



בס"ד

Ani-Map גל חלילי



שם בית הספר: תיכון ע"ש פנחס אילון

סמל בית הספר:



שם פרויקט: Ani-Map

שם המגישה: גל חלילי

תעודת זהות: 317997039

שם המנחה: אבי טויל

שם החלופה: תכנון ותכנות מערכות

תאריך הגשה: 20.03.2020



## תוכן עניינים

3	<b>מבוא</b>
4	אתגרים מרכזיים
5	<b>מבנה / ארכיטקטורה של הפרויקט</b>
5	הצגת הפתרון המוצע והסיבות לבחירתו
6	מחלקות הפרויקט
8	תרשים זרימת מסכים
9	תרשים מחלקות UML
10	Use case
11	דרישות/מגבלות להפעלת התוכנה
12	<b>מדריך למשתמש</b>
12	פירוט מסכים
12	1. מסך בית – HomePage
12	2. הרשמה – SignUp
13	3. התחברות – Login
14	4. מפה – Map
15	5. העלאת נקודה – AddSpot
17	6. מידע על נקודה – EventInfo
18	7. האירועים שלך – YourSpots
19	8. מקרא – Legend
19	9. אודות – About
20	<b>בסיס נתונים</b>
20	שיטת שמירת הנתונים
21	Database
21	רשימת העמודות בטבלה
21	תתי עץ של events
23	תתי עץ של users
24	Storage
26	SharedPreferences
27	<b>מדריך למפתח</b>
27	מבנה נתונים בהם נעשה שימוש
28	מידע על המחלקות ותדפיסי הקוד
43	<b>סיכום אישי</b>
43	צדדים חזקים/חלשים בפרויקט

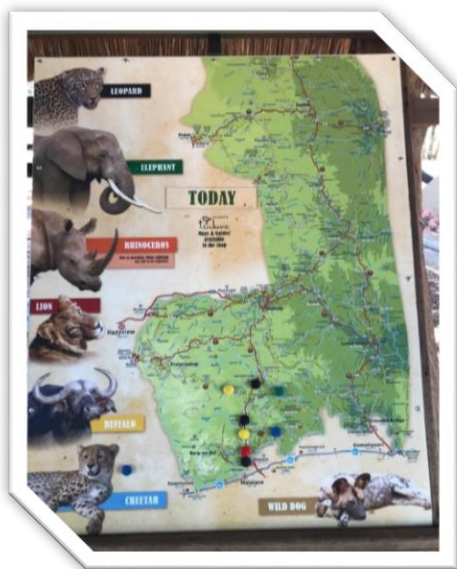


## מבוא

בתיק פרויקט זה מציג אפליקציה בשם Ani-Map, אפליקציה שאמורה לעזור לאלפי מטיילים בשמורת הטבע Pilansberg שבדרום אפריקה.

הספר יכלול הסברים על כל המחלקות והמסכים שכלולים באפליקציה, על שיטות שמירת הנתונים שבהם השתמשתי – הכל בליווי במונות מהאפליקציה ומבסיס הנתונים, וסיכום אישי שלי על חווית העבודה על הפרויקט.

האפליקציה Ani-Map עוזרת למטיילים בשמורת הטבע Pilansberg שבדרום אפריקה, יוהנסבורג. האפליקציה מאפשרת למטיילים לסמן על המפה היכן ראו חיה וכך, שאר המטיילים יוכלו להגיע למקום ולראות אותה גם כן. כל מטייל יוכל לבחור את הנקודות והאירועים שמעניינים אותו, לא לבזבז זמן מיותר בחיפוש החיות וכך לשפר את חווית הטיול שלו.



נולדתי בדרום אפריקה ואני מבקרת שם כמעט בכל חופש גדול. לפחות פעם אחת בביקור שכזה, משפחתי ואני הולכים לבקר בספארי. הרבה פעמים לא מצאנו את החיות (זאת שמורה פתוחה שאתה נוסע באוטו) ולכן היינו פותחים חלון ושואלים מבקרים אחרים אם הם ראו משהו מעניין בהמשך הדרך. הבנתי שזאת בעיה מכיוון, שהרבה אנשים יוצאים מיום טיול שכזה כשראו קרנף אחד וכמה פילים. כפתרון לבעיה זו, הספארי הציב לוח (כמו שבתמונה) בכל נקודת עצירה ובו אנשים מסמנים איפה ראו חיות. גם עם פתרון זה יש בעיות:

1. המטיילים יכולים לסמן על הלוח איפה ראו חיה רק אם הם עוצרים במקום כלומר, לא בזמן אמת.
2. מטיילים אחרים שרואים את הלוח לא יודעים מתי סומנו הנקודות על המפה ואם הן רלוונטיות.
3. על הלוח מודבקת מפה מוקטנת ולכן קשה למטיילים לסמן בדיוק איפה ראו את החיה.

לכן, חשבתי לאפשר לאנשים לסמן היכן ראו חיות בלייב דרך אפליקציה.

בתהליך בחירת הרעיון לפרויקט, נתקלתי במספר אפליקציות אשר מאפשרות למשתמשים לסמן מוקדי עניין על מפה. לדוגמה, האפליקציה Latest Sightings שנותנת למשתמשים שלה להעלות אירועים אך, המפה מתפרסת על אפריקה אמריקה ואוסטרליה כלומר היא לא על שמורה אחת. דבר שני הסימונים במפה הם על פי שעות ההעלאה ולא על פי סוג החיה.

אפליקציה זו כן פותרת את הבעיה שזיהיתי אך לדעתי היא לא הפתרון האולטימטיבי מכיוון שהמפה לא מתמקדת בשמורה אחת ולכן היא פחות נוחה למטיילים.

בנוסף, האפליקציות האחרות שראיתי בכלל לא מאפשרות למטייל לראות מידע על הנקודות שהועלו דבר שמקשה על המטיילים לדעת האם נקודה זו רלוונטית עבורם.



## אתגרים מרכזיים

הבעיה המרכזית שאיתה התמודדתי במהלך הפרויקט היא איך לבנות את האפליקציה שמצד אחד תהיה נוחה לשימוש ותענה על צרכי המטיילים, ומצד שני תעבוד בצורה טובה וחלקה. בהתחלה מאוד התלבטתי אילו אפשרויות צריך לתת למשתמש כך שהאפליקציה תיתן פתרון טוב והיעיל לבעיה שזיהיתי, איך לבנות את המסכים, מה כל מסך צריך לכלול שגם לא יהיה עמוס מידי וגם יספק את כל מה שהמשתמש צריך. הכנתי תרשים זרימת מסכים ובו כל הרעיונות והרכיבים שחשבתי עליהם ועל פי תכנון זה התחלתי לבנות את האפליקציה. כאשר עובדים על פי תרשים מסודר העבודה נהיית הרבה יותר קלה ומהנה אך מידי פעם כן הוספתי דברים שעם הזמן הבנתי שחיוניים לאפליקציה.

המוטיבציה לעבודה כמו שהסברתי מגיעה מצורך אמיתי של משפחתי ושלי. אני בטוחה שעוד מטיילים רבים נתקלו בבעיה זו ויהיו שמחים להשתמש באפליקציה שכזו.



## מבנה / ארכיטקטורה של הפרויקט

### הצגת הפתרון המוצע והסיבות לבחירתו

הפתרון שהאפליקציה Ani-Map מציעה הוא מערכת ליצירת אירועים ע"י המשתמשים שמוצגים על פני מפת שמורת הטבע Pilansberg. כל משתמש יכול ליצור אירוע שבו הוא מציין מה החיה והאירוע שראה ויכול גם לצרף תמונה מתאימה.

המשתמש גם יכול לראות אירועים שמטיילים אחרים העלו, כלומר הוא יכול לראות על מפת השמורה את כל האירועים שהועלו ב-24 שעות האחרונות, לראות על כל אירוע את המידע עליו וגם "לרענן" אירוע אם הוא נמצא במקום שצוין וראה את החיה (הרחבה בפירוט המסכים). בנוסף, למשתמש יש אפשרות למחוק אירועים שהוא עצמו העלה.

מטרת התוכנה היא לספק למשתמש תפריט נוח אשר יאפשר לו להירשם לתוכנה, להתחבר אליה, ליצור אירועים, לראות אירועים של מטיילים אחרים ואת המידע עליהם, לעדכן אותם ולמחוק אירועים שהוא העלה. אחרי הורדת התוכנה, המשתמש יצטרך להירשם דרך מסך ההרשמה ולאחר מכן להתחבר כדי להגיע למסך המפה. התוכנה נותנת למשתמש אופציות רבות בכדי שיוכל לעשות את הפעולות המוזכרות כגון: האפשרות ליצור אירועים לאחר הקלדת פרטים על האירוע (שם החיה, מה ראה באירוע ותמונה המתארת את האירוע), האפשרות למחוק ולעדכן אירועים ע"י ועוד.

