



## המכללה האקדמית ספיר המחלקה למדעי המחשב



**PotHole Tracking System (PTS)**

**אפליקציה לניתור משטח הנסיעה בזמן אמת**

צוות פיתוח: יעקב מטאו ורומן גכטברג  
מנחה: ד"ר גיא לשם

## תוכן עניינים

	1. הקדמה
3	1.1 רקע כללי
5	1.2 ארכיטקטורת המערכת
7	1.3 ניתוח שוק
8	1.4 מודל עסקי ופוטנציאל רווח
	2. תשתיות
9	2.1 TensorFlow
11	2.2 OpenCV
13	2.3 Android
14	2.4 PHP
15	2.5 MySQL
16	3. סקירת מסכים
23	4. תרשים פעילות המשתמשים
24	5. ביבליוגרפיה
25	6. קוד

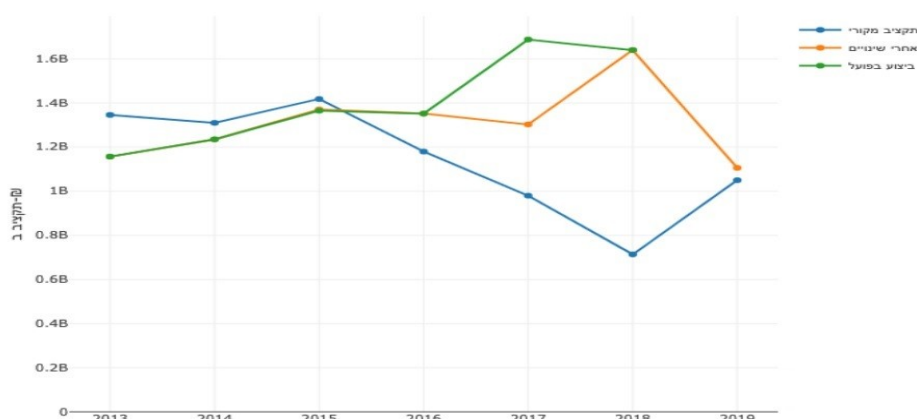
## הבעיה

במהלך נסיעות רבות בכבישי ישראל גילינו, כי למרות שמתבצעת תחזוקה שותפת מדי יום בכל הכבישים במדינת ישראל, עדיין קיימת כמות נכרת של בורות, סדקים ובעיות תחזוקה נוספות בכל הכבישים מדרום ועד צפון המדינה. דבר גורם לנזקים של כלי רכב רבים עולים על הכביש וכתוצאה מכך – כמות התביעות אדירה שמקבלים גופים ממשלתיים האחראים על אותם כבישים מדי שנה. רק חברת "נתיבי ישראל" שאחראית על כ-8000 ק"מ של כבישים בשנת 2019 קיבלה תביעות על סך של כ-10 מיליון שקלים עקב נזקים שנגרמו לכלי רכב בכבישים.

נוסף לנזקים הכספיים קיימת גם בעיית התאונות שנגרמות מאי תקינות של הכבישים. לפי סטטיסטיקת משרד התחבורה: כאחת מעשרים תאונות נגרמת מכך שהכביש לא היה לגמרי תקין לשימוש. (חלק מהמקרים האלה נובע משילוב של אי תקינות עם תנאי מזג האוויר, מקרה הקלסי – שרכב נכנס לשלולית שמסתירה מתחתיה בור עמוק – וכתוצאה מכך הוא מאבד שליטה). חלק מהתאונות האלה נגרם בפגיעות גוף שנות ואף נגרם במוות במקרים חריגים.

ומה הם הפתרונות לכך שקיימים היום? פתרון הראשון: דיווח טלפוני. בן אדם שזיהה תקלה בכביש מתקשר למוקד ומדווח על כך. דבר כזה גורם להרבה בלבול אצל אותו גורם המטפל בשטח כי איננו יכול להבין מהדיווח טלפוני איכן בדיוק התקלה. אך גם לגורם המדווח זה לא כ"כ נוח, כיוון שצריך בשביל כך לעצור בצד דרך ולהמתין דקות ערוכות למענה אנושי, וכתוצאה מכך מעט מאוד אנשים רוצים לעשות דיווחים האלה. פתרון השני: רכב מיוחד מצויד באמצעי סריקה שונים כגון: גלאי קול, חום ומצלמות, אשר פעם בשנה אחרי תקופת הגשמים עובר בכל הכבישים במדינה וסורק אותם. וככה בונים תוכנית שנתית לתחזוקת הכבישים. אך בפועל קיימת פה בעיה של התיישנות המידע, וזאת כיוון שצוותים שמיגיעים לטפל בכביש עושים זאת בדרך כלל הרבה יותר מאוחר מתאריך גילוי של התקלה. וברגע שהם מגיעים לתקלה, הם מגלים שתקלה הרבה יותר חמורה מהמתוכנן או שהתווספו עוד כמה תקלות נוספות בסביבה. וכך בעצם נוצר הפער בתקציב המתוכנן לתיקון לתקציב בפועל.

התקציב של נת"י לתיקונים בשנת 2018 עמד על 713 מיליון אך בפועל שולם כ-1.64 מיליארד שקלים

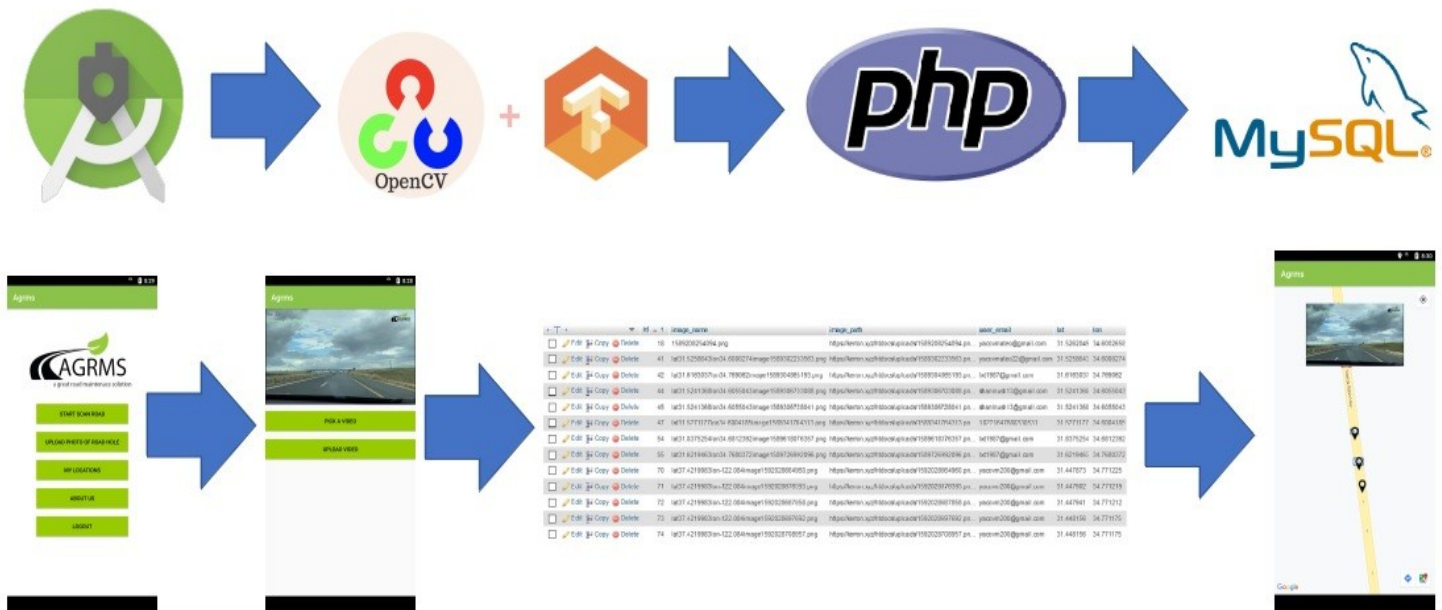


## הפתרון

הפתרון שלנו הוא דיי פשוט: אפליקציית ניתור תקלות בכבישים. כל מה שצריך לעשות זה להתקין את האפליקציה על טלפון נייד שלכם (שהיום נמצא אצל כל אחד ואחד בכיס), לשים את המכשיר נייד על ה"דשבורד" של כלי רכב שלכם, וכל עוד מצלמת הטלפון תהיה מופנית לעבר הכביש – הזיהוי ודיווח של תקלות הולך להתבצע באופן אוטומטי ומדויק. בצורה כזאת אנו נוכל לבנות מפת תקלות ארצית שמעודכנת בזמן אמת וכך גופים המטפלים ידעו להתארך בהתאם ולעשות את העבודה שלהם יותר טוב, דבר יגרום להקטנת מספר התקלות בכבישים.



## איך זה עובד?



מערכת שלנו רצה בסביבת עבודה של אנדרואיד. ברגע שנכנסים לתוכנה, משתמש צריך לבצע הזדהות באמצעות אחת החשבונות של רשתות חברתיות כגון: פייסבוק, גוגל או להירשם עם כתובת אימייל שלו, בנוסף מערכת תבקש הרשאות לשימוש במצלמה וחיבור אינטרנט. כל מה שנותר אחרי זה, זה ללחוץ על כפתור "Start", והסריקה אוטומטית תתחיל.

עם תחילת הסריקה, דבר הראשון שמערכת עושה זה זיהוי אזור הכביש ע"י שימוש בטכנולוגיית ראיית מכונה "OpenCV". ברגע שכביש זוהה בהצלחה האזור הנתון נחתך ועובר לשלב האיבוד תמונה הבא, בו המערכת מנסה לגלות אזורים הפוטנציאליים לתקלה במשטח הכביש.

אזורים הפוטנציאליים שנמצאו מועברים למודל של למידה עמוקה – בה מתבצעת הכרעה האם אכן מדובר בתקלה. אם התקלה זוהתה בהצלחה, מערכת שומרת תמונת אזור התקלה, מוסיפה לזה מיקום המכשיר הנוכחי (בעזרת ה-GPS של המכשיר) ושולחת את הנתונים האלה לשרת המרוחק. השרת מקבל את הנתונים ומכניס אותם לדטהבייס שלו בתוספת זמן המדויק של קבלתם. בשלב הזה גוף האחראי על תיקונים יכול להיכנס לאפליקציה בכל רגע נתון עם משתמש ה"אדמין" שלו ולפתוח מפת האזור כלשהו לפי רצונו – ולראות את כל התקלות שדווחו באזור הזה המעודכנות בזמן אמת.

1.3

## ניתוח שוק



## מתחרים

- חברות ניתוח וידאו
- מערכות לסריקת משטחי המראות המטוסים - ציוד נייד
- ציוד רכוב לביקורת מצב כביש

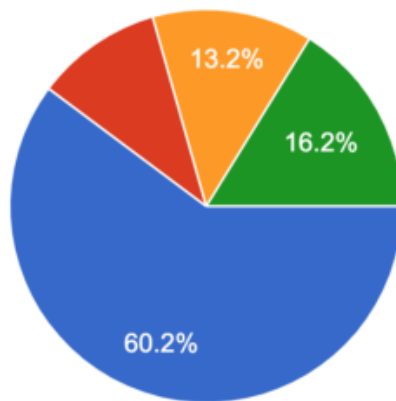
פתרון	AGRMS	Nexar	FODetect	ציוד רכוב (נת"י)
עלויות לק"מ כביש	✓	✗	✗	✗
איכות ודיוק	✓	✓	✗	✓
יכולת כיסוי בזמן אמת	✓	✗	✓	✗

## מודל עסקי ופוטנציאל רווח

- פוטנציאל השוק טמון ביכולת לחסוך עליות ניטור, להביא לרמת טיוב גבוהה של משטח הנסיעה ולשפר את בטיחות משטח הנסיעה ומתוך כך גם להימנע מתביעות.
- מודל הרווח יהיה שיתוף הלקוח במידע שנאסף.
- אתגר "אוספי המידע";

כעט האתגר הכי גדול של מערכת שלנו הוא אתגר אוספי המידע. והוא כרוך בכך, שאנו צריכים לשכנע מאות עד אלפי נהגים להתקין את האפליקציה שלנו ולהשתמש בה מדי יום בכדי שדטהבייס של התקלות יהיה מעודכן באופן המתוכנן. לפי סקר שערכנו בקרב הנהגים בו לקחו חלק כ-500 משתתפים קיבלנו תוצאה שמעל ל-60 אחוז מהם כואב על הבעיות בכביש בצורה שכואב לנו והם מוכנים לאסוף את המידע הזה ללא שום תמורה. ורק כ-16 אחוז מהמשתתפים לא היו עושים זאת כלל.

- ללא כל תמורה
- תמורה סמלית
- תמורה רציפה
- לא הייתי מתקין



## חלק 2

# תשתיות



## TENSORFLOW



**TENSORFLOW** – היא ספריית הקוד הפתוח שעוזרת לפתח ולהכשיר מודלים של למידה מכונה ולמידה עמוקה. היא בעלת מערכת אקולוגית כוללת וגמישה של הכלים, ספריות ומשאבים קהילתיים המאפשרת לחוקרים לדחוף את החדשנות בתחום ML ומפתחים בונים ופורסים בקלות יישומים המופעלים על ידי ML.

מודל הלמידה העמוקה אנו משתמשים בפרויקט הזה הוא מודל מבוסס CNN (Convolutional Neural Network), שהיא רשת נירונים שמתאימה ביותר לצורכי ניתוח תמונה, בעלת ארכיטקטורה של משקולות משותפות ותכונות של חוסר יכולת לתרגום.

השימוש ב-CNN נעשה בעזרת Keras API שמאפשרת ניסויים של למידה עמוקה בקלות. המודל שנלקח כבסיס הרשת הוא מודל MobileNetV2, שפותח לצורכי שימוש יעיל בסביבת מכשירים ניידים, וכתוצאה מכך בעל גודל המודל יחסית קטן וביצועים טובים על מעבדים חלשים יחסית. מודל בעל עומק של 88 שכבות ו-3.538.984 פרמטרים ניתנים ללמידה.

