוע:** בשל כך התחלנו להשתמש בצד שרת דרך firebase cloud functions מבוסס nodeJS.

אופן הביצוע:

- 1) הוספה לבסיס הנתונים של ה firebase מבנה נתונים חדש: המבנה התבסס על החיפוש הקיים במערכת (כפי שמוסבר בפרק החיפוש) בהוספה של זיהוי השתמש הייחודי.
- 2) בעזרת firebase functions נכתב קוד JS אשר מורץ בסביבת node.js ע"י שרת הDB
- 3) הקוד מופעל בכל פעם שמפורסמת הודעה חדשה. לאחר ההכנסה - כלומר דאגנו לכך שמבחינת המשתמש זמן הביצוע של הפעולה לא יעלה, הפעולה תתבצע לגמרי ברקע. אך תהיה מיידית מספיק כך שתוכל לעזור באמת גם למשתמש המקבל.
- 4) הקוד פועל בדרך הבאה:
  - a. לאחר השמירה מתקבל contex השמירה לפונקציה הטריגר המוגדרת.
  - b. הפונקציה שולפת את כל סוכני החיפוש הקיימים במערכת.
  - c. הפונקציה בודקת עבור כל אחד מן הסוכנים האם ההודעה החדשה שפורסמה עונה להגדרות החיפוש של הסוכן.
  - d. במידה וכן, לוקחת את פרטי המשתמש ושולפת מהם את הTOKEN לשליחת נוטקפציה שנשמר בפרטי המשתמש.
  - e. הפונקציה שולחת נוטיפקציה מסוג "NewPost"
  - f. פתיחת הנוטיפיקציה מפנה את המשתמש למסך המתאים.

## התראה על הודעות

שוב מתוך מחשבה על המשתמש, אשר איבד את כלבו (או המשתמש אשר מצא הכלב ומחזיק בו), הבנו כי גם הם ככל הנראה ירצו לקבל התראה מיידית כאשר נשלחת להם הודעה מתוך המערכת.

בדומה לסוכן החכם, נכתבה הפונקציונליות ב firebase cloud functions מבוסס nodeJS.

הפעם שליחת התרעה למשתמש המתאים פשוטה יותר, בעזרת זיהוי המשתמש אשר אליו נשלחת ההודעה, המערכת מחפשת את המשתמש בבסיס הנתונים ושולחת אליו התרעה כי התקבלה הודעה חדשה. בעת פתיחת המערכת דרך הנוטיפיקציה המשתמש מועבר באופן מיידי למסך ההודעות.

## טכנולוגיות המערכת

את האפליקציה פיתחנו בסביבת אנדרואיד סטודיו בשפת ג'אווה.

בהתאם לדרישות הטכנולוגיה לכל מסך באפליקציה נלווה קובץ שבו יישמנו את עיצוב הדף, וקובץ שמכיל את הלוגיקה.

### [Google maps auto complete API](#)

בתחילה נכתבה המערכת במערכת אשר מחזיקה Strings בלבד. כאשר התחלנו לעסוק במודלים שעוסקים בחיפוש הבנו כי לא ניתן לבצע חיפוש המבוסס על String בלבד – משום שזהו חיפוש חסר משמעות, במיוחד כאשר מדובר באזור (רדיוס סביב נקודה), ולא בנקודת עניין עצמה.

למשל: כלב שהלך לאיבוד ביפו יכול למצוא את עצמו בקלות בחולון, ובשל כך חיפוש טקסטים כמעט חסר כל משמעות. הבנו כי אנו רוצים להרחיב את אופציות השמירה, ולמצוא טכנולוגיה אשר תומכת במיקומים השונים.

תוך חשיבה לעתיד על הרחבת פונקציונליות שתאפשר גם תצוגה על גבי מפה.

בחנו טכנולוגיות שונות, שמציעות עבודה על נתונים. החשיבה לבסוף הכריעה את הכף לטובת שימוש ב google maps .