חובה יכולה להיעלם מהמפה בשני מקרים. מקרה ראשון הוא שכל נקודה נמחקת לאחר 24 נרגע העלאתה או עדכונה. המקרה השני הוא שהמשתמש אשר העלה את האירוע יכול מחוק אותו במסך YourSpots, באמצעות לחיצה על השורה שבה מופיע האירוע הרצוי ב-ListView ולאחר מכן לחיצה על הכפתור "מחק".



## מחלקות הפרויקט

### מחלקות מבצעות:

מחלקה לכל מסך, היורשת מהמחלקה Activity:

- מחלקת מסך הבית HomePageActivity.

- מחלקת SignUpActivity בשביל הרשמה לאפליקציה ובה מתבצעת שמירת פרטי המשתמש בבסיס הנתונים כמו שם, סיסמה וכתובת אמייל.

- LoginActivity התחברות לאפליקציה.

- מחלקת מקרא LegendeActivity.

- מחלקת אודות AboutActivity.

- מחלקה האחראית על הוספת אירועים AddSpotActivity. מחלקה זו נותנת למשתמש את האפשרות ליצור אירוע חדש. המשתמש מזין את שם החיה ומה קרה באירוע, הוא יכול להעלות גם תמונה. אירוע זה נשמר בבסיס הנתונים (עם הפרטים הללו-מזהה האירוע, שם החיה, מיקום האירוע, תאריך ושעת העלאה, שם וכתובת אמייל של מעלה האירוע, כמה אנשים ראו או לא ראו את האירוע וקישור לתמונה מה-Storage) והתמונה נשמרת ב-Storage.

- מחלקה האחראית על הצגת מידע על אירועים EventInfoActivity שאותם היא שולפת מבסיס הנתונים בהתאם למזהה של האירוע שעליו המשתמש לחץ במפה. מזהה האירוע נמסר ב- Intent מהמחלקה MapActivity, על פי המזהה ששמור עם- ה Marker שעליו לחץ המשתמש במפה. על פי מזהה זה, האירוע הנ"ל נשלף מהשרת ופרטיו מוצגים למשתמש. מחלקה זו משתמשת בפעולה לחישוב הפרש בין זמנים, שנמצאת במחלקה NearByLocationService ומציגה כמה זמן עבר מאז העלאת האירוע.

- מחלקה האחראית על המפה והצגת האירועים עליה MapActivity. שולפת מבסיס הנתונים את כל האירועים הרלוונטיים (שלא עברו מאז העלאתם 24 שעות ושהמשתמש הנוכחי לא העלה אותם) ומציגה אותם בעזרת Markers שלכל אחד מהם מצורף מזהה האירוע שהוא מסמל. בנוסף, מופעלים בה ה- BroadcastReceiver וה- Service (אליו היא שולחת את מיקום המשתמש).

- מחלקה האחראית על הצגת האירועים שהמשתמש העלה ומחיקתם YourSpotsActivity. שולפת מבסיס הנתונים את כל האירועים שהמשתמש הנוכחי העלה ומציגה אותם ב-ListView. אם המשתמש לוחץ על אחד האירועים יוצג Popup שיאפשר את מחיקת האירוע (אם המשתמש החליט למחוק אירוע הוא ימחק אוטומטית מבסיס הנתונים). תחילה נשלפים כל האירועים מהשרת לתוך ArrayList. עבור כל אירוע ב- ArrayList נעשית בדיקה האם כתובת האמייל ששמורה בו זהה לכתובת האמייל של המשתמש הנוכחי ששמורה ב- SharedPreferences.



## מחלקה מתאמת (Adapter):

EventListAdapter מתאם האירועים.

## מחלקות אחרות:

מחלקת משתמש User מגדירה את תכונות המשתמש.

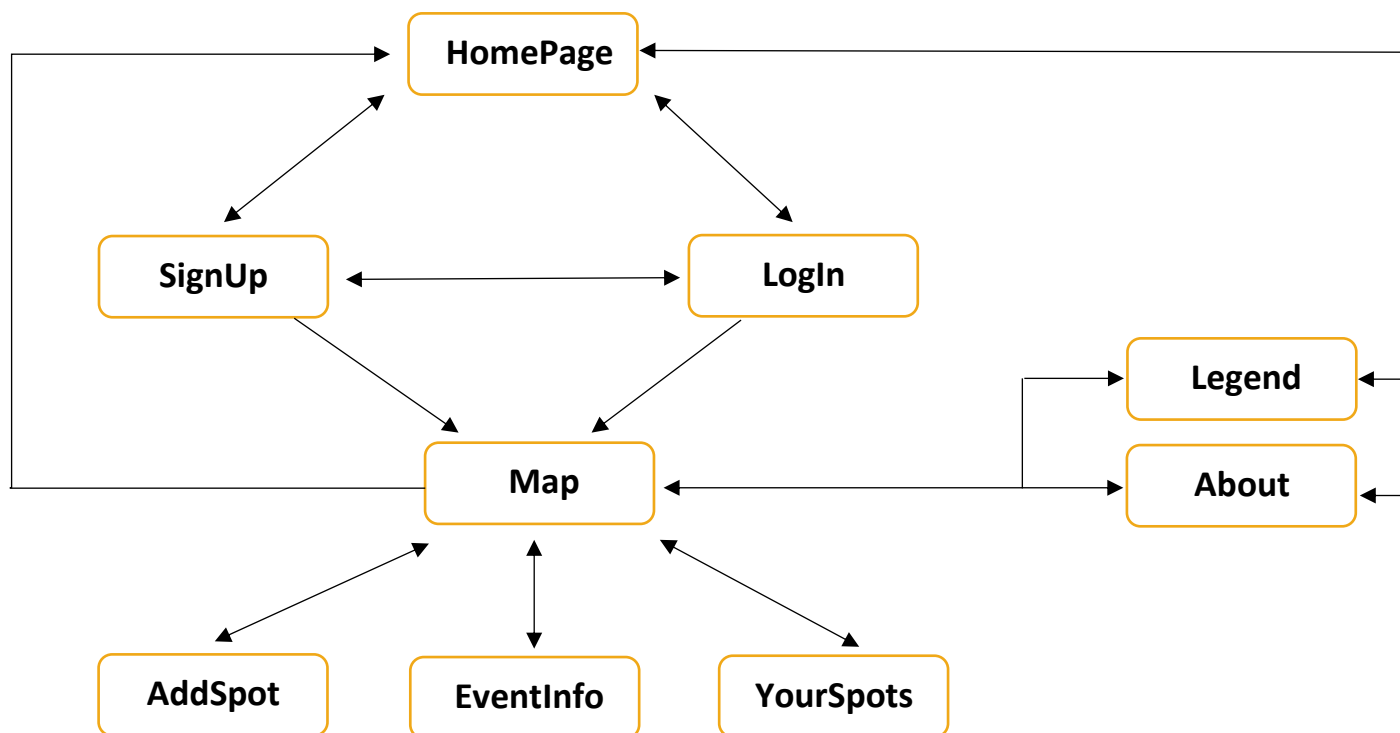
מחלקת אירוע Event מגדירה את תכונות האירוע.

מחלקת שירות NearbyLocationService היורשת ממחלקת Service. בה מתבצעת בדיקה האם יש אירועים קרובים למיקום המשתמש ושולחת עליהם notification. תחילה נשלפים כל האירועים מהשרת לתוך ArrayList. עבור כל אירוע ב-ArrayList נעשית בדיקה האם הוא הועלה לפני פחות מ-24 שעות והאם הוא קרוב למשתמש (המרחק בין מיקום המשתמש לבין מיקום האירוע מחושב בעזרת פעולה שמקבלת את שני המיקומים ומחזירה את המרחק האווירי ביניהם במטרים). אם אכן האירוע עונה על קריטריונים אלה הוא נשמר ב-ArrayList אחר. לבסוף עוברים על כל האירועים שב-ArrayList השני ונעשית בדיקה איזה אירוע הועלה לפני הכי פחות זמן ועליו תוצג ההתראה. בנוסף, במחלקה זו יש פעולה לחישוב הפרש בין שני זמנים. הפעולה מקבלת זמן (HH:mm:ss) ותאריך (dd/MM/yyyy) ומחזירה את ההפרש בין הזמן שקיבלה לבין הזמן הנוכחי בדקות.