לצורכי למידה של המודל נאספו כ-5600 תמונות של בורות וסדקים על הכביש וגם קטעי כביש תקינים. אותם תמונות חולקו ל-3 תיקיות למידה train, test and eval. מתוכם 80% זה ללמידה ו-10% לבדיקה ווידוי כל אחת. תהליך למידה התבצע בעזרת טכנולוגיית CUDA – שמאפר למידה על מעבדי GPU – וכך מזרז רבות את התהליך.

## TENSORFLOW

לצורך למידת ה-CNN וקבלת רשת נירונים זריזה, התמונות שאספנו מועברים קודם כל לרזולוציה של 128X128 פיקסלים ואז עוברים תהליך איבוד בו באופן אקראי משנים את התקריב, סיבוב, גובה, רוחב, גזירה והיפוך של התמונה ועם זה אנחנו בונים את ה-Batches בגודל 32 אותם נעביר קדימה ואחורה דרך כל הרשת. (גודל 32 נבחר על ידנו כגודל אופטימלי יחסית לגודל תיקיות הלמידה שהברשותינו).

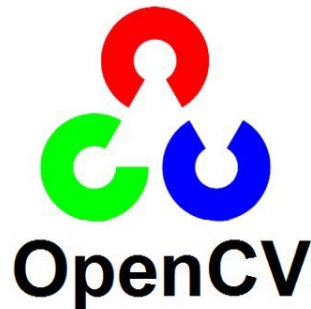
מודל עצמו מבוסס על שני סוגי האובייקטים. ראשון – כל מה שמוגדר כ-"בור". שני – כל מה שמוגדר כ-"כל השער". בורות וסדקים שנאספו נכללו כ-בור, ובכל השער נכנסו חלקי כביש התקינים, חלקי רכבים, קטעים של סימוני הכביש כגון: סימוני נתיב הנסיעה, שוליים, מעברי חציה וכו'...

למידה של המודל התבצעה בצורה של "שבלונה". בו כהתחלה אנחנו קובעים את התחומים של "hyperparameters". ואז מריצים את הלמידה בלולאות על הפרמטרים האלה עד לקבלת מודל בעלת דיוק המרבי. (בנוסף שומרים את הנתונים של פרמטרים ה"טובים" שאותם אפשר לנתח כדי לראות מהי הקונפיגורציה הטובה ביותר למקרה הנתון)

אחרי בחירת המודל המוצלח ביותר אנו ממירים אותו לגרסת "לייט" של TF – גרסה, בה אפשר להשתמש בסביבת פיתוח Android.

כל הניסויים, הרצות ובנייה של המודל נעשו בסביבת עבודה IntelliJ Idea, בספת Python.

## OPENCV



ספריית "OpenCV" היא ספריית ראיית מכונה בעלת קוד פתוח, שפותחה לצורכי עיבוד תמונה ספרתית בשיטת אלגוריתמיקה וגם למידת מכונה. בפרויקט שלנו היא מבצעת חלק העיקרי של ליבת המערכת. לצורכי פיתוח השתמשנו בגרסת הספרייה בספת Python, שתורגמה לספת Java בשלבים של חיבורה לסביבת Android.

דבר ראשון שאנו עושים בעזרת OpenCV זה העתק של התמונה בשחור לבן. על תמונת שחור לבן אנחנו מפעילים אלגוריתם של Canny, כדי למצוא קצבות של כל האובייקטים בהם יש שינוי חד בבהירות ביניהם לסביבה. מיד אחרי זה מבעצים גזירת חלק העליון של התמונה. מה שזה עושה, זה מוציא את חלק של ה"שמיים" מהתמונה כך שלא נצטרך לבזבז כוח עיבוד עליו. בשלב הזה התמונה מוכנה לעבור את האלגוריתם חיפוש נתיבי הנסיעה.

בשביל למצוא נתיב הנסיעה אנחנו קודם כל מפעילים אלגוריתם של HoughLines, אשר מוצא קווים ישרים בתמונה. אז אנחנו מנתחים את הקווים האלה כדי לבחור את שני הקווים: שמאלי וימני שהכי מתאימים להיות הקווים של הנתיב הנסיעה. אחרי שמצאנו את הנתיב אנחנו שוב פעם גוזרים את התמונה כדי להוריד את השער הדברים מיותרים מאזור החיפוש של בור שעכשיו קיבלנו.

## OPENCV

כדי למצוא בורות באזור הנתון אנחנו משתמשים בשיטת ה"סף אדפטיבי".  
ערך הסף מחושב עבור אזורים קטנים יותר ולכן יהיו ערכי סף שונים עבור אזורים שונים.  
בצורה כזאת אנו יכולים למצוא אזורים יוצאי דופן, קווי מתאר, ואת הגבולות שלהם.  
עבור כל קו המתאר אנחנו מחשבים את שטח והיקף של האזור. אנחנו פוסלים את  
האזורים בהם קו המתאר בנוי מכמות הקטנה של הנקודות (זה מרמז לנו על צורה פשוטה  
של האזור, למשל ריבוע, שלא יכולה להיות צורה של בור), ובנוסף גם פוסלים אזורים בעלי  
היקף ושטח קטן או גדול מדי. כל מה שנשאר לאחר פסילות מועבר למודל הלמידה  
העמוקה להכרעה הסופית.

## ANDROID



הפרויקט שלנו הוא בעצם אפליקציית אנדרואיד נועדת לטלפונים ניידים אשר ישבו על מתקן ברכב. גרסת מערכת אנדרואיד עברה פותחה האפליקציה היא 4.4 (Kit Kat), אשר מכסה כ-98% מכשירי אנדרואיד בשוק. במערכת קיימים כמה מסכי משתמש (כגון מסך כניסה למערכת) אשר מבוססים על קבצי class המתאימים וגם קבצי activity.xml הנלווים. קבצי class משמשים עבור ניהול המסך ספציפי וגם מבצעים בתוכם פונקציות איבוד מידע שונות כמו: שליחת וקבלת נתונים מהשרת, קריא וכתובה של הגדרות המערכת שונות וכו'. קבצי xml הם בעצם קבצים מבוססי סימונים אשר מגדירים את היררכיה ועיצוב של אמצעי שליטה של המשתמש בכל מסך תוכנה הספציפי.

בנוסף על קבצים שפיתחנו השתמשנו גם ב-API שונים בשביל להרכיב את הפונקציונליות של האפליקציה. בשביל כניסה למערכת, אנו בנוסף לרישום רגיל בשרת המרוחק, משתמשים ב-API של פייסבוק וגוגל, אשר מזהים את המשתמש הרשום אצלם ומשטפים את השרת שלנו במידע הרלוונטי לצורכי רישום. API נוסף שאנו משתמשים בו הוא: מפות גוגל, כאשר בעזרתו אנחנו מסוגלים להארות את כל המידע שנאסף בשרת על בורות, על מפה אחת.

## PHP



PHP בפרויקט שלנו אחראי על צד השרת. הוא מבצע מגוון דברים, כגון: עלייה והורדה של קבצים אשר משתמשים מעלים במסגרת סריקת הכביש, עברת הנתונים של משתמשים לצורכי זיהוי, וגם ניהול של מוסד הנתונים אשר נמצא באותו שרת.

לצורכי העלאת התמונות שהתקבלו במהלך הסריקה אנו משתשים בהרחבת "volley" אשר מתוך האנדרואיד מסדרת נתונים על בסיס מבנה של Map, ושולחת בעזרת HTTP:POST את הנתונים לשרת, כאשר בשרת מתבצעת בדיקה האם קובץ תמונה תקין ואינו קיים במערכת. אם הבדיקה עברה בהצלחה הקובץ מתווסף לתיקיית התמונות ובנוסף נכנס למוסד נתונים עם מידע של המיקום שהצילום בוצע בו ופרטי המשתמש.

לצורכי הורדת תמונות והצגתם על מפה אנחנו שולפים את הנתונים מדטהבייס ומכניסים אותם למערך אסוציאטיבי שאותו מקודדים בתוך אובייקט JSON. האובייקט הזה אנחנו מקבלים ע"י API של מפות גוגל, שולפים את המידע ומסדרים אותו לפי מיקום על מפה.

## MYSQL



המוסד נתונים MySQL שאנו משתמשים בו הוא מוסד MariaDB אשר יושב בשרת המרוחק שלנו בדומיין של kerron.xyz ומנוהל ע"י PHP שרץ בשרת. במוסד קיימים שני טבלאות ראשיים שהן: טבלת המשתמשים וטבלת התמונות.

בטבלת המשתמשים קיימים שדות הבאים: שם המלא של משתמש, ID הייחודי שלו (אשר מהווה מפתח הראשי של הטבלה), שם המשתמש, אימייל, סיסמה ומין המשתמש. שדות החובה למילוי הן שדות של שם המלא ואימייל, כל השער שדות רשאים להיות ריקים. וזאת כיוון שבנוסף לרישום רגיל אנו משתמשים גם ברישום דרך פייסבוק וגוגל, אשר מוסרים לנו רק אימייל ושם משתמש ללא שום מידע נוסף.

בטבלת התמונות נשמרות תמונות של בורות שעלו לשרת, וקיימים שדות הבאים: ID של התמונה (מפתח ראשי), שם של התמונה, כתובת אינטרנט שלה בשרת, אימייל של משתמש אשר עלה את התמונה, קו רוחב וקו אורך. כל השדות כאן הן שדות חובה מלבד שדה של אימייל של משתמש אשר יכול להיות ריק וזאת כיוון שבמערכת קיימת אופציה לדיווח על בורות באופן אנונימי.

## חלק 3

# סקירת מסכים



3.

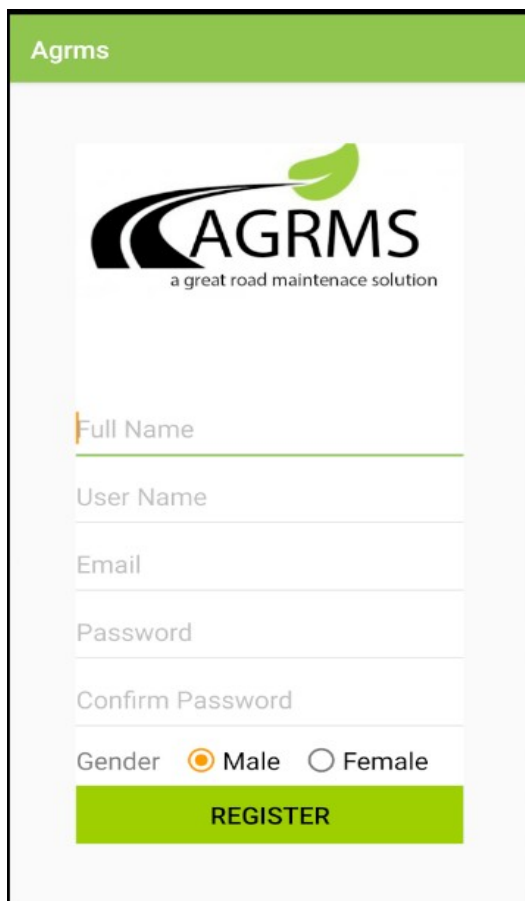
### מסך כניסה למערכת

מסך ראשון שמופיע כאשר מדליקים את האפליקציה. כאן ניתן להירשם בעזרת לחיצה על כפתור Register, או להירשם בעזרת גוגל או פייסבוק על ידי לחיצה על כפתורים המתאימים.

בשביל להיכנס למערכת למשתמש רשום, צריך להזין את אימייל וסיסמה בשדות Email ו Password ואז ללחוץ על כפתור Login. ע"י לחיצה על כפתור Keep Me Logged In, המשתמש יכול להישמר במערכת עד שיחליט לצאת. בנוסף ניתן להיכנס למערכת באופן אנונימי ע"י לחיצה על כפתור Anonymous.

3.

### מסך רישום



כאן ניתן להירשם למערכת ע"י הכנסת נתונים לכל השדות ולחיצה על כפתור Register. כל השדות הן שדות חובה ואם משתמש מפספס משהו או מזין באיזשהו שדה ערך לא תקין - יקבל התראה על כך. השדות הן: שם משתמש מלא, שם משתמש לצורכי כניסה למערכת, אימייל, סיסמה, הזנה נוספת של הסיסמה לוידוי ומין המשתמש.

אורך הסיסמה שניתן להזין הוא בין 6 ל-40 תווים, שמות המשתמש עד 35 תווים. תקינות האימייל מתבצעת ע"י פילטר PHP: FILTER\_VALIDATE\_EMAIL

3.