ההכרעה באה מתוך מספר שיקולים :

1. ממשק שמציע אפשרויות הרחבה רבות.
2. מוכר למשתמשים בצורה הרחבה ביותר.
3. ממשקים קלים לעבודה מול סביבת android ( אחרי הכל, אנחנו תחת אותו יוצר )
4. המיפוי הוא כלל עולמי מלא.
5. תמיכה בלוקליזציה (כלומר: אפשרות להקיש בכל שפה שארצה)

ממשק זה הוסף לשני מסכים במערכת :

1. לטופס **מצאתי/איבדתי חיה** -
  - א. בטופס קיימת אפשרות על דיווח ממקום אחר השונה מהמיקום הנוכחי.
  - ב. בנוסף במצבים בהם לא ניתנת הרשאה לשימוש בפרטי המכשיר – מצב זה הוא הכרחי על מנת לקבל מיקום מדויק, שיכול לשמש פעולות נוספות במערכת.
2. לטופס **חיפוש** – זהו פרמטר חיפוש אפשרי אשר מצטרף לנתון הרדיוס. כאשר המיקום שנבחר משמש מרכז אזור שטח החיפוש, יחד עם רדיוס חיפוש (ק"מ).

לבסיס הנתונים איננו שולחים את הנתון המלא של המיקום, מאחר והוא ארוך ומלא בנתונים שרובם אינם רלוונטיים.

בחרנו לשמור שני ערכים בלבד :

1. שם המיקום – המערכת נבנתה בשלב הראשוני כך שתתמוך בנתון טקסטואלי ובמטרה למנוע בעיה עם תאימות לאחר, החלטנו לשמור על נתון זה – נתון זה נשמר עבור טופס ההזנה בלבד ולא עבור נתוני החיפוש.

2. הערך השני – נשמור כ-CLASS נוסף אשר הקמנו במערכת. שם ה-class הינו LatLan והוא מכיל את שני הערכים (Latitude/Longitude ברשת GPS העולמית).  
הסיבה להקמת ה-CLASS הייתה חוסר זיהוי של ה-CLASS החלקי בחזרה מתוך בסיס הנתונים ב-FIREBASE.

### [BootStrap android](#)

Bootstrap הינה טכנולוגיה שהחלה בבית הפיתוח של twitter ובהמשך שוחררה להיות פרויקט קוד פתוח. Bootstrap מספקת חבילה שלמה אשר מאפשרת ייצור של פקדים שונים מעוצבים היטב, אשר נתמכים בלוגיקה מתאימה. ובעצם מאפשרת למפתח לקבל בקלות יחסית, פקד מעוצב ושלים.  
הכרנו את ה bootstrap ממסגרות שונות, בהם עבדנו עם טכנולוגיה זאת בעיקר בסביבת WEB. כאשר גילנו כי קיימת תמיכה גם לסביבת ה android החלטנו לתת לה ניסיון גם פה.  
השימוש גם פה, נעשה לאחר כתיבת האפליקציה הבסיסית, והוחלט ליישם את השימוש בה לטפסים העיקריים של המערכת: טופס החיפוש, וטופס הפרסום .  
לצערנו תוך שימוש בטכנולוגיה, גילנו כי בניגוד למה שאנו מורגלים מסביבת ה web הטכנולוגיה עדיין לא מותאמת מספיק לסביבת האנדרואיד. קבלת מידע כגון טקסט מ-Drop\_Dwon – הפך להיות משימה מורכבת בהרבה.

### [NODE.JS – מבוסס java script -לישימוש ב firebase cloud function.](#)

לצורך מימוש הסוכנים החכמים נדרשנו לפתח טריגר על ה-DB. השיטה לעבודה זאת התבצעה דרך node\_js , בהתאם לדרישות ה fireBase . מאחר ועבדנו על בסיס נתונים קטנים החלטנו לא להשתמש ב Type script אלא עבודה ישירה ב js .

### [FCM -firebase colud messaing](#)

מאחר ורצינו לשלוח את ההודעות בעת הכנסה ל db , השתמשנו בטכנולוגיה זאת שמופעלת ישירות ממערכת ההפעלה ולא דרך האפליקציה.

## תיאור בסיס הנתונים

### סביבת עבודה

את ניהול המשתמשים ואחסון המידע אנו עושים בעזרת firebase-real time

הבחירה בבסיס הנתונים, הזה נבעה ממספר סיבות:

1. קלות השימוש בבסיס הנתונים מול סביבת הפיתוח שלנו
2. מאחר וכמעט כל המודלים אינם "עומדים" בפני עצמם ולא קיימים קשרים מסובכים אין לנו סיבה להחזקה של DB רלציוני.
3. בסביבת ה firebase נמצאו שתי סביבות נתונים firebase real time וה cloudStore אשר נמצאת בגרסת הבטא שלה. בחרנו בשימוש ב real time מאחר ורצינו סביבה יציבה אשר ניתן להסתמך עליה, ולא תספוג שינויים בעתיד.