מחלקת שידורים LowBatteryBroadcastReceiver היורשת ממחלקת BroadcastReceiver. בודקת האם למשתמש יש סוללה נמוכה. אם אכן יש לו סוללה נמוכה ישלח Popup שישאל האם ירצה לעבור למצב חסכוני בסוללה (אם המשתמש בוחר לעבור למצב חסכוני, קצב העדכון של ה-GPS יקטן).



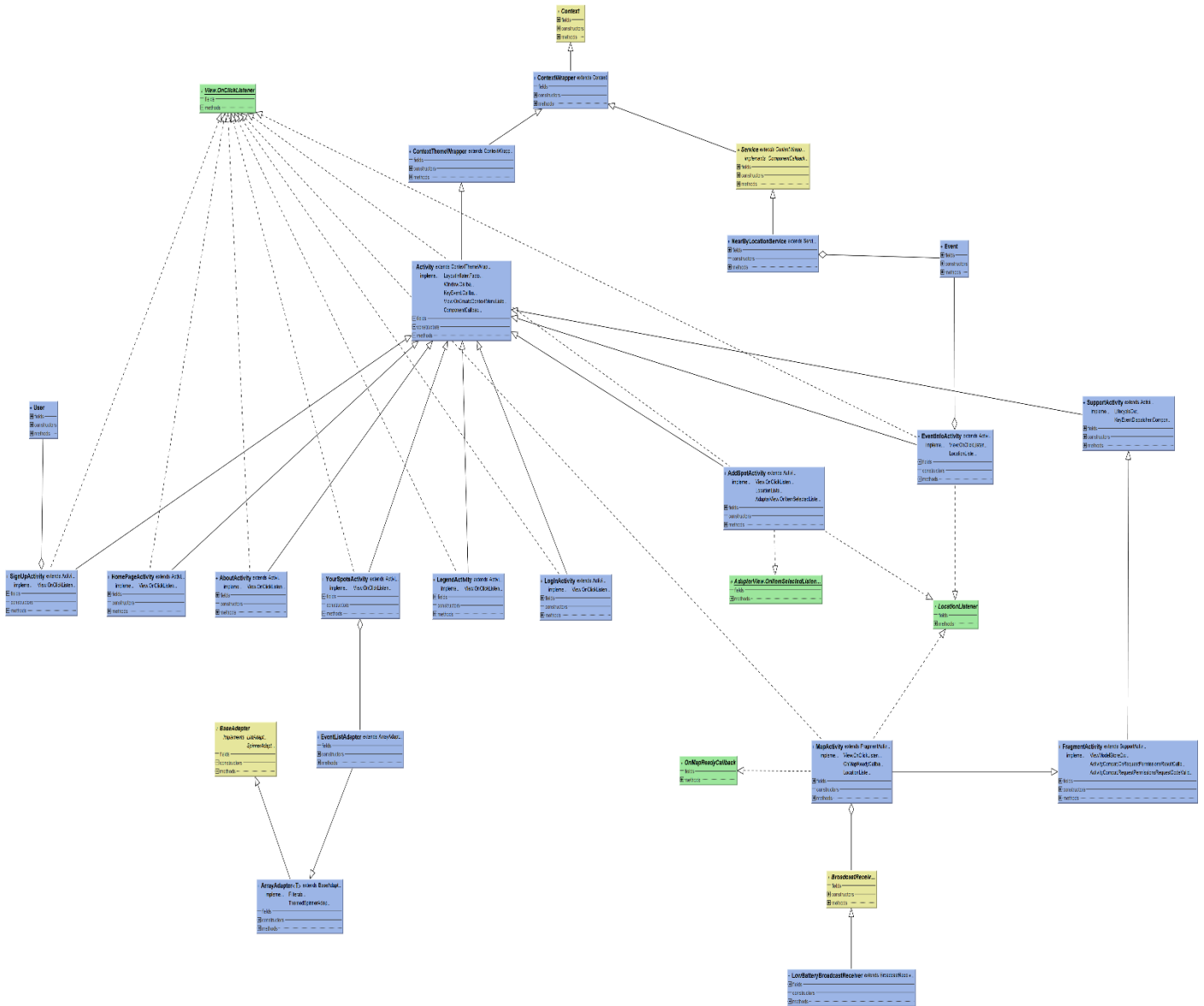
## תרשים זרימת מסכים







# תרשים מחלקות UML





## Use case

התהליך/השימוש המרכזי באפליקציה הוא העלאת נקודות/אירועים למפה.

תחילה המשתמש נרשם לאפליקציה ומזין את שמו, סיסמה וכתובת האמייל שלו. משתמש חדש זה והפרטים שלו, נשמרים ב-Firebase database תחת תת העץ users. מעמוד ההרשמה המשתמש עובר לעמוד המפה. בעמוד זה המשתמש החדש יכול לראות אירועים שמשתמשים אחרים העלו או לעבור ממנו לעמוד העלאת אירוע.

בשביל להעלות אירוע חדש, על המשתמש לעבור לעמוד הוספת נקודה ולהזין את שם החיה ומה קרה באירוע. המשתמש יכול גם לצרף גם תמונה שתשמר ב-Storage. מיקום האירוע נקבע על פי מיקומו הנוכחי של המשתמש שמועבר מעמוד המפה ב-Intent. כאשר המשתמש מעלה את האירוע, פרטי האירוע (מזהה האירוע, שם החיה, מיקום האירוע, תאריך ושעת העלאתה, שם וכתובת אמייל של מעלה האירוע, כמה אנשים ראו או לא ראו את האירוע וקישור לתמונה מה-Storage) נשמרים בבסיס הנתונים תחת תת העץ events כאירוע חדש.

המשתמש חוזר לעמוד המפה שם הוא לא יראה את האירוע שהעלה, אלא רק אירועים של משתמשים אחרים. כל האירועים ששמורים בבסיס הנתונים נשלפים ממנו ואלה שלא עברו 24 שעות מהעלאתם מוצגים על המפה. כאשר המשתמש ילחץ על אחד האירועים במפה, הוא יועבר לעמוד מידע על אותה הנקודה שמקבל ב-Intent את מזהה האירוע. בעמוד זה נעשית שליפה מבסיס הנתונים של האירוע בעזרת המידע מה-Intent, ופרטיו מוצגים למשתמש (רק שם החיה, מה קרה באירוע, מספר האנשים שראו את האירוע והתמונה אם יש). בנוסף, מוצג לפני כמה זמן הועלה האירוע. מידע זה מחושב בפונקציה שנמצאת במחלקה NearByLocationService שמקבלת תאריך וזמן ומחשבת את ההפרש בין זמן זה לבין הזמן הנוכחי.

יתרה מזאת, המשתמש יכול לראות את כל האירועים שהוא העלה בעמוד הנקודות שלך. בעמוד זה יש ListView שאליו נשלפים מבסיס הנתונים רק האירועים שהאמייל של מעלה האירוע זהה לאמייל ששמור ב-SharedPreferences, כלומר האמייל של המשתמש הנוכחי. בכל שורה מוצג אירוע אחר עם שם החיה, תאריך ושעת העלאתה ותמונה של ה-Marker שמסמן את אותה חיה במפה. משתמש יכול למחוק את הנקודות שהעלה על ידי לחיצה על השורה של אותו אירוע ב-ListView ואירוע זה ימחק מבסיס הנתונים.



## **דרישות/מגבלות להפעלת התוכנה**

לתוכנה יש מספר מגבלות חשובות:

התוכנה מחייבת הרשאה לגשת למיקום של המשתמשים וחיבור לאינטרנט (רשת) בכדי לפעול – העלאת האירועים לבסיס הנתונים ועדכוןם, מחיקת אירועים מתבצעים רק בזמן שהטלפון מחובר לאינטרנט וה-GPS מופעל. בנוסף, שליחת התראות מן האפליקציה תתבצע רק כאשר יש אינטרנט.

יתרה מזאת, בכדי שהאפליקציה תעבוד כראוי, על המשתמש לספק גישה למצלמה, לגלריית התמונות ששמורה בטלפון שלו וגישה ליצירת תמונות חדשות מן האפליקציה.