### מסך הראשי



אחרי שמשתמש נכנס למערכת באופן כלשהו. בפניו מופיע מסך הראשי שבו ניתן לבחור בכמה אפשרויות:

- start road scan – להפעיל את תהליך סריקת הכביש אוטומטי.
- upload photo of road hole – כאן יפתח מסך להעלאת תמונה בודדת של הבור.
- pothole map – לפתוח את מפה של כל הבורות שקיימות במערכת.
- about us – מסך "צור קשר" בו מופיעים נתונים של המפתחים והחברה.
- logout – מסך יציאה מהמערכת. (מעביר למסך הכניסה)

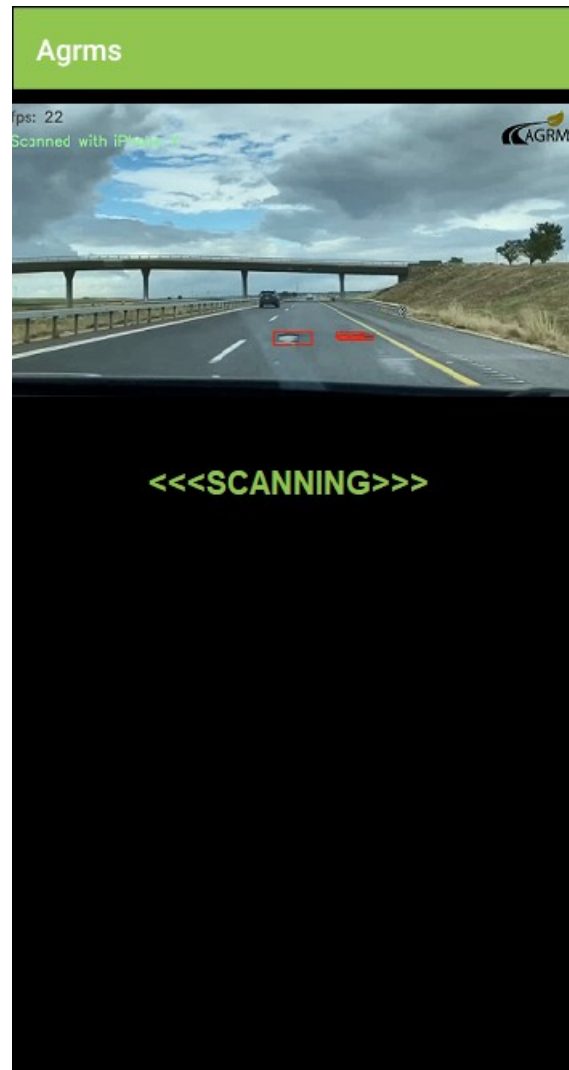
3.

### מסך העלאת תמונה בודדת

1. לוחצים על Select Image, (בפעם ראשונה מערכת תבקש הרשאות למצלמה)
2. מצלמים תמונת בור.
3. לוחצים על Upload captured image on server, (גם כאן בפעם ראשונה המערכת תבקש הרשאות לגישת GPS)
4. המערכת תתעד את המיקום ותשלח תמונה לשרץ.

3.

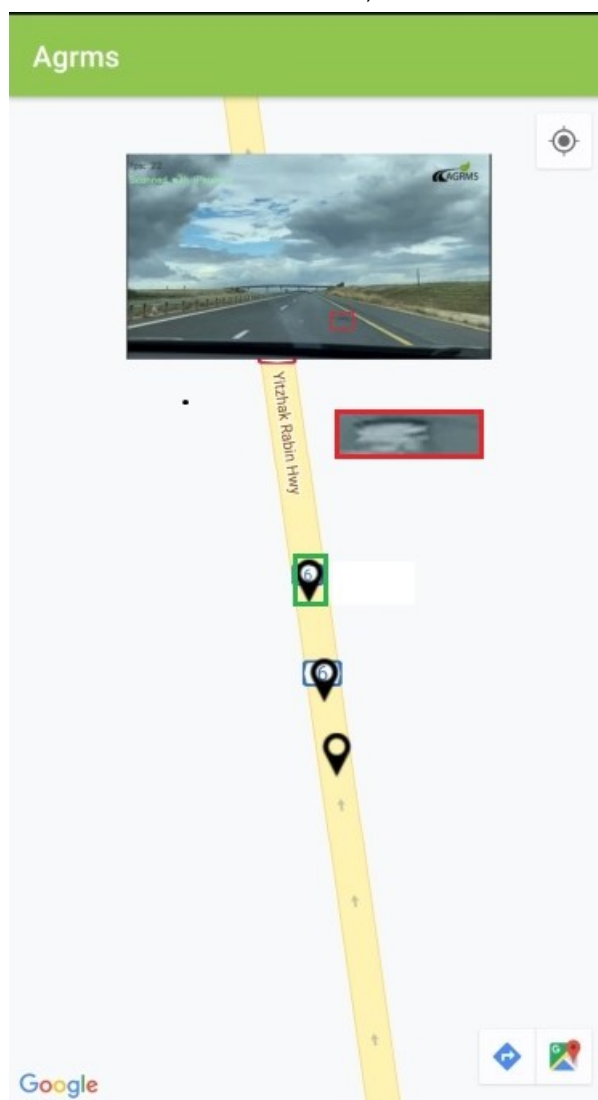
### מסך סריקת הכביש



במסך נ"ל לא ניתן לעשות שום דבר מלבד לצפות בביצוע של סריקה אוטומטית. בהפעלת סריקה בפעם ראשונה – מערכת תבקש הרשאות למצלמה ו-GPS. (בהמשך במסך הזה הולכת להתווסף פונקציה של "minimize" – כדי לאפשר למערכת לעבוד ברקע)

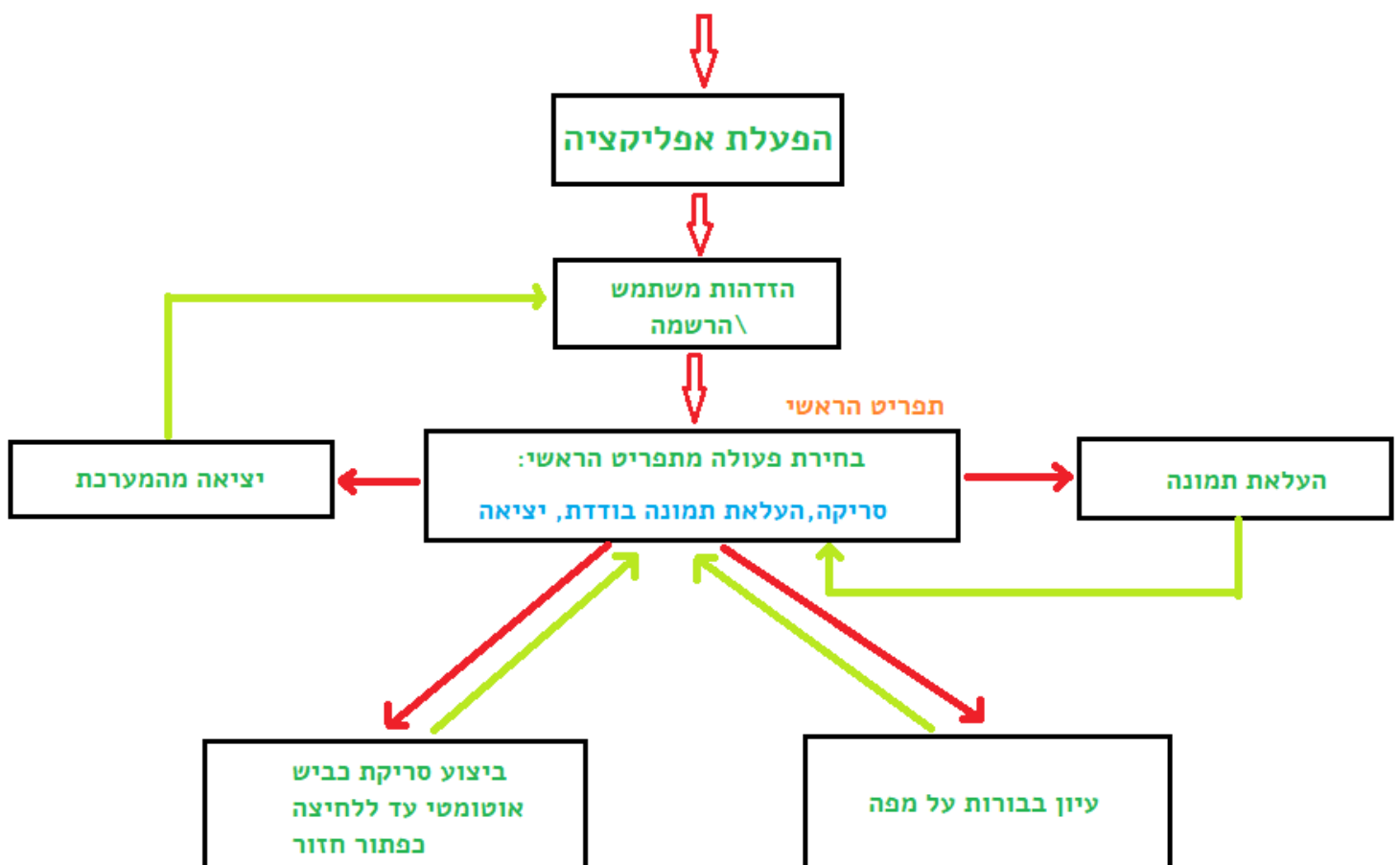
3.

### מסך מפת הבורות



כאן ניתן בעזרת מפות גוגל לראות את כל הבורות שנמצאים במוסד הנתונים. הבורות מסומנים בסימן שחור. כאשר לוחצים עליו התמונה של בור מופיע בחלק העליון של המסך. לחיצה במקום ריק כלשהו על מסך תסתיר את התמונה.

## תרשים פעילות המשתמשים



.4

## ביבליוגרפיה

### **:Tensorflow and Deep Learning**

[/https://www.udemy.com/share/101uGqCUcceFZURH4](https://www.udemy.com/share/101uGqCUcceFZURH4)  
[/https://www.udemy.com/share/101qRiCUcceFZURH4](https://www.udemy.com/share/101qRiCUcceFZURH4)  
<https://towardsdatascience.com/machine-learning/home>  
[/https://machinelearningmastery.com](https://machinelearningmastery.com)  
[/https://keras.io](https://keras.io)  
<https://www.tensorflow.org/overview?hl=en>

### **:OpenCV**

[https://www.youtube.com/playlist?list=PLS1QulWo1RIa7D1O6skqDQ-JZ1GGHKK-](https://www.youtube.com/playlist?list=PLS1QulWo1RIa7D1O6skqDQ-JZ1GGHKK-K)  
[\\_K  
/https://opencv.org](https://opencv.org)

### **:Android**

[/https://www.udemy.com/share/101rnWCUcceFZURH4](https://www.udemy.com/share/101rnWCUcceFZURH4)  
[/https://www.udemy.com/share/101rbkCUcceFZURH4](https://www.udemy.com/share/101rbkCUcceFZURH4)  
<https://developer.android.com>

### **:PHP**

<https://www.w3schools.com/PHP>

### **:MySQL**

<https://www.w3schools.com/sql/default.asp>



## חלק 6

## קוד

## TENSORFLOW

למידה של המודל התבצעה בצורה של "שבלונה". בו כהתחלה אנחנו קובעים את התחומים של "hyperparameters". ואז מריצים את הלמידה בלולאות על הפרמטרים האלה עד לקבלת מודל בעלת דיוק המרבי. (בנוסף שומרים את הנתונים של פרמטרים ה"טובים" שאותם אפשר לנתח כדי לראות מהי הקונפיגורציה הטובה ביותר למקרה הנתון

PotHole Model.py

לוקחים את המודל של מוביילנט.

```
from tensorflow.keras.applications.mobilenet_v2 import MobileNetV2 as Pretrained-
Model, \
    preprocess_input
```

טוענים את הדטהסט. קובעים גודל תמונה של 128 על 128.

```
train_path = 'C:/Users/Roman/Documents/Python/Project/Data/mainSet/train'
test_path = 'C:/Users/Roman/Documents/Python/Project/Data/mainSet/test'
eval_path = 'C:/Users/Roman/Documents/Python/Project/Data/mainSet/eval'
model_path = 'C:/Users/Roman/Documents/Python/Project/Data/models/'
```

```
# Image parameters initialization
IMAGE_SIZE = [128, 128]
batch_size = 1
folders = glob(train_path + '/*[!.txt]')
eval_folders = glob(eval_path + '/*[!.txt]')
class_no = len(folders) # number of classes
class_names_file = pd.read_fwf(eval_path + '/classes.txt', header=0)
class_names = {}
for line in range(0, class_names_file.__len__()):
    class_names[class_names_file.loc[line][0].split('--')[0]] =
str(class_names_file.loc[line][0].split('--')[1])
```

```
train_image_files = glob(train_path + '/*/*')
test_image_files = glob(test_path + '/*/*')
eval_image_files = glob(eval_path + '/*/*')
# train_image_files.sort()
# test_image_files.sort()
```

```
# eval_image_files.sort()
```

מכילים דטהסט לבדיקה הסופית של מודל.

```
# Prepare prediction testing images
prediction_images = []
for c in range(0, class_no):
    image_files = glob(eval_path + '/' + str(c) + '/*')
    class_batch = []
    for e in image_files:
        Img = load_img(e)
        rImg = Img.resize(IMAGE_SIZE)
        Img_array = img_to_array(rImg)
        Img_array = Img_array.reshape((1,) + Img_array.shape) # Converting into 4
dimension array
        class_batch.append(Img_array)
    prediction_images.append(class_batch)
```

כאן מתבצעת התאמה של המודל לצרכים שלנו, מורידים את ה-Dense layer.

```
# Load Pretrained Model with new model parameters
ptm = PretrainedModel(
    input_shape=IMAGE_SIZE + [3],
    weights='imagenet',
    include_top=False,
    classes=class_no
)
```

קביעת גודל של צד מתבצעת בעזרת חלוקה של כלל הדטהסט בגודל הבאטצ'.

```
# Define steps amount
train_steps = int(np.ceil(len(train_image_files) / batch_size))
validation_steps = int(np.ceil(len(test_image_files) / batch_size))
eval_steps = int(np.ceil(len(eval_image_files) / batch_size))
```

הגנרטור של התמונות, מערבב את התמונות של הדטהסט ומשנה את הפרמטרים שלהם באופן אקראי לפי תחומים שקבנו. משתמשים בו לצורכי הסחה של הקלט התמונות (במיוחד שגודל שלו יחסית קטן – דבר גורם להורדות יכולת הלמידה)

```
# Keras image data generator returns classes one-hot encoded
# create an instance of ImageDataGenerator
gen_train = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    zoom_range=0.2,
    horizontal_flip=True,
```

```

        preprocessing_function=preprocess_input
    )

    gen_test = ImageDataGenerator(preprocessing_function=preprocess_input)
    gen_eval = ImageDataGenerator(preprocessing_function=preprocess_input)

    train_generator = gen_train.flow_from_directory(
        train_path,
        shuffle=True,
        target_size=IMAGE_SIZE,
        batch_size=batch_size,
    )

    test_generator = gen_test.flow_from_directory(
        test_path,
        target_size=IMAGE_SIZE,
        batch_size=batch_size,
    )

