# דו"ח עבודה

מגישים: אריה גרוס

אבי ווסרברגר

## תיאור המערכת

### המערכת כוללת שלושה מחלקות מרכזיות:

**המחלקה WiggleWifi** שמעבירה מקבצי WiggleWifi למסד נתונים, הפונקציה המרכזית מקבלת נתיב של תקיה עם קבצים כאלו בתוכה

**המחלקה CSV** שמעבירה ממסד נתונים לקובץ CSV ומקובץ CSV למסד נתונים, המחלקה מכילה שתי פונקציות מרכזיות, אחת מקבלת מסד נתונים ונתיב של קובץ ויוצרת csv, מחזירה שגיאה אם יש בעיה עם הקובץ. ושנייה מקבלת קובץ ויוצרת מסד נתונים

**המחלקה KML** מעבירה ממסד נתונים לקובץ KML, המחלקה מכילה פונקציה מרכזית אחת שמקבלת מסד נתונים ונתיב של קובץ ויוצרת קובץ, הקובץ יממש timeline

### מחלקות עזר:

**המחלקה databaseFunctions** מכילה פונקציה שמוחקת ממסד הנתונים mac כפולים ומשאירה את הגבוה ביניהם, פונקציה שמחזירה את הMAC הכפולים יחד, פונקציה שמכניסה מאגר נתונים אחד לתוך השני, פונקציה ששואלת את המשתמש באיזה פילטר להשתמש ומחזירה אותו

**המחלקה genericFunctions** נועד בשביל פונקציות בודדות, מכילה פונקציה שסופרת מספר שורות בקובץ ומחזירה (הורדנו מהרשת ושינינו, כתוב בקוד עצמו בהערות)

המחלקה find places מכילה פונקציה פומבית שבהנתן מסד נתונים של סריקות משערכת את מיקום הראוטרים (בעזרת מספר שמייצג לפי כמה סריקות לבדוק) ומכילה פונקציה שבהנתן שני מסדי נתונים משערכת באמצעות הראשון את מיקום הסריקות של השני (נעזרת במספר שמייצג בכמה סריקות להשתמש כדי לשערך)

**המחלקה test** מהווה מחלקת בדיקות לכלל הפרוייקט, משתמשת בקבצים שנמצאים בתקיה test

**המחלקה ex0** מהווה את הפונקציות שבקשו מאיתנו בex0

**המחלקה ex2** מהווה את האלגוריתמים שבקשו מאיתנו בex2

### עצמים:

**העצם wifi** שמכיל פרטים על wifi

**העצם** **sameScanWifi** שמכיל עד 10 wifi ומיקום איפה נמדד ות.ז. של המכשיר המודד וזמן המדידה

**העצם wifiWithCoordinates** יורש מwifi וכן מכיל קואורדינטות (נועד למציאת מיקום wifi בודד)

מאגר מידע הוא בעצם ווקטור של sameScanWifi

### ממשקים:

**ממשק ומימוש של סינונים**, הממשק מכיל נתונים לסינון ופונקציה שנועדה למימוש של הסינון עצמו

הפונקציות משתמשות בפונקציה שמצאנו ברשת שסופרת מספר שורות בקובץ (המקור רשום שם).

## כלי תוכנה

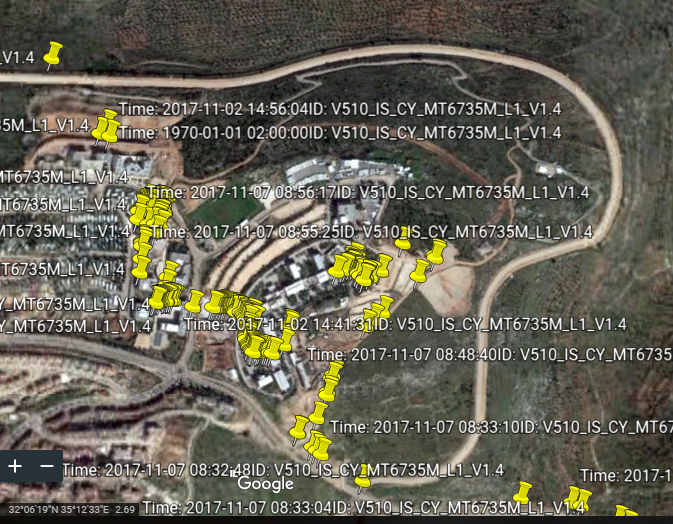
השתמשנו בספריות הרגילות בג'אווה, בממשק, בלכידת שגיאות [try-catch](על מנת לסנן קלט שגוי) וב java api for kml (jak) ובאיטרטור ובווקטור ןב hashmap ובירושה