### תיאור סכימת בסיס הנתונים

בבסיס הנתונים שלנו מורכב מ-4 מודלים מרכזים (4 מודלי המערכת)

#### 1. משתמשים מכיל את המידע הבא:

- a. פרטי המשתמש (כולל תמונה ושם משתמש, להצגה בכל המודלי המערכת)
- b. notificatioToken - נשמר במערכת בעת קבלה ראשונה של notificiaion מידע זה נשמר על מנת לאפשר לסוכן החכם, ולאגוריתם אפשרות לשלוח מידע למשתמש מסוים.

gTzyERInGEbAzW6sRJ2ed0Z4j0v2

```
email: "elhanan789@gmail.com"
id: "gTzyERInGEbAzW6sRJ2ed0Z4j0v2"
notificationTokens
  cZovj_E09Xs:APA91bFolt5_Jq-r_qvJrdC4DVULCgZ2jgbv3Sz2QrVTsma_Aqrpc0fB4oyRgMsDUSW6MecQvGSUiuyca9gRPdX1
photoUrl: "https://firebasestorage.googleapis.com/v0/b/fin..."
username: "Elchanan"
```

#### 2. רשימת מודעות, זהו ה DB המרכזי, כאשר עבור כל העלאת מודעה אנו שומרים את כל פרטיה, ואת המשתמש שיצר אותה.

סכמה זאת מחולקת לשתי סכמות נוספות:

1. סכמה 1 עבור חיות שנמצאו
2. סכמה 2 עבור חיות שאבדו

הסיבה העיקרית לשמירה זאת, כי בכל הגעה למערכת, ובכל חיפוש המודלים נפרדים לחלוטין. אין יישומות בהצגה משולבת של הודעות מצאתי ואיבדתי מצד המשתמש, ולכן החלוקה מאפשרת זמני שליפה טובים ביותר ומאפשרת עבודה טובה יותר של המשתמש מול בסיס הנתונים.

המודעות השונות מכילות את כל הפרטים שהזין המשתמש בעת דיווח על מודעת איבוד/מצאת חיה. בכל מודעה מופיעה פרטי המשתמש אשר יצר את המודעה, בכדי לאפשר מחיקה של הודעה/שליחת הודעה ליוצר ההודעה.

יש לשים לב, שלמען ביצוע החיפוש, נשמרים בבסיס הנתונים לא רק מחרוזות, אלא גם משתנים מורכבים יותר.

מקום לדוגמא, נשמר כאובייקט אשר מכיל ציון קואורדינטות אורך ורוחב (ברשת GPS העולמית).

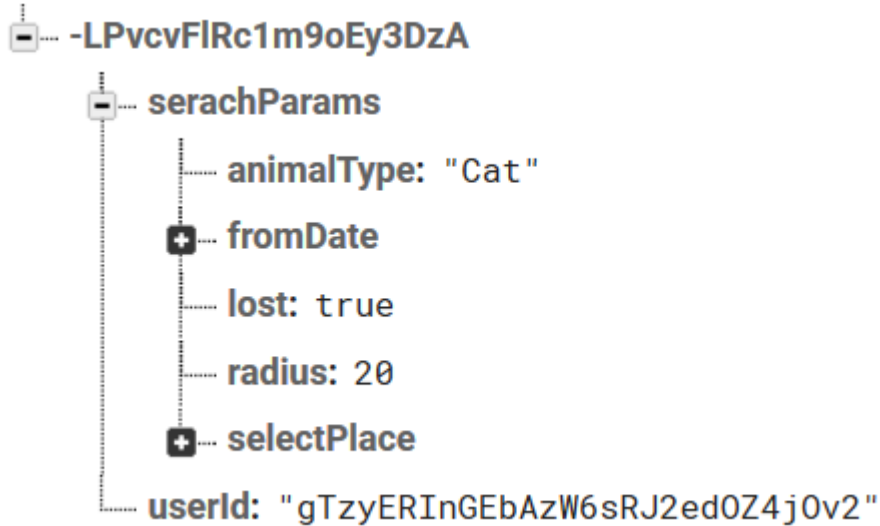
זמן – נשמר כמחרוזת , אך נשמר גם כ unixTime , בכדי לאפשר סינון של נתון זה .

```
-LPITvZ5Sbo4be_wfbMS
  animalName: "צ'צ'כ"
  animalSize: "Medium"
  animalType: "Dog"
  chipNumber: ""
  date: 1540575966847
  description: ""
  foundOrLost: "Lost"
  iHavelt: false
  isDangerous: false
  itemId: "-LPITvZ5Sbo4be_wfbMS"
  loc
    location: "מיכלאנג'לו, תל אביב יפו, ישראל"
    timeAndDate: "יום ו', 26 אוק', 2018, 20:46"
    title: "ר"
    userId: "gTzyERInGEbAzW6sRJ2ed0Z4j0v2"
```