```

כאן נשתמש בגנרטור שני (שמכין תמונות לבדיקה) – לא נערבב כלום כיוון שאנחנו רוצים דיוק יחסי לסט האמיתי.

```

# create a 2nd train generator which does not use data augmentation
# to get the true train accuracy
eval_generator = gen_eval.flow_from_directory(
    eval_path,
    target_size=IMAGE_SIZE,
    batch_size=batch_size,
)

```

נקבע את רשימת ה-Hyperparameters, שאנו הולכים להשתמש בהם לצורכי למידה.

```

# @off
optimizer_list = [
    tf.keras.optimizers.SGD,          # 0
    tf.keras.optimizers.RMSprop,      # 1
    tf.keras.optimizers.Adagrad,      # 2
    tf.keras.optimizers.Adadelta,     # 3
    tf.keras.optimizers.Adam,         # 4
    tf.keras.optimizers.Adamax,       # 5
    tf.keras.optimizers.Nadam         # 6
]

```

```

optimizer_rate = [
    # 1          # 0
    # 0.1        # 1
    0.01,       # 2
]

```

```

0.001,          # 3
0.0001,         # 4
0.00001,        # 5
0.000001        # 6
]

loss_functions_list = [
    'mean_squared_error',          # 0
    'mean_absolute_error',         # 1
    'mean_absolute_percentage_error', # 2
    'mean_squared_logarithmic_error', # 3
    'squared_hinge',               # 4
    'hinge',                       # 5
    'categorical_hinge',           # 6
    'logcosh',                     # 7
    'huber_loss',                  # 8
    'categorical_crossentropy',     # 9
    'sparse_categorical_crossentropy', # 10
    'binary_crossentropy',          # 11
    'kullback_leibler_divergence',  # 12
    'poisson',                     # 13
    'cosine_proximity',             # 14
    'is_categorical_crossentropy'    # 15
]

```

```

metrics_list = [
    'accuracy',                    # 0
    'binary_accuracy',             # 1
    'categorical_accuracy',        # 2
    'sparse_categorical_accuracy',  # 3
    'top_k_categorical_accuracy',   # 4
    'sparse_top_k_categorical_accuracy', # 5
    'cosine_proximity'             # 6
]

```

## פרמטרים של שבלונה הבסיסית.

```

# train loop routine:          # default:
cut = 0 # ----- 0 - only dense layer cut
layers_to_train = 1 # ----- 1 - only new dense layer trained
epochs = 5 # ----- 5 epochs
loss = 9 # ----- 9 - categorical_crossentropy
opt = 4 # ----- 4 - adam
opt_rate = 3 # ----- 1 - 0.001
metrics = 0 # ----- 0 - accuracy

```

כאן מתחילים להריץ אותה, קובעים בכל לולאה את התחומים של פרמטר ספציפי.

```
Start train routine #
for cut in range(0, 1): # -----
number of top layers cut
for layers_to_train in range(1, 31): # ----- #
number of top layers to train
for epochs in range(6, 15): # ----- number
of epochs
for loss in range(0, len(loss_functions_list)): # ----- #
loss functions
for opt in range(0, len(optimizer_list)): # ----- optimizers
for opt_rate in range(0, len(optimizer_rate)): # ----- optimizer
rates
for j in range(0, 10): # ----- iterations
()model_start_time = time.time
```

כמות השכבות שמורידים ממודל המקורי.

```
x = Flatten()(ptm.layers[-cut].output)
```

מגדירים את השכבה ה"אחרונה" החדשה, משתמשת בהפעלה Softmax, אשר נועדה לחלוקת הסתברויות של רשת בעלת קלאסים רבים.

```
x = Dense(class_no, activation='softmax')(x)
```

כאן מתחיל תהליך הלמידה. מאפסים משקלים על שכבות שכן נבחרו, כל השער - מקפאים אותם. (הרי אנחנו רוצים להשתמש במידע שכבר נלמד על הרשת המקורית ImageNet, לפחות באופן חלקי).

```
# create a model object
model = Model(inputs=ptm.input, outputs=x)

# reset trainable layers
for layer in model.layers:
    layer.trainable = True

# freeze pretrained model weights up to specific value
for layer in model.layers[:-layers_to_train]:
    layer.trainable = False

model.compile(
    loss=loss_functions_list[loss],
```

```
optimizer=optimizer_list[opt](lr=optimizer_rate[opt_rate]),
metrics=(metrics_list[metrics])
)

model.fit(
    train_generator,
    validation_data=test_generator,
    epochs=epochs,
    steps_per_epoch=train_steps,
    validation_steps=validation_steps,
)

eval_val = model.evaluate(
    eval_generator,
    steps=eval_steps
)
```

אם מהבדיקה הפנימית של הלמידה קיבלנו שדרגת האבדן לא עולה על 1, נעשה הרצה של המודל על סט הבדיקה נוסף שהכנו, אשר מדמה את המצב אמת בו אנחנו שואלים את המודל על כל תמונה ותמונה – האם זוהה בור בדיוק שמתחת ל 50%? האם כן – נחשב את זה בתור כישלון, אחרת – הצלחה. בסוף נחשב את האחוז ההצלחות מול הכישלונות ונקבע דיוק המודל המרבי.

```
if eval_val[0] < 1: # if loss is not bigger than 1
    # predict on whole evaluation set
    # iterating class by class and predicting on each image to see if model misses
    prediction_results = []
    for i in range(0, class_no):
        misclassified = 0
        for p in range(0, len(prediction_images[i])):
            predictions = model.predict(prediction_images[i][p])
            if predictions[0][i] < 0.5:
                misclassified += 1
        prediction_results.append(1 - (misclassified / len(prediction_images[i])))

    predict_sum = 0
    for p in prediction_results:
        predict_sum += p
    detection_chance = predict_sum / len(prediction_results)

    model_name = (model_type + '_' + str(IMAGE_SIZE[0]) + 'x' + str(IMAGE_SIZE[1]))
    + '_' +
        str(len(train_image_files)) + '_' + 'Detection_Chance=[' +
        str(detection_chance.__round__(4)) + ']' +
        str(datetime.datetime.now()).replace(':', '_'))
```

ברגע שסיימנו את הלמידה, אנחנו רוצים לשמור את הנתונים של הפרמטרים בהם השתמשנו לתוך הקובץ ביחד עם הדיוק שקיבלנו, כדי שנוכל להסיק מסקנות לגבי השפעת הפרמטרים לטובת הדיוק ע"י קבוצת המודלים שקיבלנו.

```

Prepare parameters to log #
print(eval_val)
+ '__' + hyper_parameters = (model_type
ImageSize=' + str(IMAGE_SIZE[0]) + 'x' + '
+ '__' + str(IMAGE_SIZE[1])
+ '__' + FileNumber=' + str(len(train_image_files))'
+ '__' + Eval=' + str(eval_val)'
+ '__' + BatchSize=' + str(batch_size)'
+ '__' + Epochs=' + str(epochs)'
+ '__' + Optimizer=' + str(type(model.optimizer).__name__)'
+ '__' + Opt.Rate=' + str(optimizer_rate[opt_rate])'
+ '__' + Loss=' + str(loss_functions_list[loss])'
+ '__' + Metrics=' + metrics_list[metrics]'
+ '__' + Cut=' + str(cut)'
(Layers=' + str(layers_to_train)'

if eval_val[0] < 0.2: # if loss is not bigger than 0.2

model_name = (model_type + '_' + str(IMAGE_SIZE[0]) + 'x' +
+ '_' + str(IMAGE_SIZE[1])
+ ']=str(len(train_image_files)) + '_' + 'Loss
+ '_' + str(eval_val[0].__round__(4))
(('_' ,':')str(datetime.datetime.now()).replace

model.save(model_path + model_name)
result_file = open(model_path + model_name + '/HyperParameters.txt',
'w')
:('__')for h in hyper_parameters.split
result_file.write(h + '\n')
()result_file.close
print(hyper_parameters)
print("--- %s seconds ---" % (time.time() - model_start_time))
()tf.keras.backend.clear_session

```



```
(gc.collect
print("--- %s seconds ---" % (time.time() - start_time))
```

לצורך בדיקה נכונות של מודל וביצועים שלה לעומק, אנחנו משתמשים בספריית matplotlib, אשר מאפשרת לנו לבנות גרף ויזואלי של ניתוח המודל.

## PotHole test.py

מתודה לצורכי טענה של התמונות על הגרף. לוקחת את התמונות ומסדרת אותם בשורה, בזוית מסוימת. בנוסף מוסיפה לכל תמונה כותרת (אשר נקבעת לפי תוצאות הרצת המודל במקרה שלנו – לצורך העניין האם בור שבתמונה זוהה כבור או שכ"כל השאר")

```
# Makes plot of images array
def plots(ims, figsize=(12, 6), rows=1, interp=False, titles=None):
    if type(ims[0]) is np.ndarray:
        ims = np.array(ims).astype(np.uint8)
        if (ims.shape[-1] != 3):
            ims = ims.transpose((0, 2, 3, 1))
    f = plt.figure(figsize=figsize)
    cols = len(ims) // rows if len(ims) % 2 == 0 else len(ims) // rows + 1
    for i in range(len(ims)):
        sp = f.add_subplot(rows, cols, i + 1)
        sp.axis('Off')
        if titles is not None:
            sp.set_title(titles[i], fontsize=8, rotation=30)
        plt.imshow(ims[i], interpolation=None if interp else 'none')
```

מתודה לצורכי טענה של "מטריצת בלבול". מסדרת על הגרף מטריצה בצורה של לוח שח, אשר לכל קלאס יהיו שני משבצות – משבצת תוצאות שהמודל חזה, ומשבצת תוצאות אמת של אותם מקרים.

```
# Plot confusion matrix
def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues):
    """
    # This function prints and plots the confusion matrix.
    # Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
```

```

else:
    print('Confusion matrix, without normalization')

print(cm)

plt.imshow(cm, interpolation='nearest', cmap=cmap)
plt.title(title)
plt.colorbar()
tick_marks = np.arange(len(classes))
plt.xticks(tick_marks, classes, rotation=45)
plt.yticks(tick_marks, classes)

fmt = '.2f' if normalize else 'd'
thresh = cm.max() / 2.
for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
    plt.text(j, i, format(cm[i, j], fmt),
             horizontalalignment="center",
             color="white" if cm[i, j] > thresh else "black")

plt.tight_layout()
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()

```

טוענים את המודל, ואז קיימים שני אפשרויות: או שלבדוק מודל על תמונה אחת, או שלבדוק אותה על סט של תמונות (בדרך כלל 64-32). ברגע שמודל תחזיר את התוצאות נכניס אותם לאחת המתודות מלמלה כדי לראות את התוצאות על הגרף.

```

modelName = 'MobileNetV2_224x224_2860_Hole=[0.94736842105263]_Non-
Hole=[0.99122807017544]'
test_path = 'C:/Users/Roman/Documents/Python/Project/Data/PotHole Dataset/models/'
new_model = tf.keras.models.load_model(test_path + modelName)
# Check its architecture
# new_model.summary()

# Single Image Prediction
IMAGE_SIZE = [int(modelName.split('_')[1].split('x')[1]), int(modelName.split('_')[1].split('x')[0])]
Img = load_img('C:/Users/Roman/Documents/Python/Project/Data/PotHole Dataset/eval/2
hole/930.JPG')
rImg = Img.resize(IMAGE_SIZE)
Img_array = img_to_array(rImg)
Img_array = Img_array.reshape((1,) + Img_array.shape) # Converting into 4 dimen-
sion array
print(Img_array.shape)
predictions = new_model.predict(Img_array)
print('predictions shape : ', predictions.shape)
print(predictions[0][0], predictions[0][1])
print(round(predictions[0][0], 2), round(predictions[0][1], 2))

```

```
# Multi Image Prediction
IMAGE_SIZE = [int(modelName.split('_')[2].split('x')[0]), int(modelName.split('_')[2].split('x')[1])]
batch_size = 64
test_path = 'C:/Users/Roman/Documents/Python/Project/Data/PotHole Dataset/eval'
test_batches = ImageDataGenerator().flow_from_directory(
    test_path,
    target_size=IMAGE_SIZE,
    batch_size=batch_size,
    classes=['1 not a hole', '2 hole'],
)

# Predict batch
predictions = new_model.predict(test_batches, steps=1)
# print('predictions shape : ', predictions.shape)
# print('predictions: ', predictions)

# Round the predictions
for i in range(0, len(predictions)):
    predictions[i][0] = round(predictions[i][0], 0)
    predictions[i][1] = round(predictions[i][1], 0)
print("Rounded predictions: ", predictions)

# # Show first batch of images
images, labels = test_batches[0]
labels = labels[:, 0]
test_labels = np.array(len(labels))
for i in range(0, len(labels)):
    if labels[i] == 0:
        test_labels = np.append(test_labels, ['Hole ' + str(i)])
    else:
        test_labels = np.append(test_labels, ['Non-Hole ' + str(i)])
plots(images, titles=test_labels[1:])
plt.show()

# # Show confusion matrix
cm = confusion_matrix(labels, predictions[:, 0])
cm_plot_labels = ['hole', 'not a hole']
plot_confusion_matrix(cm, cm_plot_labels)
plt.show()

# Show some misclassified examples
misclassified_idx = np.where(predictions[:, 0] != labels)[0]
print("Misclassified list: ", misclassified_idx, type(misclassified_idx))
for i in misclassified_idx:
    print(images[i].shape)
    plt.imshow(images[i].astype('uint8'))
    plt.title("True label: %s Predicted: %s" % (labels[i], predictions[:, 0][i]))
    plt.show();
```