## מדריך למשתמש

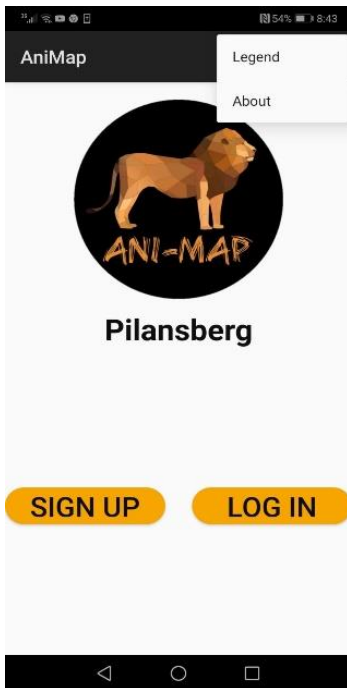
### פירוט מסכים

#### 1. מסך בית - HomePage

זהו המסך הראשון שהמשתמש רואה כאשר הוא נכנס לאפליקציה

פירוט רכיבים במסך-

- לוגו הפרויקט (ImageView)
- טקסט כותרת (TextView)
- שני כפתורים (Button):



SIGN UP

הרשמה - מעביר למסך SignUp

LOG IN

התחברות - לעבור למסך Login

בנוסף, ניתן לעבור לעוד שני מסכים בעזרת menu:

מקרא למפה - Legend

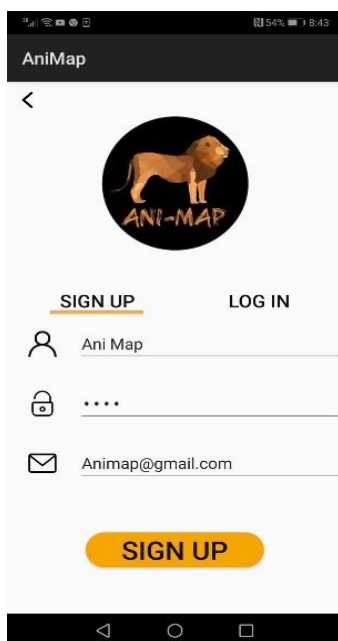
אודות/הסבר על האפליקציה - About

#### 2. הרשמה - SignUp

במסך זה המשתמש יכול להירשם לאפליקציה בכך שזין שם משתמש, סיסמה וכתובת אמייל בשלוש תיבות טקסט (EditText).

פירוט רכיבים במסך-

- לוגו הפרויקט (ImageView)
- טקסט כותרת משנה Sign Up (TextView)
- שלושה כפתורים (Button):



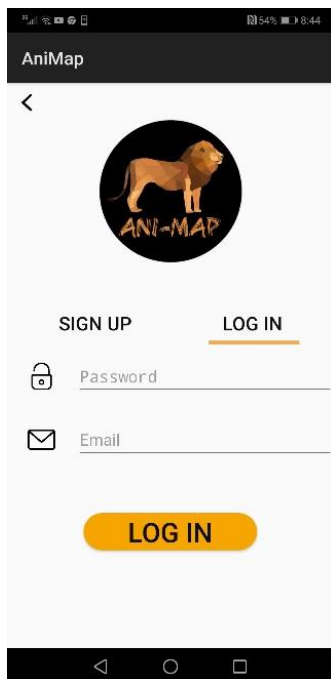
1. חזור - מחזיר למסך HomePage.

2. התחברות - לעבור למסך Login.

3. הרשמה - להירשם ולעבור למסך Map.

SIGN UP

החשבון של המשתמש יועבר דרך השרת לבסיס הנתונים בו האפליקציה משתמשת (Firebase) ויישמר שם.



### 3. התחברות - Login

במסך זה המשתמש יכול להתחבר לאפליקציה בכך שיזין שם משתמש וסיסמה בשתי תיבות טקסט (EditText).

פירוט רכיבים במסך-

- לוגו הפרויקט (ImageView)
- טקסט כותרת משנה Log In (TextView)
- שלושה כפתורים (Button):



1. חזור - מחזיר למסך HomePage.

2. הרשמה - לעבור למסך SignUp.

3. התחברות - להתחבר ולעבור למסך Map.

LOG IN



#### 4. מפה - Map



במסך זה המשתמש יכול לראות את המפה (Fragment) של שמורת הטבע Pilanesberg.

על המפה יש מוקדי עניין / אירועים שמטיילים בשמורה העלו. כל אירוע כזה מיוצג על ידי משתנה מטיפוס Marker בצבע שונה בהתאם לסוג החיה שהוא מתאר (אל המקרא של הצבעים ניתן לעבור דרך ה-Menu). בנוסף, המשתמש יכול לראות את מיקומו על המפה בכל רגע נתון.

פירוט כפתורים-

1. חזור - מחזיר למסך HomePage.
2. העלאות שלי - מעביר למסך YourSpots.
3. הוספת אירוע - מעביר למסך AddSpot.

ניתן ללחוץ על כל אירוע במפה (חוץ מהנקודה שמציגה את המשתמש) ולעבור למסך EventInfo.

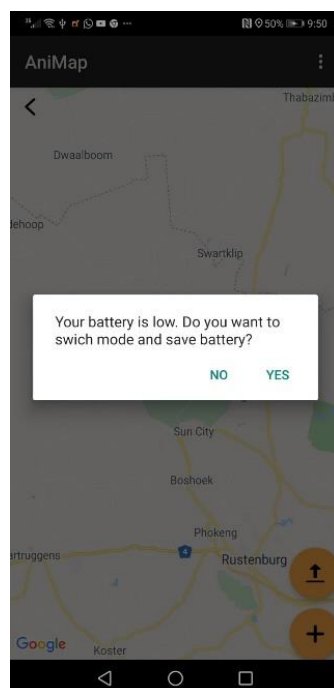
בנוסף, ניתן לעבור לעוד שני מסכים בעזרת menu:

מקרא למפה - Legend

אודות/הסבר על האפליקציה - About

#### -Popup

כאשר המשתמש נמצא במסך זה והסוללה שלו נמוכה, נשלחת הודעה ששואלת את המשתמש האם הוא רוצה לעבור למצב חיסכון בסוללה. במידה והוא מסמן שכן, קצב העדכון של המפה מה-GPS יפחת.





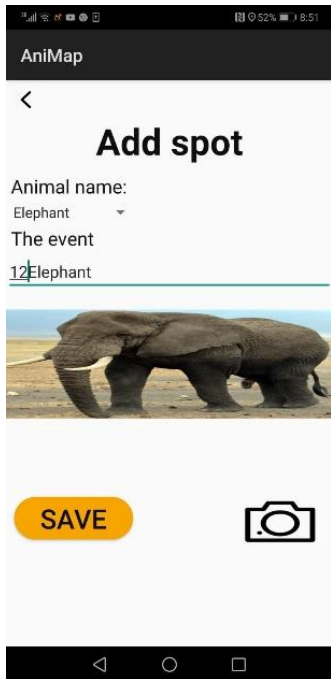
## 5. העלאת נקודה – AddSpot

במסך זה המשתמש יכול ליצור אירוע (Event) משלו ולהעלות אותו. המשתמש צריך תחילה לבחור איזו חיה הוא ראה מבין החיות שמוצגות ב-Spinner ולאחר מכן, לכתוב בתיבת טקסט מה האירוע שראה. המשתמש יכול גם להעלות תמונה של האירוע שתופיע בתוך Image View.

פירוט רכיבים במסך-

- טקסט כותרת ושתי כותרות משנה (TextView).
- בחירת שם החיה (Spinner).
- מקום למלא מה קרה באירוע (EditText)
- שלושה כפתורים (Button):

1. חזור - מחזיר למסך Map.
2. צלם - פותח Popup ששואל את המשתמש אם לפתוח מצלמה או גלריה.
3. שמור – שומר את האירוע ומחזיר למסך Map.

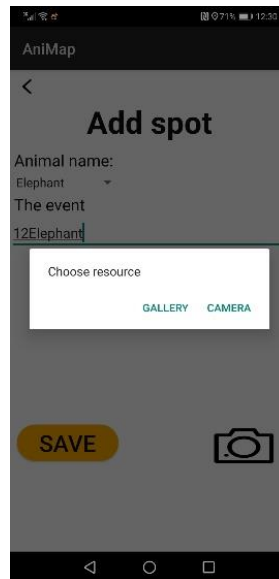




בס"ד

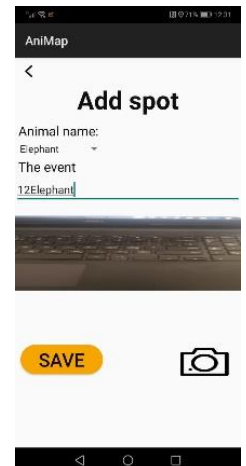
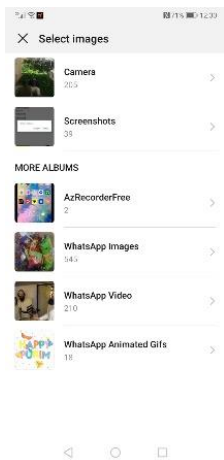
## Ani-Map גל חילילי

-Popup



משתמש בחר Gallery

משתמש בחר Camera



1

2

3

2

1





## 6. מידע על נקודה – EventInfo

במסך זה המשתמש יכול לראות מידע על הנקודה שעליה לחץ במפה כגון: שם החיה, מה האירוע שהועלה, מה המרחק בין מקומך למיקום האירוע, לפני כמה זמן הועלה (אם זמן זה מגיע ל-24 שעות הנקודה תמחק אוטומטית מהמפה) וכמה אנשים ראו את החיה. בנוסף, הוא יכול לראות את התמונה שצרפו לאירוע זה.