### 3. פרטי החיפושים לשימוש בסוכן החכם –

1. פרטי החיפוש – אילו הפרמטרים אשר יועלו אוטומטית בעת טעינת מסך הבית במידה והוחלט לשמור את הנתונים, ובנוסף ישמשו אותנו לסינון הודעה בסוכן החכם.
2. פרטי המשתמש - אשר אליו משויך החיפוש.




#### 4. הודעות בין משתמשים

עבור כל משתמש אשר מקבל הודעה, הוגדרה תיקייה (נוצרת באופן אוטומטי בשליחה ראשונה של הודעה למשתמש).

כל הודעת המשתמש מופיעות תחת תייקיה זאת. הבחירה במבנה זה נבעה גם כן משיקולי שימוש במערכת :

1. בהצגת הודעה ירצה המשתמש לראות את ההודעות שנשלחו אליו – אחסון באופן זה מאפשר הבאה קלה יחסית של הנתון.
2. בשליחה התראה על קבלת הודעה חדשה, ההודעה עצמה פחות קריטית מהמשתמש, גישה שכזאת למשתמש מאפשרת שליחת הודעות בצורה יעילה יותר .  
עבור כל הודעה נשלחים פרטי המשתמשים (המחבר והנמען) .  
ונשלחים פרטי המודעה בה הם עוסקים.  
הטקסטים הקודמים של ההודעה (במצב של הודעה שהתקבלה כמענה)  
טקסט ההודעה וכוורת.

KHL0RlRw8iUrv2gexdZWH90UfFX2

 -LPw1XEY1nCqj3tn1cNr

 -LPwA30Y3P3fxSOjdsuD  

```
fromUser: "gTzyERInGEbAzW6sRJ2ed0Z4j0v2"
```

itemKey: "-L0hY4nXC\_gQdAw1q9WQ"

```
itemType: "lostAnimals"
```

messageId: "-LPwA30Y3P3fxS0jdsuD"

**prevText:** "זוהי ההודעה המקורית שנשלחה"

... sendDate

**text:** "זו הודעה התשובה אליה"

**title:** "הצגת הודעה: re"

```
toUser: "KHL0R1Rw8iUrv2gexdZWH90UfFX2"
```

## Machine learning - ולמה החלטנו שלא להוסיף מודול שתומך בכך

אחד המרכיבים העיקריים שחשבנו לשלב בפרויקט היה נושא של זיהוי תמונה. נושא זה היה אמור לשדרג את הפרויקט בצורה משמעותית – לשאיפתנו היה מאפשר הפעלה טובה יותר של הסוכן החכם, ומאפשר למשתמש הודעה כמעט מידית בנוגע לזיהוי של חיה מסוימת.

המחשבה הייתה לתת למשתמש להזין תמונה ובעזרת אלגוריתמים קיימים של זיהוי תמונה, למצוא תמונות של בעלי חיים דומים.

אך מחשבה לחוד ומציאות לחוד, התחלנו לבחון את שני הכלים המרכזיים המשמשים כיום מפתחי אנדרואיד לביצוע פעולות אלו:

- [GOOGLE Cloud Vision API](#)
- [FIREBASE Cloud-based image labeling](#)

שניהם מבית גוגל.

מאחר וכל האפליציה מתבססת על firebase - חשבנו כי אך מתבקש בשביל להשלים את החבילה כי ניגש לחבילה זאת. עברנו על החבילה ומצאנו מספר פונקציות שנראו כי יכולות להתאים לאפליקציה.

