שנקר-בי"ס גבוה להנדסה ולעיצוב

הפקולטה להנדסה

המחלקה להנדסת תוכנה

מערכת ניהול וניווט מזל"ט AR.DRONE דרך טלפונים חכמים

פרויקט גמר

מגישים:

שמואל מלמד, 036690295

ליטל מוטולה, 200330983

מנחה:

ד"ר אמנון דקל

מוגש כחלק מהדרישות לקבלת תואר ראשון

בוגר במדעים (B.Sc.).

30/08/2013

תודות

אנו רוצים להודות בראש ובראשונה למנחה הפרויקט, ד"ר אמנון דקל, שידע להכוון אותנו בכל מהלך תיכון ופיתוח הפרויקט ועל ההנחיות שנתן לנו בקורס "סמינר פרויקט גמר".

אנו רוצים להודות לכל סגל מכללת שנקר להנדסה ברמת גן על העזרה והתמיכה בעת הצורך.

תודה רבה גם למשפחות ולחברים שתמכו בנו לאורך כל הדרך.

תודה לסטודנטים שעזרו לנו במהלך הפרויקט.

תודה רבה,

שמואל מלמד וליטל מוטולה.

תקציר מנהלים

אהבתנו לטיסנים הייתה הבסיס לרעיון הפרויקט. הטיסנים שהכרנו עד כה היו מבוססים על תקשורת המגבילה את הטסתם לטווח של בין 150 מטר למספר קילומטרים ספורים. הגבלה נוספת שזיהינו היא כמות האינפורמציה על הטיסה שניתן לקבל בזמן אמת וקבלתה רק ממקור אחד ומיקום פיזי יחיד. מכאן הגיע הרעיון למצוא דרך בא נוכל להטיס טיסן כמעט מכל מקום וללא התחשבות במיקומו הפיזי ולייצר אפשרות בא ניתן לראות נתוני טיסה רבים יותר ומכל מקום ללא התחשבות במיקום הפיזי של האפליקציה/שלט.

הטיסן שעמד לרשותנו הינו טיסן של חברת Parrot הנקרא בשם Ar.Drone, טיסן העובד בטכנולוגיית Wi-Fi המגבילה את טווח הטיסה לכ-150 מטר בלבד. השליטה על הטיסן נעשית ע"י אפליקציה ייעודית המותאמת גם ל-IOS וגם ל-Android.

לאחר סקירה של כלל הטכנולוגיות, מצאנו כי הדרך הטובה ביותר למימוש הרעיון הינו שימוש באנטנות סלולריות הפרוסות בכל בעולם. אנטנות אלו מספקות אינטרנט למשתמשים בטלפון חכם ובכך מאפשרות לתקשר עם כל אחד מכל מקום, על כן תכננו והקמנו מערכת המאפשרת שליחת הודעות על גבי אנטנות סלולריות מטלפון חכם אחד לטלפון חכם אחר וממנו לטיסן ע"י תקשורת Wi-Fi בה הטיסן יודע לתקשר.

על מנת להעביר ולנהל את ההודעות בין המכשירים הסלולריים ייצרנו שרת שדרכו הם מתקשרים, מכאן שהשרת חייב להיות פעיל על מנת שיהיה קשר בין המשתמש לטיסן, במידה והשרת נופל תכננו מספר אלגוריתמים שבהם הטיסן ידע לבצע חזרה לנקודת מוצא או נחיתה במקום בטוח.

לצורך מימוש הפרויקט השתמשנו בטכנולוגיות שאפשרו לנו מגוון רחב של פעולה ולכן הוספנו לרעיון את האפשרות של טיסה אוטומטית שהפך את הטיסן שלנו למזל"ט ואפשר לנו למצוא לטיסן דרכים שימושיות לתעשייה ולא רק בתור תחביב למשתמש הביתי.

אנו תקווה שפרויקט זה לא הסתיים בהגשתו אלא יהיה בסיס לעתיד טוב יותר.

תוכן עניינים

מבוא

הקדמה

תיק הפרויקט מתאר את תהליך הפיתוח של אפליקציית AA-UAV בהנחיית ד"ר אמנון דקל, הפרויקט יכול להתאים לכל גרסאות טיסני Ar.Drone של חברת Parrot.

הפרויקט מוגש כחלק אינטגרלי מהדרישות לקבלת תואר במדעים (B.Sc) ב"שנקר" – בית ספר גבוה להנדסה ועיצוב.

הפרויקט בנוי ממספר חלקים עיקריים על מנת שהמערכת כולה תעבוד :

- אפליקציית משתמש – שליטה ובקרה על הטיסן.
- אפליקציית טיסן – העברת פקודות הבקרה לרכיב Arduino Mega 250.
- רכיב Arduino Mega 250 – העברת הפקודות לרכיב ה Wi-Fly.
- כרטיס Wi-Fly – הגדרת הכרטיס ליצירת תקשורת עם הטיסן.
- תוכנת השרת – העברת ההודעות מאפליקציית משתמש לאפליקציית הטיסן ולהיפך, הצגת הנתונים והעברת ההודעות הרלוונטיות לבסיס הנתונים MySQL.
- אתר – הסבר כללי על ביצוע הפרויקט וניתור אחר כל הטיסנים המשתמשים באפליקציה ע"י הצגתם על מפה של גוגל והצגת נתוני הטיסה של כל טיסן בזמן אמת.

במסגרת הפרויקט בחרנו שהמערכת "תרוץ" על גבי טלפון חכם אשר מבוסס פלטפורמת Android. בהמשך יינתן פירוט על פלטפורמות נוספות לפיתוח והסבר מדוע בחרנו בפלטפורמת Android של חברת © Google.