## בדיקות

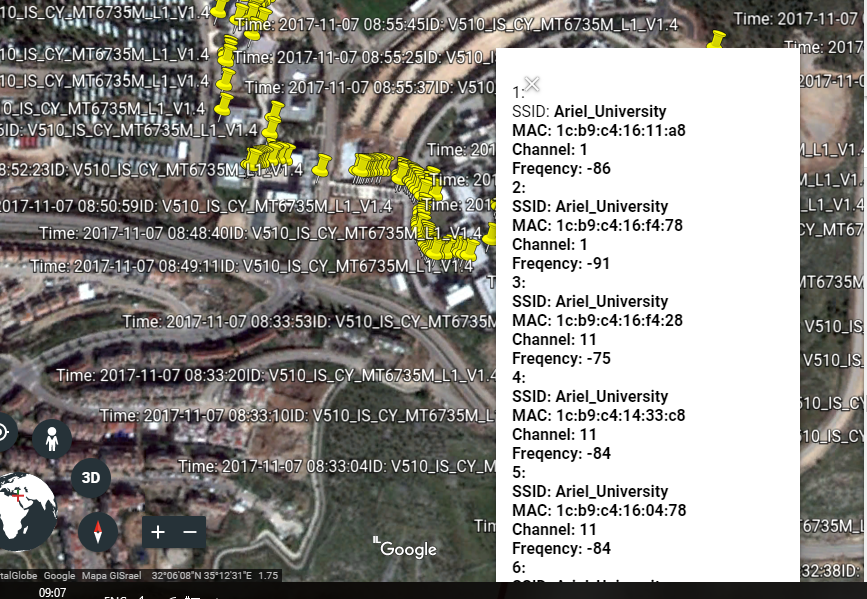
מצורפת תיקיה בשם test עם קבצי אפליקציה תקינים ולא תקינים וכן קבצי csv כנ"ל ומחלקת junit בשם test שבודקת את האפליקציה בעזרתם

# מטלה 0

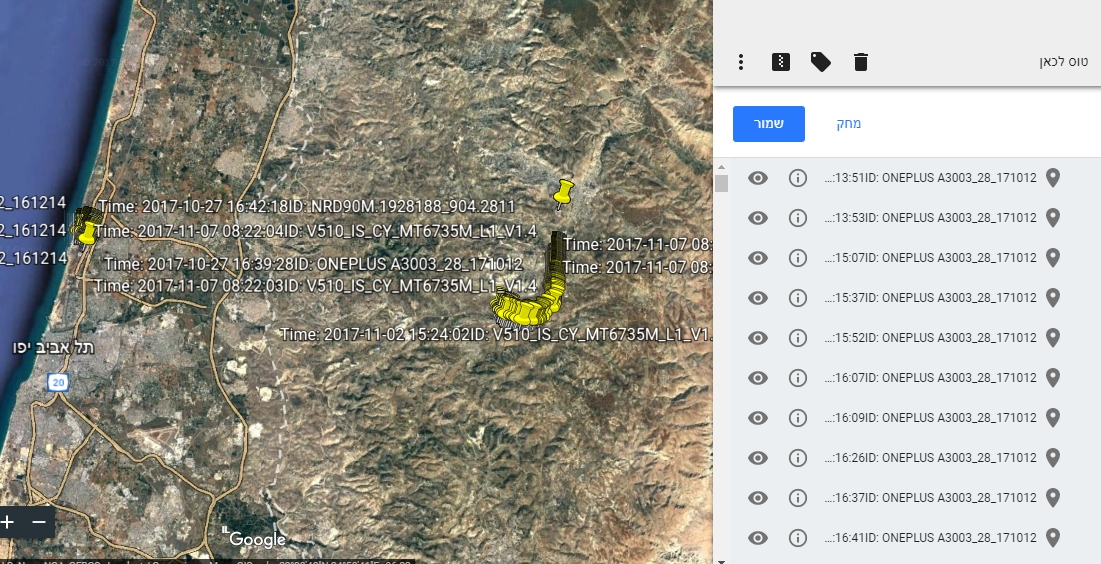
## ניסויים

ביצענו ניסוי באוניברסיטה

למעלה ניתן לראות את נקודות הדגימה עם זמן והID של הבודק

למטה לחצנו על אחת נקודות הדגימה כדי שתציג את התוכן

תמונה של כלל הדגימות כולל הדגימות שהועלו למודל ונסיעה מהר ברכה לאוניברסיטה (אפשר לראות שהgps של הפלאפון לא מדוייק בדגימה הבודדת שנמצאת בשכם בערך, עושה הרבה בעיות עם waze):



# מטלה 1

1. בוצע
2. <https://github.com/amgross/munche_prog>
3. ביצענו חלוקה למחלקות והוספנו עצמים
4. קיים, בתיקייה test
5. השתמשנו בjak, היא עושה רושם התוכנה הכי ידועה ורשמית שעושה את זה ולכן יש לה הרבה תוספות ופיצ'רים וכן הכי הרבה שאלות והסברים ברשת עליה
6. הוספנו
7. הוספנו

# מטלה 2

שינינו את המבנה שישמש למבנה נתונים כללי ולא כפונקציות שונות, שאר הפירוט נמצא בתקיה docs בקובץ word שם