פירוט רכיבים במסך-

- טקסט כותרת וארבע כותרות משנה (TextView).
- תיבות טקסט אשר מציגות את המידע על האירוע (TextView)
- התמונה שהמשתמש שהעלה את האירוע צירף (ImageView)
- שלושה כפתורים (Button):



1. חזור - מחזיר למסך Map.
2. האירוע נמצא - מאפס את הזמן ומשנה את המיקום למיקום של המשתמש שלחץ על הכפתור.
3. האירוע לא נמצא - כאשר ה- Counter של כמה אנשים לא ראו את החיה יגיע לחמישים אנשים האירוע ימחק אוטומטית מהמפה.



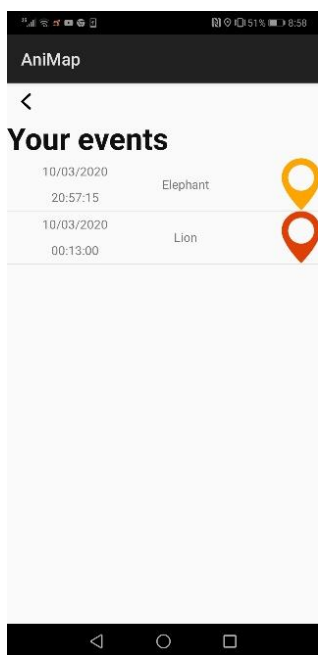
לאחר לחיצה על **FOUND** הזמן מתאפס, המיקום משתנה למיקום של המשתמש שלחץ על הכפתור וה- Counter מתעדכן גם כן. כל השינויים הללו השמרים בבסיס הנתונים.



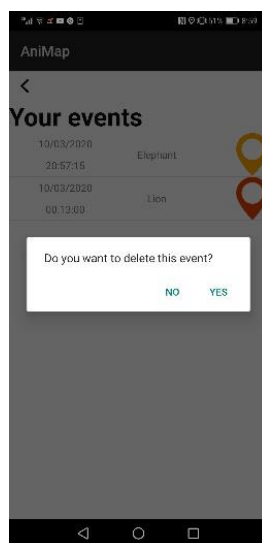
## 7. האירועים שלך – YourSpots

במסך זה כל משתמש יכול לראות List view של הנקודות שהוא העלה ונמצאות עדיין על המפה כלומר, הן הועלו או מישהו לחץ על הנקודות ורענן אותן ( לחץ על **FOUND** ) לפני פחות מ-24 שעות. בכל נקודה הוא רואה את תאריך ושעת ההעלאה, את שם החיה ואת הצבע שמסמל אותה (ImageView). יתרה מזאת, למשתמש יש אפשרות למחוק את הנקודות שהעלה.

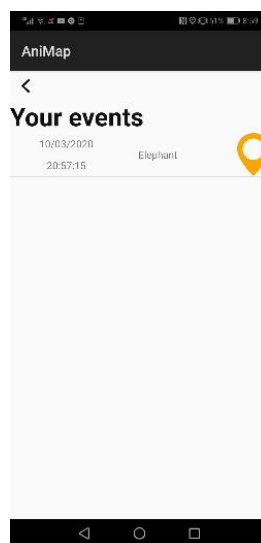
בעמוד זה יש רק כפתור אחד-  
חזור - מחזיר למסך Map.



אם המשתמש לוחץ על אחת מהשורות ב- List view יופיע לו Popup-



1 המשתמש בחר למחוק את הנקודה השנייה



2



## 8. מקרא – Legend

במסך זה המשתמש יכול לראות מקרא של המפה - לכל חיה יש צבע משלה.

בכל שורה יש ImageView שמראה את הצבע שמסמל את החיה על המפה ו- TextView שמציג את שם החיה.

בעמוד זה יש רק כפתור אחד-

חזור - מחזיר למסך HomePage או Map. <

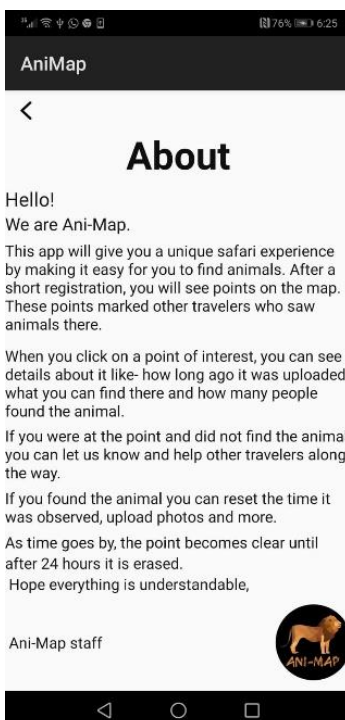


## 9. אודות – About

במסך זה המשתמש יכול קרוא קצת על האפליקציה ואיך היא עובדת.

בעמוד זה יש רק כפתור אחד-

חזור - מחזיר למסך HomePage או Map. <





## בסיס נתונים

### שיטת שמירת הנתונים

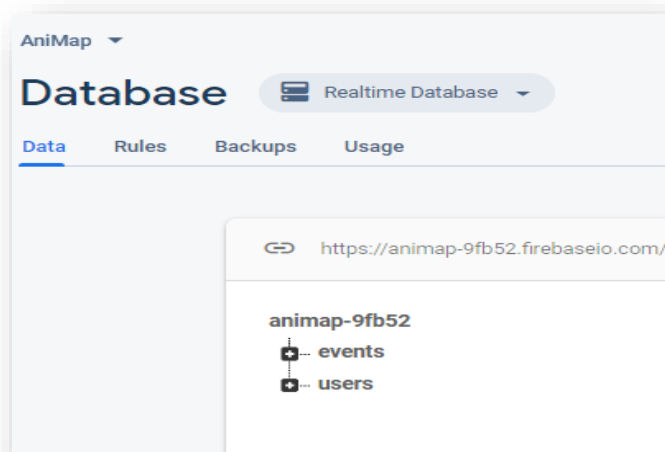
האפליקציה שומרת נתונים בשלוש דרכים:

1. **FirestoreDatabase** – בדרך זו נשמרים כמעט כל הנתונים שהאפליקציה שומרת, וביניהם: מידע על כל המשתמשים והאירועים. דרך ה-database מתבצעים גם השינויים בכל שדות אלן.
2. **FirestoreStorage** – בדרך זו נשמרות התמונות שהמשתמש מעלה לשרת מהאירוע שתיעד. כל פעם שמתבצעת העלאת תמונות או שליפתן, ה-S-torage מתעדכן.
3. **SharedPreferences** – הנתונים שנשמרים כך הם נתונים שעוברים ממסך אחד לשני ורק אותו משתמש צריך לראות אותם או את השינוי שנעשה בהם לדוגמה המידע על החשבון שלו, שכולל את כתובת המייל שלו ועוד.



## Database

### רשימת העמודות בטבלה



המפתחות הראשיים בבסיס הנתונים הם-  
events - האירועים שהמשתמשים האלו.  
users - המשתמשים שנרשמו לאפליקציה.

### תתי עץ של events:

כל עץ שבו שמור המידע על אחד האירועים הוא מטיפוס Event.  
כאשר משתמש יוצר אירוע חדש בעמוד AddSpotActivity,  
הוא בעצם יוצר משתנה חדש מטיפוס Event עם כל הפרטים שהזין  
ואובייקט זה נשמר בבסיס הנתונים.

התמונה היא לא שדה חובה מכיוון שלמשתמש שמעלה את יש  
אפשרות לא לצרף תמונה אך, שאר השדות הם בגדר חובה והם  
לא יכולים להיות null.