מצאנו את הפונקציה הראשונה בה שקלנו להשתמש – השוואת התמונות, מבוססת על מנגנון השוואת התמונות של גוגל. המנגנון עובד היטב על זיהוי פרצופים ומסוגל להבדיל היטב בין בני אדם. אך מסתבר כי האלגוריתמים הקיימים אינם מספקים תמיכה מספקת עבור חיות.

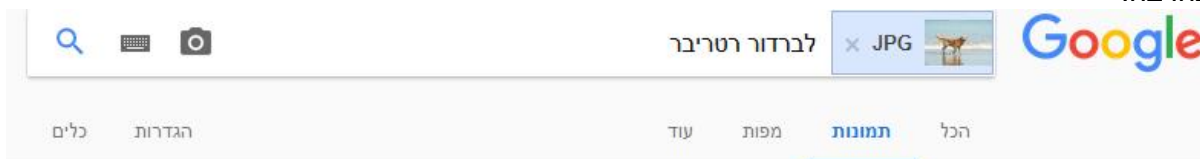
האלגוריתם מנתח את התמונה בכללותה, ולא מתמקד בחיה בלבד. כלומר כאשר ניקח שתי תמונות של אותו הכלב, מצולם מאותה הזווית, תמונה אחת תכיל את הכלב שוכב על הדשא ואילו התמונה השנייה תכיל את אותו הכלב על חוף הים. מנגנון זיהוי התמונה של גוגל יחפש כלבים דומים בחוף, ויזהה אותו כדומים יותר מאשר אותו הכלב אשר שוכב על הדשא (צילום מסך של חיפוש בגוגל – במסך הבא).

התמונה אשר מוצגת ראשונה הינה הכלב אותו אנחנו מחפשים. (מבחינת שימוש במערכת, נניח כי משתמש ראה משוטט לבדו בים). תוצאות החיפוש שקיבלנו, מראים לנו אומנם כלבים מאותו גזע. אבל במבט מהיר בתמונה ניתן לראות כי חלקם גורים, צורות פרווה שונות, ואפילו הצבעים בתמונה האחרונה הינם תואמים כלל לחיפוש אותו ביצענו.

הגענו למסקנה אחרת מן הנושא- כפי שניתן לראות בשורת החיפוש, האלגוריתם ידע לזהות לבד את סוג החיה (כלומר זיהוי גזע). ניתן להשתמש באלגוריתמים הקיימים בשביל לקבל פרטי מידע על החיה, כלומר אומנם לא נבצע התאמת תמונה מלאה אבל נוכל בקלות לקבל מגוגל מידעים (בצורת תגיות) – הרי גוגל ידע לזהות כי מדובר בכלב מסוג גולדן רטיבר. נוכל להזין את טקסט זה בשורת התיאור שלנו. פשוט נבקש מבסיסי הנתונים לתאר לנו את התמונה.

התחלנו לחקור את הנושא לעומק, בעזרת השימוש ב test של האפליקציה, ראינו מהר מאוד כי מתבצע תיאור מלוא התמונה, עבור תמונת הכלב בחוף הים, לא קיבלנו מידעים רלוונטיים על הכלב (צבע לדוגמה), קיבלנו זיהוי שהצליח לטעות מספר פעמים בנוגע לסוג החיה ותיאור של התמונה. כלומר, אם ניקח צבע מתמונה של כלב על דשא, נקבל ירוק ולא את צבע הכלב. רקע התמונה איננו רלוונטי לנו כלל. הסקת מיקום מן התמונה מיותר, בשל העובדה שאנו משתמשים במכשיר בשביל להציג את נתוני המיקום וזהו נתון מדויק

בהרבה.



כ-25,270,000 תוצאות (0.68 שניות)

גודל תמונה:  
630 × 434

חפש גדלים נוספים של תמונה זו:  
כל הגדלים - קטן



הניחוש הטוב ביותר לתמונה זו: לבדור טריבר

CE – ויקיפדיה

▼ <https://he.wikipedia.org/wiki/CE>

CE (אות קטנה: ce) היא ליגטורה של האותיות e ו-c, אשר שהפכה לאות ושימשה בלטינית, צרפתית ונורדית עתיקה, וכיום נמצאת בשימוש באנגלית עבור מילים שאולות מלטינית ויוונית.

[How to pronounce "CE" sound in French \(Learn French With ...](#)

▼ תרגם דף זה <https://www.youtube.com/watch?v=wtbFi2TFwWs>

SUBSCRIBE HERE ► <http://learnfren.ch/YouTubeLFWA> for Learn French With Alexa's - 2015 בפבר' 18  
FREE French lessons. Alexa Polidoro, from ...