## גרסת עבור מודל בינרי (ניסוי)

```
# Show some misclassified examples v2 (binary)
misclassified_idx = np.where(predictions[:, 0] != labels)[0]
if len(misclassified_idx) != 0:
    print("Misclassified numbers: ", misclassified_idx)
    Miss_imgs = np.zeros((len(misclassified_idx), images.shape[1], images.shape[2],
images.shape[3]))
    Miss_labels = np.array(len(misclassified_idx))
    k = 0
    for i in misclassified_idx:
        Miss_imgs[k] = images[i]
        if labels[i] == 0:
            Miss_labels = np.append(Miss_labels, ['Missed Hole ' + str(i)])
        else:
            Miss_labels = np.append(Miss_labels, ['Missed Non-Hole ' + str(i)])
        k += 1
    plots(Miss_imgs, titles=Miss_labels[1:])
    plt.title = 'Wrong Binary Predictions'
    plt.show()
```

## Scan.java

בשביל להפעיל את המודל באנדרואיד אנחנו צריכים קודם כל להמיר אותו לגרסת ה-Lite של TensorFlow. דבר נעשה בעזרת TFLiteConvert.py. בשביל לחסוך בטעינה מדיסק לזיכרון (שזו אופרציה כבדה במיוחד במכשירים ניידים) אנחנו טוענים את המודל באנדרואיד לתוך המבנה נתונים MappedByteBuffer:

```
private MappedByteBuffer loadModelFile() throws IOException {
    AssetFileDescriptor assetFileDescriptor = this.getAssets().openFd("PotHoleDe-
tector_128x128.tflite");
    FileInputStream fileInputStream = new FileInputStream(assetFileDescriptor.get-
FileDescriptor());
    FileChannel fileChannel = fileInputStream.getChannel();
    long startOffset = assetFileDescriptor.getStartOffset();
    long length = assetFileDescriptor.getLength();
    return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, length);
}
```

השימוש במודל נעשה בשיתוף עם קוד של ראיית המכונה, אשר מכין את התמונה למצב בו אפשר לעשות חיזוי ע"י למידה עמוקה. ברגע שגילינו אזור הפוטנציאלי של הבור, אנחנו מעתיקים ממטריצה המייצגת את התמונה כללית את מיקום של בור הפוטנציאלי בתור תת מטריצה, ממירים אותה לגודל והפורמט המתאים למודל שלנו ומבצעים חיזוי. אם אכן נמצא הבור בסיכוי יותר גדול מ 0.5, אנחנו מעבירים אותו למתודה `uploadImage` לצורכי שליחה לשרת.

```
...
image_found = frame_copy.submat(rect);
resize(image_found, image_found, contour_size);
Bitmap bitmap_found = Bitmap.createBitmap(128, 128, Bitmap.Config.ARGB_8888);
Utils.matToBitmap(image_found, bitmap_found);
inputImageBuffer.load(bitmap_found);
interpreter.run(inputImageBuffer.getBuffer(), outputProbability-
Buffer.getBuffer().rewind());
float[] resultArray = outputProbabilityBuffer.getFloatArray();
if (resultArray[1] > 0.5) {
    rectangle(original_frame, new Point(rect.x, rect.y), new Point(rect.x + rec-
t.width, rect.y + rect.height),
        new Scalar(255, 0, 0), 5);
    long currentTime = System.currentTimeMillis();
    if (currentTime - sendTime > 5000) {
        Log.e("Location: ", " Lat/Long" + Arrays.toString(locationFinder.getLoca-
tion()));
        imageUploader.uploadImage(locationFinder.getLocation(), bitmap_found);
        sendTime = System.currentTimeMillis();
    }
}
...
```

## OpenCV

כמו שמתואר בחלק 3, בעזרת ראיית מכונה אנחנו מכינים את התמונה לחיזוי של הבור. לצורך כך, אנו מבצעים כמה פעולות של איבוד תמונה, חלק הניכר מהן מתבצע בקלאס הנ"ל:

### OpenCVUtils.java

מתודה הבאה מקבלת תמונה ומטריצה, המייצגת מיקום הנקודות על התמונה, וחותכת את התמונה לפי הנקודות האלה כך שנוצר שטח חדש שהוא בעצם "אזור העניין", אשר מועבר הלאה להמשך האיבוד. כל האזור התמונה מחוץ לאזור העניין שנוצר נצבע בשחור וכך בעצם קל לגלות אותו בתור שטח מת – שלא צריך איבוד מצד אחד, והתמונה נשארה בגודל המקורי מצד שני.

```
public Mat region_of_interest(Mat img, List<MatOfPoint> vertices) {
    Mat mask = new Mat(img.size(0), img.size(1), CvType.CV_8U, Scalar.all(0));
    Scalar match_mask_color = new Scalar(255);
    Mat masked_image = new Mat();
    fillPoly(mask, vertices, match_mask_color);
    bitwise_and(img, mask, masked_image);
    return masked_image;
}
```

מתודה הבאה לוקחת נקודות מהצורה של {double, double} וממירה אותם לצורה של MatOfPoint אשר ניתן להשתמש בה לכל מני צרכים של OpenCV.

```
public List<MatOfPoint> elementToPoints(double[][] points_to_add) {
    List<Point> points = new ArrayList<>();
    for (int i = 0; i < points_to_add.length; i++) {
        points.add(new Point(points_to_add[i]));
    }
    MatOfPoint mPoints = new MatOfPoint();
    mPoints.fromList(points);
    List<MatOfPoint> finalPoints = new ArrayList<MatOfPoint>();
    finalPoints.add(mPoints);
    return finalPoints; }
}
```

מתודה עיקרית, בה, בינתן אזור תמונה שעבר תהליך איבוד ע"י אלגוריתם של Canny, אנחנו מחפשים נתיבי נסיעה.

תהליך מתבצע בכמה שלבים: (1) בעזרת אלגוריתם Hough Lines אנחנו מחפשים את כל הקווים הישרים באזור הנתון. הקווים מתקבלים מיוצגים ע"י שני נקודות של הקצבות (2) אם אכן מצאנו קווים, אזור כל קו אנחנו בונים נוסחת שיפוע ואז מנסים לגלות האם קו בעל שיפוע מסוים בתחום שמתאים להיות נתיב שמאלי או ימני. (3) ברגע שמצאנו אותם, אנחנו חותכים את השטח בין שני הנתיבים האלה בעזרת מתודות שמעל ומחזירים את אזור החדש במקביל לתמונה עיקרית שעליה אנחנו מציירים את הנתיבים שמצאנו.

```
public Mat[] find_road_area(Mat cropped_frame, Mat original_frame, Mat
grey_copy) {
    Mat[] toReturn = new Mat[2];
    Mat lines = new Mat();
    Mat roi_frame = new Mat(original_frame.size(0), original_frame.size(1), Cv-
Type.CV_8U, Scalar.all(0));
    double x1, x2, y1, y2, m, b, mx, cross_x, cross_y;
    HoughLinesP(cropped_frame, lines, 1, 3.14159265359 / 180, 35, 80, 200);
    double[] line_left = {0, 0, 0, 0, 0, 0};
    double[] line_right = {0, 0, 0, 0, 0, 0};
    double[][] roi_vertices = {{0, 0}, {0, 0}, {0, 0}};
    boolean found_left = false;
    boolean found_right = false;
    if (!lines.empty()) {
        for (int i = 0; i < lines.rows(); i++) {
            x1 = (lines.get(i, 0)[0]);
            y1 = (lines.get(i, 0)[1]);
            x2 = (lines.get(i, 0)[2]);
            y2 = (lines.get(i, 0)[3]);
            m = ((y2 - y1) / (x2 - x1));
            if (0.7 > m && m > 0.3 && !found_right) {
                found_right = true;
                b = y1 - (m * x1);
                line_right[0] = x1;
                line_right[1] = y1;
                line_right[2] = x2 + 10;
                line_right[3] = y2;
                line_right[4] = m;
                line_right[5] = b;
                Line(original_frame, new Point(x1, y1), new Point(x2, y2), new
Scalar(0, 255, 0), 5);
                if (found_left) {
                    break;
                }
            }
            if (-0.7 < m && m < -0.3 && !found_left) {
```

```

        found_left = true;
        b = y1 - (m * x1);
        line_left[0] = x1 + 20;
        line_left[1] = y1;
        line_left[2] = x2;
        line_left[3] = y2;
        line_left[4] = m;
        line_left[5] = b;
        line(original_frame, new Point(x1, y1), new Point(x2, y2), new
Scalar(0, 255, 0), 5);
        if (found_right) {
            break;
        }
    }
}
}
if (found_right && found_left) {
    if (line_left != line_right) {
        mx = line_right[4] - line_left[4];
        if (mx != 0) {
            b = line_left[5] - line_right[5];
            if (b == 0) {
                cross_x = 0;

            } else {
                cross_x = b / mx;
            }
            cross_y = line_right[4] * cross_x + line_right[5];
            roi_vertices[0][0] = (int) line_left[0];
            roi_vertices[0][1] = (int) line_left[1];
            roi_vertices[1][0] = (int) cross_x;
            roi_vertices[1][1] = (int) cross_y;
            roi_vertices[2][0] = (int) line_right[2];
            roi_vertices[2][1] = (int) line_right[3];
            roi_frame = region_of_interest(grey_copy,
elementToPoints(roi_vertices));
        }
    }
} else {
    putText(original_frame, "Lane Not Found",
        new Point(original_frame.size(0) / 3.0, original_frame.-
size(1) / 2.0), FONT_HERSHEY_SIMPLEX,
        1.5, new Scalar(255, 0, 0), 4, LINE_AA);
}
toReturn[0] = roi_frame;
toReturn[1] = original_frame;
return toReturn;
}

}

```



## Scan.java

כאן מתבצעת הכנה של תמונה לקראת למידה העמוקה. אזור הכביש שמועבר לתוך מתודה נסרק בעזרת סף אדפטיבי וכך אנחנו מקבלים קווי המתאר של מקומות הפוטנציאלים של בור (לצורך העניין כל אזור שהוא שונה ממשטח הכביש הרגיל). ברגע שקיבלנו קווי המתאר אנחנו בודקים שהיקף שלהם נע בין 100 ל-300 פיקסלים, אם זה המצב, אנחנו סוגרים את קווי המתאר העומד בתנאי בריבוע בגבולות שלו על צירים X ו-Y. הריבוע הזה אנחנו שומרים בתור מטריצה שאותה נעביר למודל הלמידה.

```
public Mat detect_holes(Mat grey_frame, Mat original_frame, Mat frame_copy) {
    Mat thresh = new Mat();
    Mat image_found = new Mat(contour_size, CvType.CV_8U, Scalar.all(0));
    List<MatOfPoint> contours = new ArrayList<>();
    Mat hierarchy = new Mat();
    double perimeter;
    adaptiveThreshold(grey_frame, thresh, 1, 0, 0, 61, 10);
    findContours(thresh, contours, hierarchy, RETR_TREE, CHAIN_APPROX_NONE);
    Rect rect = null;
    for (int x = 0; x < contours.size(); x++) {
        perimeter = arcLength(new MatOfPoint2f(contours.get(x).toArray()), true);
        Scalar color = new Scalar(0, 0, 255);
        if ((100 < perimeter) && (perimeter < 300)) {
            drawContours(original_frame, contours, x, color, 2, LINE_8, hierarchy,
0, new Point());
            rect = boundingRect(contours.get(x));
        }
    }
    ...
}
```

## Tensorflow

```
...
    putText(original_frame, "Sending...",
            new Point(original_frame.size(0) / 3.0, original_frame.size(1) / 2.0), FONT_HERSHEY_SIMPLEX,
            1.5, new Scalar(0, 255, 0), 4, LINE_AA);
    ...
}
```

מתודה ראשית מנהלת תהליך ראיית מכונה. לוקחים תמונה מהמצלמה של המכשיר בצורת מטריצה, עושים לה עותק והופכים אותו לשחור לבן, אותו נכניס לאלגוריתם של Canny כדי לקבל תמונה, בה יש רק קצבות של כל האובייקטים. אותה נחתוך בחלק העליון בשל להתפטר מהשטח לא רצוי (שמיים וכו'). בתוצאה שקיבלנו נמצא את נתיבי הנסיעה, נחתוך שוב את השטח מחוץ לנתיבים שמצאנו, ועליו נחפש את הבורות.

```
public Mat onCameraFrame(CvCameraViewFrame inputFrame) {
    if (allSet) {
        frame = inputFrame.cvtColor();
        fixed_frame = frame.t();
        Core.flip(frame.t(), fixed_frame, 1);
        resize(fixed_frame, fixed_frame, frame.size());
        cvtColor(fixed_frame, grey_frame, COLOR_RGBA2GRAY, 1);
        frame_copy = fixed_frame.clone();
        grey_copy = grey_frame.clone();
        Canny(grey_frame, canny_image, 100, 150);
        cropped_frame = UtilsM.region_of_interest(canny_image, roi_vertices);
        roi_frame = UtilsM.find_road_area(cropped_frame, fixed_frame, grey_copy);
        result_frame = detect_holes(roi_frame[0], roi_frame[1], frame_copy);
        return result_frame;
    }
    else return inputFrame.cvtColor();
}
```

## Android

### Login.java

קובץ ניהול מסך ההזדהות.

```
public class Login extends AppCompatActivity {
    private MaterialEditText email, password;
    private CheckBox LoginState;
    private SharedPreferences sharedPreferences;
    private static final String EMAIL = "email";

    CallbackManager callbackManager;
    GoogleSignInClient mGoogleSignInClient;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
    }
}
```

אם משתמש הוא אנונימי - פשוט עוברים למסך הראשי.

```
findViewById(R.id.btn_anonymous).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(Login.this, MainActivity.class));
    }
});
```

מאתחלים את כפתור ומודול כניסה בעזרת גוגל:

```
GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestEmail()
    .build();
mGoogleSignInClient = GoogleSignIn.getClient(Login.this, gso);

SignInButton signInButton = findViewById(R.id.btn_google_sign_in);
signInButton.setSize(SignInButton.SIZE_STANDARD);
signInButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
    }
});
```

```
Intent signInIntent = mGoogleSignInClient.getSignInIntent();
startActivityForResult(signInIntent, 89);
}
});
```