שם תת העץ	מה הוא מכיל	טיפוס
Date	תאריך העלאת האירוע	String
eventId	זמן העלאתו ומה קרה באירוע	String
eventName	שם החיה שנראתה באירוע	String
Image	התמונה שצילם מעלה האירוע	String
Latitude	קו רוחב של מיקום האירוע	String
Longitude	קו אורך של מיקום האירוע	String
Time	זמן העלאת האירוע	String
unexistsCounter	מונה כמה אנשים לא ראו את החיה	int
watchCounter	מונה כמה אנשים ראו את החיה	int
uploadUserName	שם מעלה האירוע	String
uploadUserEmail	אמייל מעלה האירוע	String

טבלת תתי העצים/ התכונות  
של events

עץ האירועים -> עץ האירוע ->

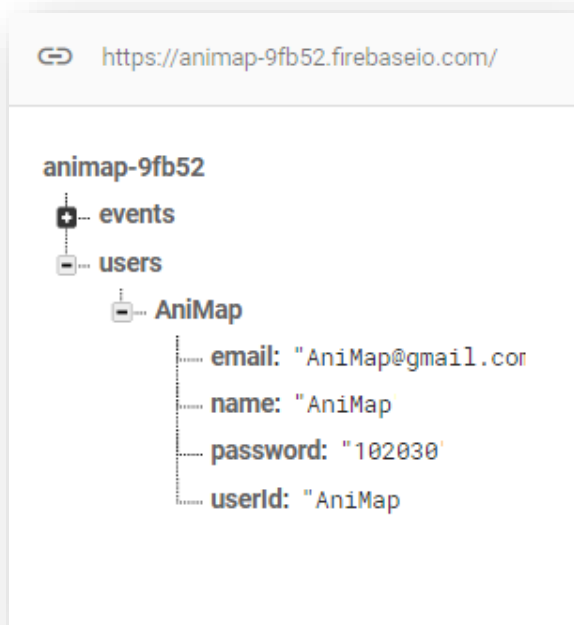
תאריך העלאת האירוע	(טיפוס String)
מזהה האירוע (ID)	(טיפוס String)
זמן העלאת האירוע	(טיפוס String)
שם החיה	(טיפוס String)
תמונה	(טיפוס String)
קו אורך	(טיפוס String)
קו רוחב	(טיפוס String)
שם מעלה האירוע	(טיפוס String)
כתובת מייל מעלה האירוע	(טיפוס String)
מונה אנשים שראו	(טיפוס int)
מונה אנשים שלא ראו	(טיפוס int)



## תתי עץ של users :

כל עץ שבו שמור המידע על אחד המשתמשים הוא מטיפוס User. כאשר משתמש נרשם לאפליקציה ויוצר משתמש בעמוד SignUpActivity, הוא בעצם יוצר משתנה חדש מטיפוס User עם כל הפרטים שהזין ואובייקט זה נשמר בבסיס הנתונים.

השדות בתת עץ זה הינם בגדר חובה והם לא יכולים להיות null.



## טבלת תתי העצים/ התכונות של users

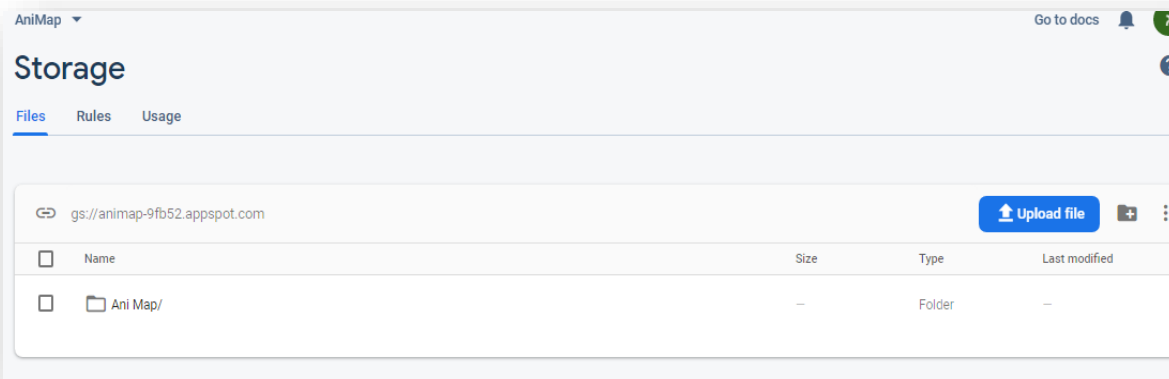
שם תת העץ	מה הוא מכיל	טיפוס
email	כתובת המייל של המשתמש	String
name	שם המשתמש	String
password	סיסמה שהמשתמש הזין	String
userId	תחילת כתובת המייל של המשתמש (עד השטרודל)	String

עץ המשתמשים -> עץ המשתמש ->

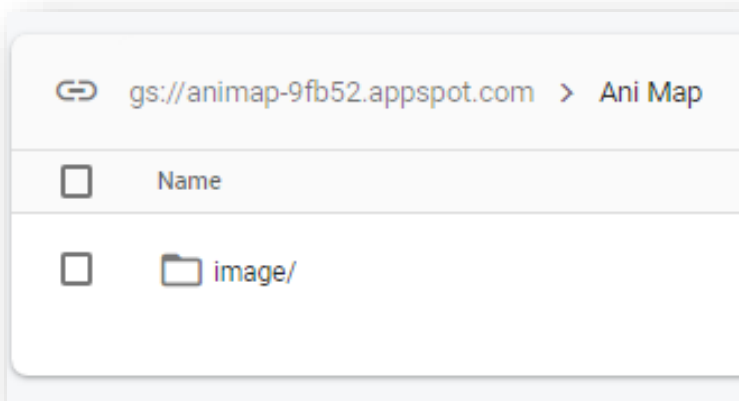
כתובת המייל של המשתמש	(טיפוס String)
שם המשתמש	(טיפוס String)
סיסמה	(טיפוס String)
תחילת כתובת המייל של המשתמש (עד השטרודל)	(טיפוס String)



## Storage



שמירת תמונות  
ב-Storage על  
פי שם  
המשתמש



לאחר לחיצה על תיקיית התמונות  
של אחד המשתמשים





## כל התמונות שאותו משתמש העלה

gs://animap-9fb52.appspot.com > Ani Map > image

Upload file





<input type="checkbox"/>	Name	Size	Type	Last modified
<input type="checkbox"/>	 image12:31:58	463.33 KB	image/jpeg	Mar 11, 2020
<input type="checkbox"/>	 image13:15:13	2.33 MB	image/jpeg	Feb 27, 2020
<input type="checkbox"/>	 image19:06:48	27.68 KB	image/jpeg	Feb 24, 2020

image12:31:58



Name  
[image12:31:58](#)

Size  
474,454 bytes

Type  
image/jpeg

Created  
Mar 11, 2020, 12:32:04 PM

Updated  
Mar 11, 2020, 12:32:04 PM

שם המשתמש -> תיקיית התמונות -> שם התמונה

Username -> image -> UploadTime

ב-FirebaseStorage, לכל משתמש שהעלה תמונה אחת לפחות נוצרת תיקייה ששמה זהה לשם שהכניס בדף ההרשמה. תחת תיקייה זו יש תיקיה בשם image שבתוכה נשמרות כל התמונות שהמשתמש העלה. לדוגמה, אם העליתי תמונה לשרת דרך האפליקציה, אמצא אותה במקום הבא:  
Name/image/image10:50:09

שם הקובץ של התמונה יתחיל תמיד עם הטקסט image ואחריו, תבוא השעה שבא העלה המשתמש את האירוע שכולל את התמונה הזו.



## SharedPreferences

טבלה זו מציגה את כל המחלקות שבהן נעשה שימוש ב- SharedPreferences, מה המידע שנשמר ובאילו מחלקות משתמשים בו.

המחלקה/ות שמשתמשות	המידע שנשמר	המחלקה ששומרת
NearByLocationService YourSpotsActivity AddSpotActiviy MapActivity	שם, סיסמה וכתובת אמייל של הנרשם	SignUpActivity
EventInfoActivity	ה-ID של האירוע שעליו לחץ המשתמש	MapActivity
EventInfoActivity	את ה-ID של האירוע שהועלה ואת זמן העלאת התמונה	AddSpotActiviy
EventInfoActivity	ה-ID של האירוע שעליו האפליקציה שולחת התראה (Notification)	NearByLocationService



## מדריך למפתח

### מבנה נתונים בהם נעשה שימוש

במחלקה YourSpotsActivity ישנם שני משתנים מטיפוס ArrayList:

1. שומר בתוכו את כל האירועים שנמצאים בבסיס הנתונים (Firestore).
2. שומר רק את האירועים של המשתמש שעכשיו באפליקציה (מתוך כל האירועים שבתוך ה- ArrayList הראשון).