תמונות דומות



דיווח על תמונות

בשלב זה החלטנו לנסות את google\_api\_vision, אומנם firebase מבוסס על אלגוריתמים דומים אך טכנולוגיה זאת הייתה אמורה לספק לנו מיידעים רלוונטיים יותר. השתמשנו במערכת ההדגמה שלהם בשביל להבין מה התוצאות אשר הם מאפשרים לנו להציג. התוצאות היו מפתיעות, בניגוד ל firebase הוצג לנו מגוון רחב של תגיות שונות, בהעלאת תמונה של כלב לדוגמא, התאפשר לנו לראות את האחוזים השונים כי מדובר בתוצאה מדויקת. העלנו מספר תמונות, להבנת טיב האפליקציה, ומהר מאד גילנו כי עבור תמונות של כלבים קיבלנו תוצאה כי מדובר בחתול – בהסתברות גבוהה יותר מאשר הסיכוי כי מדובר בכלב.


קראנו על הנושא והבנו כי זיהוי של רכיב בתמונה ( לדוגמא זיהוי פנים) – מבוסס על אלגוריתמים מיוחדים שפותחו לצורך כך.

זיהוי הפנים מתבסס בעצם על שני אלגוריתמים:

1. זיהוי הפנים מתוך התמונה
2. לבדיקת תאימות משתמשים במספר נקודות שנבחרו בדיקת סימטרית ומבנים שונים.

סט האלגוריתמים הזה פותח במיוחד לצורך כך. לצערנו לא מצאנו ממשקים ישימים לשימוש אשר תומכים בזיהוי בעלי חיים.


דוגמא לשימוש ב google vision API :

Labels	Web	Document	Properties	Safe Search																
 20160128_202723.jpg			<table><tr><td>Cat</td><td>94%</td></tr><tr><td>Small To Medium Sized Cats</td><td>90%</td></tr><tr><td>Dog Breed Group</td><td>88%</td></tr><tr><td>Dog</td><td>86%</td></tr><tr><td>Dog Like Mammal</td><td>85%</td></tr><tr><td>Floor</td><td>82%</td></tr><tr><td>Flooring</td><td>79%</td></tr><tr><td>Cat Like Mammal</td><td>75%</td></tr></table>	Cat	94%	Small To Medium Sized Cats	90%	Dog Breed Group	88%	Dog	86%	Dog Like Mammal	85%	Floor	82%	Flooring	79%	Cat Like Mammal	75%	
Cat	94%																			
Small To Medium Sized Cats	90%																			
Dog Breed Group	88%																			
Dog	86%																			
Dog Like Mammal	85%																			
Floor	82%																			
Flooring	79%																			
Cat Like Mammal	75%																			

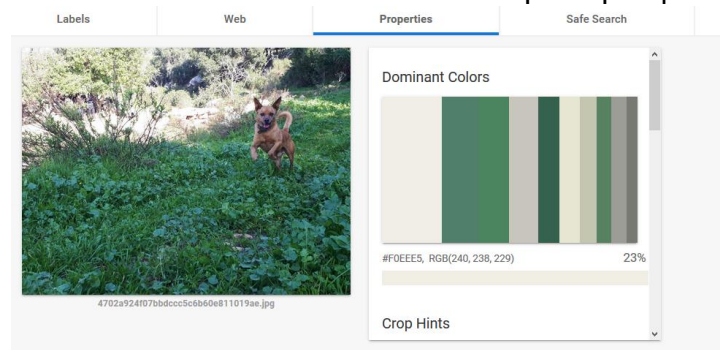
ניתן לראות כי הכלבה בתמונה זוהתה בצורה די וודאית בתור חתול. בנוסף גודל ה"חתול" שנקבע הינו בנוני עד קטן. מדובר בכלבה גדולה.