כנ"ל לגבי פייסבוק.

```
callbackManager = CallbackManager.Factory.create();
LoginButton loginButton = findViewById(R.id.btn_facebook_sign_in);
loginButton.setPermissions(Arrays.asList(EMAIL));

AccessToken accessToken = AccessToken.getCurrentAccessToken();
boolean isLoggedIn = accessToken != null && !accessToken.isExpired();
if (isLoggedIn) {
    LoginManager.getInstance().logout();
}

loginButton.registerCallback(callbackManager, new FacebookCallback<LoginResult>() {
    @Override
    public void onSuccess(LoginResult loginResult) {

        GraphRequest.newMeRequest(AccessToken.getCurrentAccessToken(), new
        GraphRequest.GraphJSONObjectCallback() {
            @Override
            public void onCompleted(JSONObject object, GraphResponse response) {

                try {
                    facebook_api(object.getString("name"), object.getString("id"));

                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }).executeAsync();
    }

    @Override
    public void onCancel() {
        Toast.makeText(Login.this, "Cancel", Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onError(FacebookException error) {
        Toast.makeText(Login.this, "" + error.getMessage(),
        Toast.LENGTH_SHORT).show();
    }
});
```

```
    }  
});
```

```
sharedPreferences = getSharedPreferences("UserInfo", Context.MODE_PRIVATE);
```

מאתחלים את שדות של הזדהות מול השרת שלנו.

```
email = findViewById(R.id.email);  
password = findViewById(R.id.password);  
LoginState = findViewById(R.id.checkbox);  
findViewById(R.id.btn_login).setOnClickListener(new View.OnClickListener()  
{  
    @Override  
    public void onClick(View v) {  
        String txt_email = email.getText().toString();  
        String txt_password = password.getText().toString();  
  
        if (!validateMailAndPassword(txt_email, txt_password)) {  
            Toast.makeText(Login.this, "Login failed",  
Toast.LENGTH_SHORT).show();  
        } else {  
            login(txt_email, txt_password);  
        }  
    }  
});  
  
findViewById(R.id.btn_register).setOnClickListener(new View.OnClickLis-  
tener() {  
    @Override  
    public void onClick(View v) {  
        startActivity(new Intent(Login.this, Signup.class));  
        finish();  
    }  
});
```

חלק מActivity אשר בודק עם בהגדרות המערכת משתמש כבר מחובר אם כן – מעבירים אותו למסך ראשי בצורה אוטומטית.

```
String loginStatus = sharedPreferences.getString(getResources().getString(R.string.prefLoginState), "");  
if (loginStatus.equals("loggedin")) {  
    startActivity(new Intent(Login.this, MainActivity.class));  
}
```

מתודה בדיקת תקינות תכנית של אימייל המשתמש וסיסמה. אימייל נבדק על פי תבנית מוגדרת. קריטריון של הסיסמה: שדה לא ריק. אם קיימת שגיאה – מדפיסים בשדה הרלוונטי.

```
private boolean validateMailAndPassword(String txt_email, String txt_password)
{
    boolean valid = true;
    if (txt_email.isEmpty() || !Patterns.EMAIL_ADDRESS.matcher(txt_email).matches()) {
        this.email.setError("please enter Valid Email Address");
        valid = false;
    }
    if (txt_password.isEmpty()) {
        this.password.setError("please enter Password");
        valid = false;
    }
    return valid;
}
```

מתודה שמתבצעת כאשר לוחצים LOGIN. לוקחת את הנתונים אשר הוכנסו בשדות אימייל וסיסמה, מבצעת בדיקות הבסיסיות של התקינות (בעזרת מתודות מעלה), ושולחת נתונים לשרת בעזרת PHP.POST. השרת מבצעה בדיקות מעמיקות, משווה נתונים עם מוסד הנתונים ומחזיר תשובה. לפי תשובה מחליטים האם להיכנס למערכת או שלא. אם הזדהות עברה בהצלחה המשתמש נשמר בהגדרות של המכשיר ובכניסה הבא עובר זיהוי אוטומטי.

```
private void login(final String email, final String password) {
    final ProgressDialog progressDialog = new ProgressDialog(Login.this);
    progressDialog.setCancelable(false);
    progressDialog.setIndeterminate(false);
    progressDialog.setTitle("Registering New Account");
    String URL_REGIST = "https://kerron.xyz/htdocs/login.php";
    String request = new String(Request.Method.POST, URL_REGIST,
    new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            if (response.equals("Login success")) {
                progressDialog.dismiss();
                SharedPreferences.Editor editor = sharedPreferences.edit();
                if (LoginState.isChecked()) {
```

```

        editor.putString(getResources().getString(R.string.prefLo-
ginState), "loggedin");
    } else {
        editor.putString(getResources().getString(R.string.prefLo-
ginState), "loggedout");
    }
    editor.apply();

    SharedPreferences preferences = getSharedPreferences("user",
MODE_PRIVATE);

    SharedPreferences.Editor edit = preferences.edit();
    edit.putString("id", ((EditText) findViewById(R.id.email)).get-
Text().toString());
    edit.apply();

    startActivity(new Intent(Login.this, MainActivity.class));

    } else {
        progressDialog.dismiss();
        Toast.makeText(Login.this, response,
Toast.LENGTH_SHORT).show();
    }
}

}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        progressDialog.dismiss();
        Toast.makeText(Login.this, error.toString(),
Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> param = new HashMap<>();
        param.put("email", email);
        param.put("password", password);

        return param;
    }
};

request.setRetryPolicy(new DefaultRetryPolicy(30000, DefaultRetryPolicy.DE-
FAULT_MAX_RETRIES, DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
MySingleton.getInstance(Login.this).addToRequestQueue(request);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

```

```

        if (requestCode != 89) {
            callbackManager.onActivityResult(requestCode, resultCode, data);
        } else {
            Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccount-
FromIntent(data);
            handleSignInResult(task);
        }

        super.onActivityResult(requestCode, resultCode, data);
    }

    private void handleSignInResult(Task<GoogleSignInAccount> completedTask) {
        try {
            GoogleSignInAccount account = completedTask.getResult(ApiException.-
class);

            if (account != null) {
                google_api(account.getDisplayName(), account.getId(), account-
t.getEmail());
            }
        } catch (ApiException e) {
            Log.e("API", "API crashed");
        }
    }
}

```

מתודה דומה לזו של הזדהות הרגילה, הבדל העיקרי רק בזה ששרת שמעבד את הנתונים הוא שרת של פייסבוק. אשר מבצע הזדהות של משתמש קיים אצלו ואם זה מצליח מעביר לנו את האימייל של המשתמש – אותו אנחנו רושמים במוסד הנתונים. (בשביל לקשר עם התמונות שמשתמש פייסבוק מעלה).

```

void facebook_api(final String name, final String id) {

    final ProgressDialog dialog = new ProgressDialog(Login.this);
    dialog.setMessage("Loading...");
    dialog.show();

    StringRequest stringRequest = new StringRequest(Request.Method.POST,
"https://kerron.xyz/htdocs/facebook.php", new Response.Listener<String>() {
        @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
        @Override
        public void onResponse(String result) {
            Toast.makeText(Login.this, "" + result, Toast.LENGTH_SHORT).show();
            Log.d("result", result);
            dialog.dismiss();
            if (result.equals("0")) {
                Toast.makeText(Login.this, "Some error occur",
Toast.LENGTH_SHORT).show();
            }
        }
    })
}

```



```

    } else {
        dialog.dismiss();

        SharedPreferences.Editor editor = sharedPreferences.edit();
        if (LoginState.isChecked()) {
            editor.putString(getResources().getString(R.string.prefLo-
ginState), "loggedin");
        } else {
            editor.putString(getResources().getString(R.string.prefLo-
ginState), "loggedout");
        }
        editor.apply();

        SharedPreferences preferences = getSharedPreferences("user",
MODE_PRIVATE);

        SharedPreferences.Editor edit = preferences.edit();
        edit.putString("id", result);
        edit.apply();
        startActivity(new Intent(Login.this, MainActivity.class));
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        dialog.dismiss();
        Toast.makeText(Login.this, "" + error.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<String, String>();
        params.put("id", id);
        params.put("name", name);
        return params;
    }
};

stringRequest.setRetryPolicy(new RetryPolicy() {
    @Override
    public int getCurrentTimeout() {
        return 30000;
    }

    @Override
    public int getCurrentRetryCount() {
        return 30000;
    }

    @Override

```

```

        public void retry(VolleyError volleyError) throws VolleyError {
            }
        });
        MySingleton.getInstance(Login.this).addToRequestQueue(stringRequest);
    }

```

מתודה דומה של גוגל.

```

void google_api(final String name, final String id, final String email) {
    final ProgressDialog dialog = new ProgressDialog(Login.this);
    dialog.setMessage("Loading...");
    dialog.show();

    StringRequest stringRequest = new StringRequest(Request.Method.POST,
        "https://kerron.xyz/htdocs/googleLogin.php", new Response.Listener<String>() {
            @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
            @Override
            public void onResponse(String result) {
                Toast.makeText(Login.this, "" + result, Toast.LENGTH_SHORT).show();
                Log.d("result", result);
                dialog.dismiss();
                if (result.equals("0")) {
                    Toast.makeText(Login.this, "Some error occur",
                        Toast.LENGTH_SHORT).show();
                } else {

                    SharedPreferences.Editor editor = sharedPreferences.edit();
                    if (LoginState.isChecked()) {
                        editor.putString(getResources().getString(R.string.prefLo-
                            ginState), "loggedin");
                    } else {
                        editor.putString(getResources().getString(R.string.prefLo-
                            ginState), "loggedout");
                    }
                    editor.apply();

                    SharedPreferences preferences = getSharedPreferences("user",
                        MODE_PRIVATE);

                    SharedPreferences.Editor edit = preferences.edit();
                    edit.putString("id", result);
                    edit.apply();
                    startActivity(new Intent(Login.this, MainActivity.class));
                }
            }
        }, new Response.ErrorListener() {
            @Override

```

```

    public void onErrorResponse(VolleyError error) {

        dialog.dismiss();
        Toast.makeText(Login.this, "Error: " + error.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<String, String>();
        params.put("id", id);
        params.put("name", name);
        params.put("email", email);
        return params;
    }
};

stringRequest.setRetryPolicy(new RetryPolicy() {
    @Override
    public int getCurrentTimeout() {
        return 30000;
    }

    @Override
    public int getCurrentRetryCount() {
        return 30000;
    }

    @Override
    public void retry(VolleyError volleyError) throws VolleyError {

    }
});
MySingleton.getInstance(Login.this).addToRequestQueue(stringRequest);
}
}

```

## SignUp.java

קלאס רישום למערכת.

```
public class Signup extends AppCompatActivity {
    private MaterialEditText fullname, username, email, password, c_password;
    private RadioGroup radioGroupGender;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_signup);

        fullname = findViewById(R.id.fullname);
        username = findViewById(R.id.username);
        email = findViewById(R.id.email);
        password = findViewById(R.id.password);
        c_password = findViewById(R.id.confirmPassword);
        radioGroupGender = findViewById(R.id.radioGender);
        findViewById(R.id.btn_register).setOnClickListener(new View.OnClickListener() {
            {
```

ברגע שמשתמש לחץ על הכפתור רישום מבצעים בדיקת תקינות הקלט בעזרת מתודה validate אם התוצאה תקינה קוראים למתודה register לביצוע רישום מול שרת.

```
        @Override
        public void onClick(View v) {
            String txt_fullname = fullname.getText().toString();
            String txt_username = username.getText().toString();
            String txt_email = email.getText().toString();
            String txt_password = password.getText().toString();
            String txt_c_password = c_password.getText().toString();
            String txt_gender = ((RadioButton) findViewById(R.id.radioGroupGender).getCheckedRadioButtonId()).getText().toString();

            if (!validate(txt_fullname, txt_username, txt_email, txt_password, txt_c_password)) {
                Toast.makeText(Signup.this, "Sign up failed", Toast.LENGTH_SHORT).show();
            } else {
                RegisterNewAccount(txt_fullname, txt_username, txt_email, txt_password, txt_gender);
            }
        }
    });
```

מתודה בדיקת תקינות הקלט. אם משהו לא עבר – מעדכנים בהודעת שגיאה באותו שדה.

```
private boolean validate(String txt_fullname, String txt_username, String
txt_email, String txt_password, String txt_c_password) {
    boolean valid = true;

    if (txt_fullname.isEmpty() || txt_fullname.length() > 35) {
        this.fullname.setError("please enter valid full name");
        valid = false;
    }
    if (txt_username.isEmpty()) {
        this.fullname.setError("please enter valid user name");
        valid = false;
    }
    if (txt_email.isEmpty() || !Patterns.EMAIL_ADDRESS.matcher(txt_email).matches()) {
        this.email.setError("please enter valid Email Address");
        valid = false;
    }
    if (txt_password.isEmpty()) {
        this.password.setError("please enter password");
        valid = false;
    }
    if (txt_c_password.isEmpty()) {
        this.c_password.setError("please confirm password");
        valid = false;
    }
    if (!txt_password.equals(txt_c_password)) {
        this.c_password.setError("passwords are not matched");
        valid = false;
    }

    return valid;
}
```

מתודת רישום מול שרת, שולחים נתונים שהוזנו לשרת בעזרת PHP.POST ומקבלים תשובה. אם הרישום הסתיים בהצלחה - מייצגים הודעה בהתאם אם לא - מייצגים הודעת שגיאה עם פרטים.

```
private void RegisterNewAccount(final String fullname, final String username,
final String email, final String password, final String gender) {
    final ProgressDialog progressDialog = new ProgressDialog(Signup.this);
    progressDialog.setCancelable(false);
    progressDialog.setIndeterminate(false);
    progressDialog.setTitle("Registering New Account");
    String URL_REGISTER = "https://kerron.xyz/htdocs/register.php";
    StringRequest request = new StringRequest(Request.Method.POST, URL_REGIS-
```

```

TER, new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        if (response.equals("Successfully Registered")) {
            progressDialog.dismiss();
            Toast.makeText(Signup.this, response,
                Toast.LENGTH_SHORT).show();
            startActivity(new Intent(Signup.this, Login.class));
            finish();
        } else {
            progressDialog.dismiss();
            Toast.makeText(Signup.this, response,
                Toast.LENGTH_SHORT).show();
        }
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        progressDialog.dismiss();
        Toast.makeText(Signup.this, error.toString(),
            Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> param = new HashMap<>();
        param.put("fullname", fullname);
        param.put("username", username);
        param.put("email", email);
        param.put("password", password);
        param.put("gender", gender);

        return param;
    }
};
request.setRetryPolicy(new DefaultRetryPolicy(30000, DefaultRetryPolicy.DE-
FAULT_MAX_RETRIES, DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
MySingleton.getInstance(Signup.this).addToRequestQueue(request);
}
}

```

## MainActivity.java

קובץ מסך הראשי. בסך הכל טוען את התפריט הראשי ובלחיצת כפתור הבחירה מעביר את המשתמש למסך הרצוי.