במחלקה NearByLocationService גם כן ישנם שני משתנים מטיפוס ArrayList:

1. שומר בתוכו את כל האירועים שנמצאים בבסיס הנתונים (Firestore).
2. שומר רק את האירועים שקרובים למיקום המשתמש והועלו לפני פחות מ-24 שעות (מתוך כל האירועים שבתוך ה- ArrayList הראשון).

במחלקה MapActivity יש ArrayList אחד ששומר את כל האירועים שנמצאים בבסיס הנתונים (Firestore). בנוסף, יש HashMap אחד שגם ה- Key וגם הערכים שהוא שומר הם מטיפוס String. הערכים שהוא שומר הם בעצם ה- eventId של האירוע שכל marker מציין.



## מידע על המחלקות ותיעוד הקוד

שם המחלקה: `HomePageActivity`

תפקיד המחלקה: העמוד הראשי של האפליקציה, משם המשתמש יכול לעבור לעמוד ההרשמה (`SignUpActivity`) או לעמוד ההתחברות (`LogInActivity`).

תוכן המחלקה / תדפיס קוד:

- `onCreateOptionsMenu`

```
public boolean onCreateOptionsMenu(android.view.Menu menu)
```

This function creates a menu and inflates it from XML resource.

Overrides:

```
onCreateOptionsMenu in class android.app.Activity
```

Parameters:

menu - The options menu.

- `onOptionsItemSelected`

```
public boolean onOptionsItemSelected(android.view.MenuItem item)
```

Called whenever an item in the menu is selected.

Overrides:

```
onOptionsItemSelected in class android.app.Activity
```

Parameters:

item - the selected item.

- `onClick`

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

`btnSignUpHomepage`- Going to `SignUpActivity`.

`btnLogInHomepage` - Going to `LogInActivity`.

Specified by:

```
onClick in interface android.view.View.OnClickListener
```

Parameters:

view - The view that was clicked.



שם המחלקה: SignUpActivity

תפקיד המחלקה: עמוד הרשמה לאפליקציה, משם המשתמש יכול לעבור לעמוד ההתחברות (LogInActivitiy).

תוכן המחלקה / תדפיס קוד:

- *onCreate*

```
protected void onCreate(android.os.Bundle savedInstanceState)
```

This function saves in SharedPreferences the user name, email address and password and uses this information.

**Overrides:**

```
onCreate in class android.app.Activity
```

**Parameters:**

```
savedInstanceState.
```

- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBack- Going back to HomeActivity.

btnSignUp-Saving information on the user in the SharedPreferences and the Firebase and going to MapActivity.  
btnLogIn-Going to LogInActivity.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```



שם המחלקה: LoginActivity

תפקיד המחלקה: עמוד ההתחברות לאפליקציה, משם המשתמש יכול לעבור לעמוד ההתחברות (SignUpActivity).

תוכן המחלקה / תדפיס קוד:

#### *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBack- Going back to HomeActivity.

btnSignUp-Going to SignUpActivity.

btnLogIn-Going to MapActivity.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```



## שם המחלקה: MapActivity

**תפקיד המחלקה:** עמוד שמציג את המפה ועליה האירועים שהמשתמשים האלו, משם המשתמש יכול להגיע לכל מסכי האפליקציה.

## תוכן המחלקה / תדפיס קוד:

- *onCreate*

```
protected void onCreate(android.os.Bundle savedInstanceState)
```

This function activates GPS, stars the LowBatteryBroadcastReceiver and marks the relevant events on the map.

**Overrides:**

```
onCreate in class android.support.v4.app.FragmentActivity
```

**Parameters:**

```
savedInstanceState.
```

- *onLocationChanged*

```
public void onLocationChanged(android.location.Location location)
```

This function is called when the user location has changed, saves the current location and sends it to NearByLocationService.

**Specified by:**

```
onLocationChanged in interface android.location.LocationListener
```

**Parameters:**

```
location - current user location.
```

- *onStart*

```
public void onStart()
```

This function stars the BroadcastReceiver.

**Overrides:**

```
onStart in class android.support.v4.app.FragmentActivity
```

- *onStop*

```
public void onStop()
```

This function stops the BroadcastReceiver

**Overrides:**

```
onStop in class android.support.v4.app.FragmentActivity
```



- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBack- Going back to HomeActivity.

fabAdd- Going to AddSpotActivity.

fabYourEvents- Going to YourSpotsActivity.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```

- *onCreateOptionsMenu*

```
public boolean onCreateOptionsMenu(android.view.Menu menu)
```

This function creates menu and inflates a menu from XML resource.

**Overrides:**

```
onCreateOptionsMenu in class android.app.Activity
```

**Parameters:**

```
menu - The options menu.
```

- *onOptionsItemSelected*

```
public boolean onOptionsItemSelected(android.view.MenuItem item)
```

This function is called whenever an item in your options menu is selected.

**Overrides:**

```
onOptionsItemSelected in class android.app.Activity
```

**Parameters:**

```
item - the selected item.
```

- *onMapReady*

```
public void onMapReady(com.google.android.gms.maps.GoogleMap googleMap)
```

This function is called when the map is ready to be used.

**Specified by:**

```
onMapReady in interface com.google.android.gms.maps.OnMapReadyCallback
```

**Parameters:**

```
googleMap - A non-null instance of a GoogleMap associated with the MapFragment.
```





- *isMyServiceRunning*

```
public boolean isMyServiceRunning(java.lang.Class<?> serviceClass)
```

This function checks if the service is already running.

**Parameters:**

`serviceClass`

**Returns:**

A boolean object that tells if the service was running.



## שם המחלקה: AddSpotActivitiy

תפקיד המחלקה: מאפשרת למשתמש להעלות אירועים ותמונות, שומרת אותם ב- Firebase וב- Storage בהתאמה.

## תוכן המחלקה / תדפיס קוד:

- onCreate

```
protected void onCreate(android.os.Bundle savedInstanceState)
```

This function activates GPS.

**Overrides:**

```
onCreate in class android.app.Activity
```

**Parameters:**

```
savedInstanceState.
```

- onActivityResult

- ```
protected void onActivityResult(int requestCode,
                                int resultCode,
                                android.content.Intent data)
```

Called when an activity you launched exits, giving you the requestCode you started it with, the resultCode it returned, and any additional data from it - The image. Then the image is saved on the ImageView.

**Overrides:**

```
onActivityResult in class android.app.Activity
```

**Parameters:**

```
requestCode - The integer request code, allowing you to identify who this
result came from.
```

```
resultCode - The integer result code returned by the child activity.
```

```
data - An Intent, which can return result data to the caller (The image).
```

- onClick

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBack- Going back to MapActivity.

btnSave- Saves the event in the Firebase and the image in the Storage, then going to MapActivity.

btnAddImage- Displays a popup who asks which way the user want to upload a photo- Through the gallery or to take a picture.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```



- *onLocationChanged*

```
public void onLocationChanged(android.location.Location location)
```

Receiving notifications from the LocationManager when the location has changed.

**Specified by:**

```
onLocationChanged in interface android.location.LocationListener
```

**Parameters:**

```
location - current user location.
```

- *onProviderDisabled*

```
public void onProviderDisabled(java.lang.String provider)
```

Called when the provider is enabled by the user.

**Specified by:**

```
onProviderDisabled in interface android.location.LocationListener
```

**Parameters:**

```
provider - the name of the location provider that has become enabled.
```

- *onItemSelected*

- ```
public void onItemSelected(android.widget.AdapterView<?> adapterView,  
                           android.view.View view,  
                           int i,  
                           long l)
```

This function is called when the user choose animal name from the spinner

**Specified by:**

```
onItemSelected in interface android.widget.AdapterView.OnItemSelectedListener
```

**Parameters:**

```
adapterView - The AdapterView where the selection happened.
```

```
view - The view that was clicked.
```

```
i - The position of the view in the adapter.
```

```
l - The row id of the item that is selected.
```



שם המחלקה: EventInfoActivity

תפקיד המחלקה: להראות למשתמש את המידע על האירוע שעליו לחץ במפה.

תוכן המחלקה / תדפיס קוד:

- *onCreate*

```
protected void onCreate(android.os.Bundle savedInstanceState)
```

This function activates GPS, updates all event's information including how many people saw the animal, how long ago the event was uploaded, what the event was and the image that the user uploaded.