דוגמת הכלב בדשא :

1. LABEL - ניתן לראות כי אין שום מידע רלוונטי הנוגע לכלב בחיפוש הזה.

Labels	Web	Properties	Safe Search	JSON																
 4702a924f07bbdccc5c6b60e811019ae.jpg		<table><tr><td>Plant</td><td>93%</td></tr><tr><td>Grass</td><td>90%</td></tr><tr><td>Nature Reserve</td><td>84%</td></tr><tr><td>Shrubland</td><td>78%</td></tr><tr><td>Flora</td><td>78%</td></tr><tr><td>Grass Family</td><td>73%</td></tr><tr><td>Wildlife</td><td>72%</td></tr><tr><td>Pasture</td><td>69%</td></tr></table>	Plant	93%	Grass	90%	Nature Reserve	84%	Shrubland	78%	Flora	78%	Grass Family	73%	Wildlife	72%	Pasture	69%		
Plant	93%																			
Grass	90%																			
Nature Reserve	84%																			
Shrubland	78%																			
Flora	78%																			
Grass Family	73%																			
Wildlife	72%																			
Pasture	69%																			

2. **היסטוגרמת הצבעים של התמונה** – ההיסטוגרמה מכילה בעיקר את הצבע הירוק-אנו לא יכולים להסיק מתוך הנתון הזה מהו צבע הכלב – ובשל לא יכולים להסתמך גם עליה.



כמובן כי גם קיבלנו תוצאות טובות בחלק מן הבדיקות , אך המידע שיכלנו לקבל סוג חיה וגזע בלבד – עדיין דרשו אישור נוסף על נכונות מהמשתמש , מאחר ובדיקה זאת לוקחת כמספר שניות , כלומר בכל פעם המשתמש היה נאלץ לחכות את זמן זה – בשביל לאשר את התוצאות , אשר לא נתנו כמעט מידע נוסף החלטנו לוותר על המימוש בשלב זה.

מבחינת עבודה לעתיד, לאחר בניית DB יותר רחב , עם תיאור ותמונות של משתמשים הנוגעים לבעלי החיים, ניתן יהיה לפתח אלוגריתם לומד של ML בעזרת ה DB שיתקבל(בשאיפה), ולהשתמש בו.

---

## חלוקת עבודה:

### רן גרמן: כתיבת בסיס האפליקציה מבוסס מחרוזות – פרופיל , פרסום מודעה , הצגת מודעה

מסך הרשמה – לוגיקה ותצוגה  
מסך התחברות – לוגיקה ותצוגה  
טופס אבדת בעל חיים – לוגיקה ותצוגה  
מסך הבית (רשימות בע"ח) – לוגיקה ותצוגה  
מסך פרופיל – לוגיקה ותצוגה  
מסך עריכת פרופיל – לוגיקה ותצוגה  
שימוש במצלמה ובגלריית התמונות מהמכשיר  
מציאת מיקום נוכחי בהתבסס על מיקום המכשיר  
יצירת התפריט בתחתית המסך, הלשוניות בראש מסך הבית, ומעבר בין כל המסכים – לוגיקה ותצוגה.

### אלחנן סמואל: הוספת מודלי ההודעות והחיפוש , לוגיקת התראות , העשרה טכנולוגית

google\_maps הוספה לכל המסכים הרלוונטים באפליקציה – התמשקות ל - שינוי מסכי השמירה.  
הוספת bootstrap – לכל הטפסים פרט למסכי ההתחברות – שינוי מבנה המסך הביא גם לשינויים רבים בלוגיקת המסכים .  
טופס מציאת בעלי חיים – שינוי לוגיקה נדרשים- להרחבת השמירה כך שתתמוך בשמירה של הנתונים בצורה מורכבת יותר – מעבר לשמירה כמחרוזת בסיסית .  
חיפוש- יצירת מסך + לוגיקה – שמירת התוצאות  
חיפוש – הצגת תוצאות – כתיבה מחדש של הלוגיקה לרשימות בע"ח ( כך שתוכל לתמוך בביצוע חיפוש )  
הוספת עבודה עם זמנים במערכת – הוספה של ה dialog הנדרשים לצורך .  
סוכן חיפוש חכם – כתיבת הפונקציות בgoogle\_colud  
הודעות – יצירת dialog שמירת הודעה - לוגיקה ותצוגה  
מסך הצגת הודעות – לוגיקה ותצוגה  
הודעת – כתיבת הפונקציות בgoogle\_colud .  
אפשר באפליקציה של שליחת PushNotification ( גם מבחינת הגדרת משתמש , וגם מבחינת כתיבת השירותים הנדרשים באפליקציה , לוגיקת ההצגה, ולוגיקת ההפניות בעת הכניסה למערכת).  
בדיקת התכנות ( מימוש ומחיקה ) – של פונקציות ML.