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        final SharedPreferences sharedPreferences = getSharedPreferences("User-
Info", MODE_PRIVATE);

        if (!OpenCVLoader.initDebug())
            Log.e("OpenCv", "Unable to load OpenCV");
        else
            Log.d("OpenCv", "OpenCV loaded");

        setContentView(R.layout.activity_main);

        findViewById(R.id.btn_roadActivity).setOnClickListener(new View.OnClickLis-
tener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, Scan.class));
            }
        });

        findViewById(R.id.btn_upload_image).setOnClickListener(new View.OnClickLis-
tener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, UserUploadImage.-
class));
            }
        });

        findViewById(R.id.btn_hole_locations).setOnClickListener(new View.OnClick-
Listener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, Maps.class));
            }
        });

        findViewById(R.id.btn_logout).setOnClickListener(new View.OnClickListener()
{
            @Override
```

```

        public void onClick(View v) {
            SharedPreferences.Editor editor = sharedPreferences.edit();
            editor.putString(getResources().getString(R.string.prefLogin-
State),"loggedout");
            editor.apply();
            LoginManager.getInstance().logout();
            startActivity(new Intent(MainActivity.this, Login.class));
            finish();
        }
    });
}

```

## Maps.java

קובץ ניהול מסך המפות.

```

public class Maps extends AppCompatActivity implements OnMapReadyCallback {

    GoogleMap googleMap;
    ImageView imageView;
    private Marker marker;
    private View view;
    public String url;
    public String[] imageUrl;

    //for the current location on map
    LocationManager locationManager;
    public String latitude, longitude;
    private static final int REQUEST_LOCATION = 1;

```

טוענים משל של מפות גוגל. לקראת סוף התהליך מפה תעבור לאזור הנוכחי בו נמצא המשתמש.

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_maps);

    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
        .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);
    getCurrentLocation();

```



}

מתודה ראשית לניהול המפה. ברגע שפותחים את המסך המפות מתודה שולחת בקשה לשרת לצורך טעינת תמונות של בורות ומיקומם על המפה. אם בורות לא נמצאו – תוצג הודעה מתאימה, אחרת השרת ישלח נתונים של התמונות: מיקום וכתובת תמונה באינטרנט, אשר הולכים להיטען על המפה ויוצגו בעזרת סמל שחור. בלחיצה על הסמל תיפתח תמונה של הבור, בלחיצה בכל מקום פנוי – תיסגר.

```
void google_api() {
    final ProgressDialog dialog = new ProgressDialog(Maps.this);
    dialog.setMessage("Loading...");
    dialog.show();

    StringRequest stringRequest = new StringRequest(Request.Method.POST,
        "https://kerron.xyz/htdocs/get-my-locations.php", new Response.Listener<String>() {
        @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
        @Override
        public void onResponse(String result) {
            dialog.dismiss();
            if (result.equals("0")) {
                Toast.makeText(Maps.this, "No Record found",
                    Toast.LENGTH_SHORT).show();
                return;
            }
            try {
                JSONArray array = new JSONArray(result);
                imageUrl = new String[array.length()];
                for (int i = 0; i < array.length(); i++) {
                    MarkerOptions markerOptions = new MarkerOptions();
                    JSONObject object = array.getJSONObject(i);
                    imageUrl[i] = object.getString("image_path");
                    //url image from json object, to show on each mark his spe-
                    //url = object.getString("image_path");
                    Log.i("ImageUrl[i]:", "ImageUrl[i]: " + imageUrl[i]);
                    markerOptions.position(new LatLng(Double.parseDouble(object.getString("lat")), Double.parseDouble(object.getString("lon"))));
                    googleMap.addMarker(markerOptions.title("details about this marker:").snippet("more details!!!!!!")
                        .icon(BitmapDescriptorFactory.fromResource(R.drawable.blackmarker30x30))
                        .snippet(imageUrl[i]));

                    googleMap.setInfoWindowAdapter(new GoogleMap.InfoWin-
dowAdapter() {
                        @Override
                        public View getInfoWindow(Marker marker) {
                            view = getLayoutInflater().inflate(R.lay-
out.custom_marker_layout, null);
```

```

        Maps.this.marker = marker;
        ImageView image = (ImageView)
view.findViewById(R.id.imageViewRoad);
        Picasso.with(getApplicationContext())
                .load(marker.getSnippet())
                .error(R.mipmap.ic_launcher)
                .into(image);
        return view;
    }

    @Override
    public View getInfoContents(Marker marker) {
        if (Maps.this.marker != null
            && Maps.this.marker.isInfoWindowShown()) {
            Maps.this.marker.hideInfoWindow();
            Maps.this.marker.showInfoWindow();
        }
        return null;
    }
});
}
//for loop end
//move first to current location
googleMap.setMyLocationEnabled(true);//blue point off current
location

googleMap.animateCamera(
    CameraUpdateFactory.newLatLngZoom(new LatLng(Double.-
parseDouble(latitude), Double.parseDouble(longitude)), 14.0f));
    Toast.makeText(Maps.this, "current Location For Testing:" + "\
latitude " + latitude + " \nlongitude " + longitude, Toast.LENGTH_SHORT).show();
    Log.i("LAT:", "latyacov:" + latitude);
    Log.i("LON:", "lonyacov:" + longitude);

    //}
} catch (JSONException e) {
    e.printStackTrace();
}
}
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {

        dialog.dismiss();
        Toast.makeText(Maps.this, "" + error.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<String, String>();

        params.put("email", getSharedPreferences("user", MODE_PRIVATE).get-

```

```
String("id", ""));
        return params;
    }
};

stringRequest.setRetryPolicy(new RetryPolicy() {
    @Override
    public int getCurrentTimeout() {
        return 30000;
    }

    @Override
    public int getCurrentRetryCount() {
        return 30000;
    }

    @Override
    public void retry(VolleyError volleyError) throws VolleyError {
    }
});
MySingleton.getInstance(Maps.this).addToRequestQueue(stringRequest);
}

@Override
public void onMapReady(GoogleMap googleMap) {
    if (googleMap != null) {
        this.googleMap = googleMap;
        google_api();
    }
}
```

מתודת טעינת מפה במיקום של המשתמש.

```
//this function help us to open the map on the current location of the user.
private void getCurrentLocation() {
    ActivityCompat.requestPermissions(this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, REQUEST_LO-
LOCATION);
    locationManager = (LocationManager) getSystemService(Context.LOCATION_SER-
VICE);
    if (!locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER)) {
        OnGPS();
    } else {
        getLocation();
    }
}
```

מתודת טעינת מיקום בעזרת GPS – אם שירות המיקום בעזרת GPS מופעל ע"י משתמש.  
אם לא קיימים הרשאות גישה לשירות, מתודה תבקש זאת.

```
private void OnGPS() {
    final AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setMessage("Enable GPS").setCancelable(false).setPositiveButton("Yes", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            startActivity(new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS));
        }
    }).setNegativeButton("No", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });
    final AlertDialog alertDialog = builder.create();
    alertDialog.show();
}
```

מתודת טעינת מיקום בעזרת אינטרנט סלולרי – אם שירות המיקום מופעל ע"י משתמש.  
אם לא קיימים הרשאות גישה לשירות, מתודה תבקש זאת.

```
private void getLocation() {
    if (ActivityCompat.checkSelfPermission(
        Maps.this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(
        Maps.this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, REQUEST_LOCATION);
    } else {
        Location locationGPS = locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        if (locationGPS != null) {
            double lat = locationGPS.getLatitude();
            double longi = locationGPS.getLongitude();
            latitude = String.valueOf(lat);
            longitude = String.valueOf(longi);
        } else {
            Toast.makeText(this, "Unable to find location.",
                Toast.LENGTH_SHORT).show();
        }
    }
}
```

## RequestHandler.java

קלאס אחראי על שליחת וקבלת נתונים מהשרת.

```
public class RequestHandler {
```

מתודה ששולחת בקשה לשרת בשיטת PHP.GET, מתמשת ב-StringBuider כדי להוציא את מידע המוחזר מ-BufferedReader. ובעצם מחזירה לנו מחרוזת.

```
    public String sendGetRequest(String uri) {
        try {
            URL url = new URL(uri);
            HttpURLConnection con = (HttpURLConnection) url.openConnection();
            BufferedReader bufferedReader = new BufferedReader(new InputStrea-
Reader(con.getInputStream()));

            String result;

            StringBuilder sb = new StringBuilder();

            while((result = bufferedReader.readLine())!=null){
                sb.append(result);
            }

            return sb.toString();
        } catch (Exception e) {
            return null;
        }
    }
}
```

מתודה ששולחת בקשה לשרת בשיטת PHP.POST. וזאת מתודה העיקרית לשליחת נתונים אל השרת. מקבלת כתובת ו-HashMap של מחרוזת מול מחרוזת ומכניסה אותם ל-BufferedWriter. מקבלת תשובה בדומה לשיטת GET למעלה ומחזירה מחרוזת.

```
public String sendPostRequest(String requestURL,
                              HashMap<String, String> postDataParams) {

    URL url;
    String response = "";
    try {
        url = new URL(requestURL);

        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setReadTimeout(15000);
        conn.setConnectTimeout(15000);
        conn.setRequestMethod("POST");
        conn.setDoInput(true);
        conn.setDoOutput(true);

        OutputStream os = conn.getOutputStream();
        BufferedWriter writer = new BufferedWriter(
            new OutputStreamWriter(os, "UTF-8"));
        writer.write(getPostDataString(postDataParams));

        writer.flush();
        writer.close();
        os.close();
        int responseCode = conn.getResponseCode();

        if (responseCode == HttpURLConnection.HTTP_OK) {
            BufferedReader br = new BufferedReader(new InputStreamReader(con-
n.getInputStream()));
            response = br.readLine();
        } else {
            response = "Error Registering";
        }
    } catch (Exception e) {
        e.printStackTrace();
    }

    return response;
}
```

מתודה לקבלת הנתונים מהשרת. דומה לשיטות קודמות. מחזירה מחרוזת.

```
private String getPostDataString(HashMap<String, String> params) throws Unsup-
portedEncodingException {
    StringBuilder result = new StringBuilder();
    boolean first = true;
    for (Map.Entry<String, String> entry : params.entrySet()) {
        if (first)
            first = false;
        else
            result.append("&");

        result.append(URLEncoder.encode(entry.getKey(), "UTF-8"));
        result.append("=");
        result.append(URLEncoder.encode(entry.getValue(), "UTF-8"));
    }

    return result.toString();
}
```

## ImageUploader.java

תפקידו העיקרי של הקלאס נ"ל הוא להאמיר את התמונה לפורמט שניתן לשלוח לשרת, בצירוף של נתוני מיקום ופרטי משתמש.

```
public class ImageUploader extends AppCompatActivity {

    private final Context context;
    private final String ImageUploadPathOnServer = "https://kerron.xyz/htdocs/up-
load.php";
    private Bitmap selected_bitmap = null;
    RequestQueue requestQueue;

    public ImageUploader(Context context) {
        this.context = context;
    }
}
```

מתודה להמרת תמונה בפורמט Bitmap לפורמט PNG, דחיסתה ואז להמרתה למערך של ביטים.

```
public byte[] getFileDataFromDrawable(Bitmap bitmap) {
    ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
    bitmap.compress(Bitmap.CompressFormat.PNG, 100, byteArrayOutputStream);
    return byteArrayOutputStream.toByteArray();
}
```

מתודת העלאת תמונה. מבצעת פעולות מתוארות מעלה בסדר הבא: קודם כל קובעת מיקום הנוכחי של המכשיר – אם שירותי מיקום לא מופעלים – נשלחת התראה על כך. לאחר מכן נקבע זמן המערכת הנוכחי, אחריו נשלפים נתוני המשתמש ובסוף התמונה עוברת איבוד ע"י מתודה `getFileDataFromDrawable`. כל אלה מוכנסים למבנה של HashMap ונשלחים לשרת.

```
public void uploadImage(final double[] loc, Bitmap bitmap) {
    final double lat = loc[0];
    final double lon = loc[1];
    setSelected_bitmap(bitmap);
    if (lat == 0.0 || lon == 0.0) {
        runToast("Please enable location to upload image");
        return;
    }
}
```

```
VolleyMultipartRequest volleyMultipartRequest = new
VolleyMultipartRequest(Request.Method.POST, ImageUploadPathOnServer,
    new Response.Listener<NetworkResponse>() {
        @Override
        public void onResponse(final NetworkResponse response) {
            if (new String(response.data).equals("1")) {
                runToast("Image has been uploaded");
            }
            requestQueue.getCache().clear();
        }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            runToast("Error: " + error.getMessage());
        }
    }) {

    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("email", context.getSharedPreferences("user", Context.-
MODE_PRIVATE).getString("id", ""));
        params.put("lat", lat + "");
    }
}
```



```

        params.put("lon", lon + "");
        return params;
    }

    @RequiresApi(api = Build.VERSION_CODES.N)
    @Override
    protected Map<String, DataPart> getByteData() {
        Map<String, DataPart> params = new HashMap<>();
        long current_time_in_millis = System.currentTimeMillis();
        String current_time = String.valueOf(current_time_in_millis);
        params.put("image", new
VolleyMultipartRequest.DataPart(current_time + "." + (new Random().next-
tInt(1000000))
        + ".png", getFileDataFromDrawable(selected_bitmap)));
        return params;
    }
};

volleyMultipartRequest.setRetryPolicy(new DefaultRetryPolicy(
    1000,
    DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
    DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
requestQueue = Volley.newRequestQueue(context);
requestQueue.add(volleyMultipartRequest);
}

private void setSelected_bitmap(Bitmap selected_bitmap) {
    this.selected_bitmap = selected_bitmap;
}