**Overrides:**

```
onCreate in class android.app.Activity
```

**Parameters:**

```
savedInstanceState.
```

- *onLocationChanged*

```
public void onLocationChanged(android.location.Location location)
```

Receiving notifications from the LocationManager when the location has changed. In addition, calculates the distance between the event and user current location in meters.

**Specified by:**

```
onLocationChanged in interface android.location.LocationListener
```

**Parameters:**

```
location - current user location.
```

- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBack- Going back to the MapActivity.

btnFound- Updates time, location and watchCounter in database.

btnNotFound- Updates the unexistsCounter in the database.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```



שם המחלקה: YourSpotsActivity

תפקיד המחלקה: מאפשרת למשתמש לראות את כל האירועים שהוא העלה ולמחוק אותם.

תוכן המחלקה / תדפיס קוד:

- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item. In this case on click on:

btnBackYourE- Going back to the MapActivity.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```



שם המחלקה: LegendActivity

תפקיד המחלקה: מציגה למשתמש מקרא של המפה שכולל את שם החיה ואת הסימון שלה על המפה.

תוכן המחלקה / תדפיס קוד:

- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item.

btnBack- Going back to the activity he came from.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```

שם המחלקה: AboutActivity

תפקיד המחלקה: מסבירה למשתמש מה האפליקציה עושה ואיך היא עובדת.

תוכן המחלקה / תדפיס קוד:

- *onClick*

```
public void onClick(android.view.View view)
```

This function is called every time a user clicks an item.

btnBack- Going back to the activity he came from.

**Specified by:**

```
onClick in interface android.view.View.OnClickListener
```

**Parameters:**

```
view - The view that was clicked.
```

**שם המחלקה:** `NearByLocationService`

**תפקיד המחלקה:** שולחת למשתמש התראה על אירוע שקרוב אליו והועלה לפני הכי פחות זמן.

**תוכן המחלקה / תדפיס קוד:**

- *onStartCommand*

- ```
public int onStartCommand(android.content.Intent intent,
                           int flags,
                           int startId)
```

This function is called by the system every time the service starts. Also it gets in intent the user location from MapActivity.

**Overrides:**

```
onStartCommand in class android.app.Service
```

**Parameters:**

`intent` - The Intent supplied to Context.

`flags` - Additional data about this start request.

`startId` - A unique integer representing this specific request to start.

- *differenceBetweenTimes*

- ```
public static int differenceBetweenTimes(java.lang.String time,
                                         java.lang.String date)
```

This function calculates how long it has been since the event was uploaded.

**Parameters:**

`time` - event UploadTime.

`date` - event uploadDate.

**Returns:**

How long it has been since the point was raised in minutes.

**שם המחלקה:** LowBatteryBroadcastReceiver

**תפקיד המחלקה:** בודקת את מצב הסוללה של המשתמש. אם יש לו סוללה נמוכה, מוצג לו popup ששואל האם ירצה לעבור למצב חסכוני.

**תוכן המחלקה / תדפיס קוד:**

**onReceive**

```
public void onReceive(android.content.Context context,  
                      android.content.Intent intent)
```

This method is called when the BroadcastReceiver is receiving an Intent broadcast. It also checks if the user's battery is less than 20% and if so, suggest it switch to economical mode (in a popup). If the battery exceeds 20% then the battery status is canceled.

**Specified by:**

onReceive in class android.content.BroadcastReceiver

**Parameters:**

context - The Context in which the receiver is running.

intent - The Intent being received.





שם המחלקה: EventListAdapter

תפקיד המחלקה: מעדכנת את המידע ב-ListView לגבי האירועים שמוצגים בו.

תוכן המחלקה / תדפיס קוד:

#### *getView*

```
@NonNull
public android.view.View getView(int position,
                                @Nullable
                                android.view.View convertView,
                                @NonNull
                                android.view.ViewGroup parent)
```

Get a View that displays the data at the specified position in the data set.

**Specified by:**

getView in interface android.widget.Adapter

**Overrides:**

getView in class android.widget.ArrayAdapter<[Event](#)>

**Parameters:**

position - the item position in the ListView.

convertView - The old view to reuse, if possible.

parent - The parent that this view will eventually be attached to (the ListView).



שם המחלקה: Event

תפקיד המחלקה: מגדירה את תכונות האירועים.

תוכן המחלקה / תדפיס קוד:

Event

```
public Event()  
  
Event(java.lang.String eventId, java.lang.String animalName,  
java.lang.String image, java.lang.String time, java.lang.String date,  
int watchCounter, int unexistsCounter, java.lang.String uploadUserName,  
java.lang.String uploadUserEmail, java.lang.String longitude,  
java.lang.String latitude)
```

שם המחלקה: User

תפקיד המחלקה: מגדירה את תכונות המשתמשים.

תוכן המחלקה / תדפיס קוד:

User

```
public User()  
  
User(java.lang.String name, java.lang.String email, java.lang.String password)
```



## סיכום אישי

העבודה על הפרויקט הייתה עבורי חוויה מאוד לימודית ומעצימה. נהייתי לחקור דברים חדשים, לעצב מסכים, לעבוד עם אנדרואיד סטודיו בפעם הראשונה ובכללי להיחשף לכל עולם עיצוב ובניית אפליקציות.

היו נקודות שניתקתי בהן במהלך כתיבת הקוד והייתי צריכה להשקיע מחשבה בכדי לפתור אותן בצורה מיטבית. לפעמים לא הצלחתי לפתור בעיה מסוימת מה שגרם לתחושת תסכול קלה אך כשהצלחתי לפתור אותה, התחושה הייתה מאוד מספקת.

במהלך העבודה על הפרויקט צברתי מספר כלים חשובים ומשמעותיים. למדתי להשתמש באנדרואיד סטודיו, לכתוב קוד ולעבוד עם בסיס הנתונים Firebase, דבר שיכול לעזור לי בכל עבודה עתידית ובכללי בחיים. למדתי לפתור בעיות בצורה יעילה, להשתמש בכלים שסביבי ולא מיד לבקש עזרה מחברים או מהמורה. בנוסף, למדתי לעצב מסכים בצורה נוחה ושימושית כדי שלמשתמש יהיה קל לתפעל את האפליקציה. דבר רביעי ואחרון למדתי לקחת הכל בצורה טובה וחיובית וללמוד מכל בעיה.

המסקנות שלי מהעבודה על הפרויקט הן שאין דבר העומד בפני הרצון. בחטיבה, בכלל לא ידעתי מה זה קוד ואיך הוא עובד, לא ממש הבנתי בתחום המחשבים ואפילו הילדים בכיתה שלי היו צוחקים שאני לא יודעת לעשות העתק הדבק. בכל זאת החלטתי ללכת למגמת מחשבים כי רציתי להיחשף לעולמות חדשים שאני לא מכירה וללמוד דברים חדשים - אני חושבת שזאת אחת ההחלטות הטובות שעשיתי. למדתי במגמה הזאת ובפרט בעבודה על הפרויקט כל כך הרבה דברים חשובים שלא הייתי לומדת בשום מגמה אחרת.

אם הייתי מתחילה את העבודה על הפרויקט היום הייתי כותבת הערות על הקוד מבעוד מועד ולא בסוף. דבר נוסף שהייתי משנה הוא שהייתי מתכננת את שלבי העבודה שלי אחרת וכך חוסכת בזמן שבו הייתי מוסיפה אפשרות של חיפוש נקודות על המפה ועושה בדיקת תקינות קלט לאמייל.

## צדדים חזקים/חלשים בפרויקט

בפרויקט צדדים חזקים צדדים חלשים.

הפרויקט עובד כמתוכנן בלי באגים ובעיות. העיצוב של הפרויקט עדין ולא מקושקש מידי כך שכל משתמש יכול להתחבר אליו. בנוסף לכך, מערכת יצירת האירועים והופעתם על גבי המפה הינה מערכת נוחה שמספקת למשתמש את כל המידע הנחוץ לו בשביל טיול מושלם ומלא בחיות.

מצד שני, ישנם גם צדדים חלשים לאפליקציה. טעינת תמונות כבדות תיקח לאפליקציה יותר זמן (בדרך כלל כמה שניות). יתר על כן, אין בדיקת תקינות קלט לאמייל. דבר אחרון הוא שאי אפשר להעלות אירוע על כל החיות בשמורה אלא רק על החיות המעניינות ביותר שקבועות מראש בספינר (Spinner).

לסיכום, מאוד נהייתי מהעבודה על הפרויקט שלי ואני כבר מצפה להתחיל לעבוד על אפליקציה חדשה.