```

מתודה ששולחת התראות למשתמש מחוץ לתחום של Activity הנוכחי.

```

private void runToast(final String message) {
    new Thread() {
        public void run() {
            runOnUiThread(new Runnable() {
                public void run() {
                    Toast.makeText(context, message,
Toast.LENGTH_SHORT).show();
                }
            });
        }
    }.start();
}
}

```

## LocationFinder.java

קלאס אחראי על שליפת נתוני המיקום של המכשיר.

```
public class LocationFinder extends AppCompatActivity implements LocationListener {

    private double latitude;
    private double longitude;
    Context context;
    LocationManager manager;

    public LocationFinder(Context context) {
        this.context = context;
    }
}
```

מתודה המתעדכנת אוטומטית (ע"י מ"ה) ברגע שמיקום משתנה. כך אנחנו יכולים לשלוח נתונים מהמשתנים ברגע שאנחנו צריכים אותם.

```
@Override
public void onLocationChanged(Location location) {
    latitude = location.getLatitude();
    longitude = location.getLongitude();
}
```

מתודת שליפת המיקום. קוראת למתודת `LocationMethod`, שמעדכנת משתני המיקום ומחזירה אותם.

```
@RequiresApi(api = Build.VERSION_CODES.M)
public double[] getLocation() {
    LocationMethod();
    double[] toReturn = new double[2];
    toReturn[0] = latitude;
    toReturn[1] = longitude;
    return toReturn;
}
```

מתודה בודקת האם שירותי מיקום מופעלים וקיימת גישה לנתוני מיקום. אם כן – קוראת למתודת שליפה בפועל. אם לא, מבקשת ממשתמש הרשאות\ הדלקת שירותי מיקום.

```
@RequiresApi(api = Build.VERSION_CODES.M)
private void LocationMethod() {
    LocationManager lm = (LocationManager) context.getSystemService(Context.LO-
LOCATION_SERVICE);
    boolean gps_enabled = false;
    boolean network_enabled = false;

    try {
        gps_enabled = lm.isProviderEnabled(LocationManager.GPS_PROVIDER);
    } catch (Exception ignored) {
    }

    try {
        network_enabled = lm.isProviderEnabled(LocationManag-
er.NETWORK_PROVIDER);
    } catch (Exception ignored) {
    }

    if (!gps_enabled && !network_enabled) {
        new AlertDialog.Builder(context)
            .setMessage("Location is not enabled. Please enable location to
continue.")
            .setPositiveButton("Open Setting", new DialogInterface.OnClick-
Listener() {
                @Override
                public void onClick(DialogInterface paramDialogInterface,
int paramInt) {
                    startActivity(new Intent(Set-
tings.ACTION_LOCATION_SOURCE_SETTINGS));
                }
            }).setNegativeButton("Cancel", null)
            .show();
    } else {
        get_current_location();
    }
}
```

מתודה שולפת נתונים מהחיישנים של המכשיר.

```
@RequiresApi(api = Build.VERSION_CODES.M)
private void get_current_location() {
    manager = (LocationManager) context.getSystemService(LOCATION_SERVICE);

    if (ActivityCompat.checkSelfPermission(context, Manifest.permis-
sion.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED && Activity-
```

```

Compat.checkSelfPermission(context, Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
    runToast("Please allow access to you location");
} else {
    FusedLocationProviderClient fusedLocationClient;
    fusedLocationClient = LocationServices.getFusedLocationProvider-
Client(context);
    fusedLocationClient.getLastLocation()
        .addOnSuccessListener((Activity) context, new OnSuccessLis-
tener<Location>() {

        @Override
        public void onSuccess(Location location) {
            if (location != null) {
                latitude = location.getLatitude();
                longitude = location.getLongitude();
            } else {
                Criteria crit = new Criteria();
                crit.setAccuracy(Criteria.ACCURACY_FINE);
                if (ActivityCompat.checkSelfPermission(context,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManag-
er.PERMISSION_GRANTED) {

                    return;
                }
                manager.requestLocationUpdates(manager.getBest-
Provider(crit, false), 1000, 1, (android.location.LocationListener) context);
            }
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            runToast("Error Occured while getting current location");
        }
    });
}
}
}
}

```

## PHP and SQL

### Facebook.php

קוד התחברות דרך פייסבוק - מקבל אימייל משתמש מצד הלקוח. מתחבר למוסד הנתונים, בודק האם אימייל קיים - אם לא מוסיף למוסד משתמש חדש.

```
<?php
// Create connection
$conn = mysqli_connect("78.140.191.36", "kerronxy_yacov", "]k2oI1?WBRPh",
"kerronxy_users");
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT * FROM `users_table` WHERE email LIKE '" . $_REQUEST['id'] . "'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    echo $_REQUEST['id'];
} else {
    $sql = "INSERT INTO `users_table` (`fullname`, `username`, `email`, `password`,
`gender`) VALUES ('" . $_REQUEST['name'] . "', '" . $_REQUEST['id'] . "', '" . $_REQUEST['id'] . "', '" . $_REQUEST['id'] . "', '" . $_REQUEST['id'] . "')";

    if ($conn->query($sql) === TRUE) {
        echo $_REQUEST['id'];
    } else {
        echo $conn->error;
    }
}
$conn->close();

?>
```

## Get-my-Locations.php

קוד השולף נתוני תמונות מהמוסד. מבצע התחברות ואז שליפה של מיקום וכתובת התמונה באינטרנט מהמוסד. מכניס את הנתונים לאובייקט JSON ושולח ללקוח.

```
<?php

$conn = mysqli_connect("78.140.191.36","kerronxy_yacov", "k2oIl?WBRPh",
"kerronxy_users");
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT lat, lon, image_path FROM `images` ";
$result = $conn->query($sql);
$data = array();
if ($result->num_rows > 0) {

    while($row = $result->fetch_assoc()) {
        array_push($data,$row);
    }
} else {
    echo "0";
}

echo json_encode($data);
$conn->close();
?>

?>
```

## googleLogin.php

קוד התחברות דרך גוגל - מקבל אימייל משתמש מצד הלקוח. מתחבר למוסד הנתונים, בודק האם אימייל קיים - אם לא מוסיף למוסד משתמש חדש.

```
<?php
// Create connection
$conn = mysqli_connect("78.140.191.36", "kerronxy_yacov", "]k2oI1?WBRPh",
"kerronxy_users");
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
    echo "Error";
}

$sql = "SELECT * FROM `users_table` WHERE email LIKE '". $_REQUEST['email'] . "'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo $row["email"] . " logged in";
    }
} else {

    $sql = "INSERT INTO `users_table`(`fullname`, `username`, `email`, `password`,
`gender`) VALUES ('". $_REQUEST['name'] . "','', '". $_REQUEST['email'] . "','', '')";

    if ($conn->query($sql) === TRUE) {
        $last_id = $conn->insert_id;
        echo $_REQUEST['email'];
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
}
$conn->close();
?>
```

## Login.php

קוד הזדהות וכניסה למערכת. מקבל מצד לקוח אימייל וסיסמה של המשתמש, מתחבר למוסד הנתונים ומנסה לשלוח משם נתונים התואמים. אם הצליח – מאשר התחברות למערכת, אם לא – שולח הודעת שגיאה מתאימה. (כגון: סיסמה לא נכונה\ משתמש לא קיים)

```
<?php
```

```
require_once 'connect.php';
```

```
$isVaildEmail = filter_var($email, FILTER_VALIDATE_EMAIL);
```

```
if($conn){
```

```
    if($isVaildEmail === false){
```

```
        echo "This Email is not valid";
```

```
    }else{
```

```
        $sqlCheckEmail = "SELECT * FROM users_table WHERE email = '$email'";
```

```
        $emailQuery = mysqli_query($conn, $sqlCheckEmail);
```

```
        if (mysqli_num_rows($emailQuery) >0){
```

```
            $sqlLogin = "SELECT * FROM users_table WHERE email = '$email' AND
```

```
password = '$password'";
```

```
            $loginQuery = mysqli_query($conn, $sqlLogin);
```

```
            if(mysqli_num_rows($loginQuery) >0){
```

```
                echo "Login success";
```

```
            }else{
```

```
                echo "Wrong password";
```

```
            }
```

```
        }else{
```

```
            echo "This Email is not registered";
```

```
        }
```

```
    }
```

```
}else{
```

```
    echo "Connection Error";
```

```
}
```



## Register.php

בביצוע הרשמה השרת בודק תקינות אימייל וסיסמה, ואז מנסה לרשום משתמש חדש למערכת. אם אימייל כבר תפוס – הרישום יכשל ותשלח הודעה על כך.

```
<?php
```

```
require "connect.php";
```

```
$fullname = $_POST['fullname'];
$username = $_POST['username'];
$email = $_POST['email'];
$password = $_POST['password'];
$gender = $_POST['gender'];
```

```
$isVaildEmail = filter_var($email, FILTER_VALIDATE_EMAIL);
```

```
if($conn){
    if(strlen($password) > 40 || strlen($password) < 6){
        echo "Password must be less than 40 and more than 6 characters";
    } elseif ($isVaildEmail === false){
        echo "The email is not valid";
    }else{
        $sqlCheckUsername = "SELECT * FROM users_table WHERE username LIKE '$user-
name'";
        $usernameQuery = mysqli_query($conn, $sqlCheckUsername);

        $sqlCheckEmail = "SELECT * FROM users_table WHERE email LIKE '$email'";
        $emailQuery = mysqli_query($conn, $sqlCheckEmail);

        if(mysqli_num_rows($usernameQuery) > 0){
            echo "User name is already in use, try another one";
        }elseif (mysqli_num_rows($emailQuery) > 0){
            echo "This Email already registered, Try another Email";
        }
        else{
            $sql_register = "INSERT INTO users_table(fullname, username, email,
password, gender) VALUES ('$fullname', '$username' , '$email', '$password', '$gen-
der')";

            if(mysqli_query($conn, $sql_register)){
                echo "Successfully Registered";
            }else{
                echo "Failed to Register";
            }
        }
    }
}
```

## upload.php

קוד העלאת תמונה. מקבל בPOST בקשת העלאת תמונה. שולף את התמונה מדטה שהתקבל ובודק האם זה תמונה באמת או קובץ אחר כלשהו. אם אכן התקבלה תמונה, אנו שולפים מהדטה את הנתונים של המיקום שם התמונה ואימייל המשתמש, יוצרים קובץ תמונה על שרת עם נתונים נ"ל (בודקים שהוא כבר לא קיים). מכניסים את הנתונים האלה גם למוסד הנתונים ביחד עם כתובת אינטרנט שנוצרה לקובץ ושולחים בחזרה הודעה על הצלחה. (או כישלון, עם הודעה מפורטת במקרה שמהו השתבש).

```
<?php
```

```
$conn = mysqli_connect("78.140.191.36", "kerronxy_yacov", "]k2oIl?WBRPh",
"kerronxy_users");

$target_dir = "uploads/";
$target_file = $target_dir . basename($_FILES["image"]["name"]);
$uploadOk = 1;
$imageFileType = strtolower(pathinfo($target_file,PATHINFO_EXTENSION));
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
    $check = getimagesize($_FILES["image"]["tmp_name"]);
    if($check !== false) {
        echo "File is an image - " . $check["mime"] . ".";
        $uploadOk = 1;
    } else {
        echo "File is not an image.";
        $uploadOk = 0;
    }
}
// Check if file already exists
if (file_exists($target_file)) {
    echo "Sorry, file already exists.";
    $uploadOk = 0;
}

// Check if $uploadOk is set to 0 by an error
if ($uploadOk == 0) {
    echo "Sorry, your file was not uploaded.";
// if everything is ok, try to upload file
} else {
    if (move_uploaded_file($_FILES["image"]["tmp_name"], $target_file)) {
        $sql = 'INSERT INTO `images` ( `image_name`, `image_path`, `user_email`,
`lat`, `lon`) VALUES
```

```

    ("lat'._REQUEST['lat'].lon'._REQUEST['lon'].image'.base-
name($_FILES["image"]["name"]).'", "https://kerron.xyz/htdocs/uploads/'.base-
name($_FILES["image"]["name"]).'", "._REQUEST['email'].'", "._REQUEST['lat'].'",
    "._REQUEST['lon'].')");
    if ($conn->query($sql) === TRUE) {
        echo "1";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
} else {
    echo "Sorry, there was an error uploading your file.";
}
}
?>

```

\* חלק מהקוד שנכתב לא נכנס לכאן כיוון, שלא מהווה חלק בלתי נפרד מהפרויקט (כמו קבצי עזר של למידה עמוקה שמסדרים תמונות בתיקיות לשולפים תמונות קטנות מגדולות, בעזרת פענוח קבצי XML אשר מתארים נקודות בהם בתמונה גדולה נמצא בור) או לא מהווה שום אינטרס בתור קוד שצריך להסביר אותו (כגון קבצי XML של layout באנדרואיד שמתארים מיקום התוכן על מסכי המשתמש).

\*\* הפרויקט ניתן למצוא ולהוריד כאן: <https://github.com/KerronXR/PTS>