מוטיבציה

התפתחות תעשיית הטלפונים החכמים פתחה צוהר של אפשרויות המנצלות את יתרון הניידות – קבלת מידע רב ומגוון מכל מקום ובכל זמן נתון.

התפתחות תעשיית האלקטרוניקה פתחה גם היא צוהר של אפשרויות המנצלות את יתרון המזעור – רכיבים קטנים וקלים מאוד המבצעים מגוון רחב של פעולות במחירים זולים ביותר.

התפתחויות אלו נתנו לנו את האפשרות והמוטיבציה לקחת את הטיסן הפשוט ולשלב בו את הרכיבים המתאימים על מנת לאפשר לו שלל רחב יותר של אפשרויות ובכך להפוך אותו למזל"ט – מטוס זהיר ללא טייס.

אפשרויות אלו מהוות קפיצת מדרגה בתחום הטיסנים ומאפשרת בעתיד את שילובם בתחומים רבים בתעשייה.

רקע

הפרויקט התבצע ע"י שמואל מלמד וליטל מוטולה, סטודנטים בשנה ד' להנדסת תוכנה במסלול התמחות Mobile.

רעיון הפרויקט הגיע מסרטים עתידיים ומשירותו הצבאי של שמואל, שם עסק בתקשורת למזל"טים. אלו גרמו לנו לחשוב למה אין מזל"טים פרטיים וזולים ואיך אפשר ליישם מזל"ט כזה.

בסרטים עתידיים ניתן לראות כי עשו שימוש ברעיון למטרות רבות כגון אבטחה, שמירה על סדר, איסוף מידע, מעקבים, העברת מידע ועוד.

מכאן שראינו לנכון ליישם את הרעיון ולהפוך אותו למציאותי.

מילון מונחים

- פלטפורמה – סדרה משולבת של מרכיבים המאפשרת מודלים שונים לפיתוח תוכנה.
- AA-UAV – שם הפרויקט: Ar.Drone Android – Unmanned Aerial Vehicle כלומר מטוס זהיר ללא טייס מסוג Ar.Drone המופעל באמצעות אפליקציית Android.
- UAV- מזל"ט – מטוס זהיר ללא טייס: UAV – Unmanned Aerial Vehicle הוא כלי טייס, כלומר מטוס או מסוק שאינו נושא עליו בני אדם ובפרט טייס (Resident pilot).

מטרות הפרויקט

מטרת הפרויקט היא להקנות למשתמש את היכולת להטיס את הטיסן מכל מקום ללא התחשבות במיקומו הפיזי של הטיסן ולקבל נתוני טיסה בזמן אמת.

מטרות שהושגו:

- השגת כל הרכיבים הנחוצים להצלחת הפרויקט.
- שימוש ברשת הסלולרית ליצירת תקשורת בין המכשירים.
- יצירת שרת המנהל את תעבורת ההודעות בין המכשירים.
- אימות הטיסן באפליקציית המשתמש ע"י מספר סידורי רנדומלי בן 10 ספרות.
- טיסה אוטומטית ע"י בחירת נקודת יעד על גבי מפה של גוגל. (כרגע טס אך ורק ישר ללא שימוש במצפן וג'ירוסקופ).
- אתר המציג על גבי מפה של גוגל את מיקום הטיסן ונתוני הטיסה בזמן אמת.
- הצגת נתוני טיסה בזמן אמת באפליקציית המשתמש.
- נחיתה אוטומטית במידה והסוללה של המכשיר הנמצא על גבי הטיסן ירדה מתחת ל- 20%.
- שמירת נתוני הטיסה במסד נתונים.
- חלוקה לשלושה סוגי טיסנים: טיסני משטרה, תעבורה ומשחק. (כרגע סוג משחק הוא מצב ברירת המחדל).

מטרות אופציונליות שלא הושגו:

- העברת ווידיאו מהטיסן לאפליקציית המשתמש.
- שימוש בתחזית מזג האוויר לעקיפת מסלולים המסוכנים לטיסה (בזמן טיסה אוטומטית).
- שליטה על הטיסן באמצעות ג'ירוסקופ.
- שליטה על הטיסן באמצעות Voice command.
- עקיפת עצמים מסוכנים בדרך, כגון בניינים ועצים.
- הצגת סטטיסטיקות המבוססות על נתוני הטיסה שהטיסן צבר במהלך הטיסות ונשמרו במסד הנתונים.
- בחירת סוג הטיסן באפליקציית המשתמש.

דרישות הפרויקט (תקציר מסמך האפיון)

מסמך האפיון הוא דו"ח ההתקדמות הראשון במהלך הפרויקט.
דו"ח זה מכיל את הסעיפים הבאים:

- מטרת הפרויקט.
- תרחישים.
- מטרות שנבצע במידה ויתאפשר.
- תרשים זרימה המתאר את תהליך השימוש ברכיבי הפרויקט.
- בעיות פתוחות.

דרישות כלליות

- המערכת צריכה להיות ידידותית למשתמש.
- המערכת צריכה להתאים לילדים ומבוגרים כאחד.
- המערכת צריכה להיות מבוססת Android.

דרישות תפעול

- יצירת שרת לניהול התעבורה בין המכשירים.
- התממשקות השרת לבסיס נתונים.
- יצירת קשר בין מכשיר הטלפון לרכיב Arduino Mega דרך USB.
- יצירת קשר בין רכיב Arduino Mega לרכיב ה Wi-Fly.
- יצירת קשר בין רכיב ה Wi-Fly לטיסן.
- הספקת חשמל לרכיב Arduino Mega ורכיב ה Wi-Fly.
- התממשקות לגוגל מפות.

תקציר מסמך התיכון

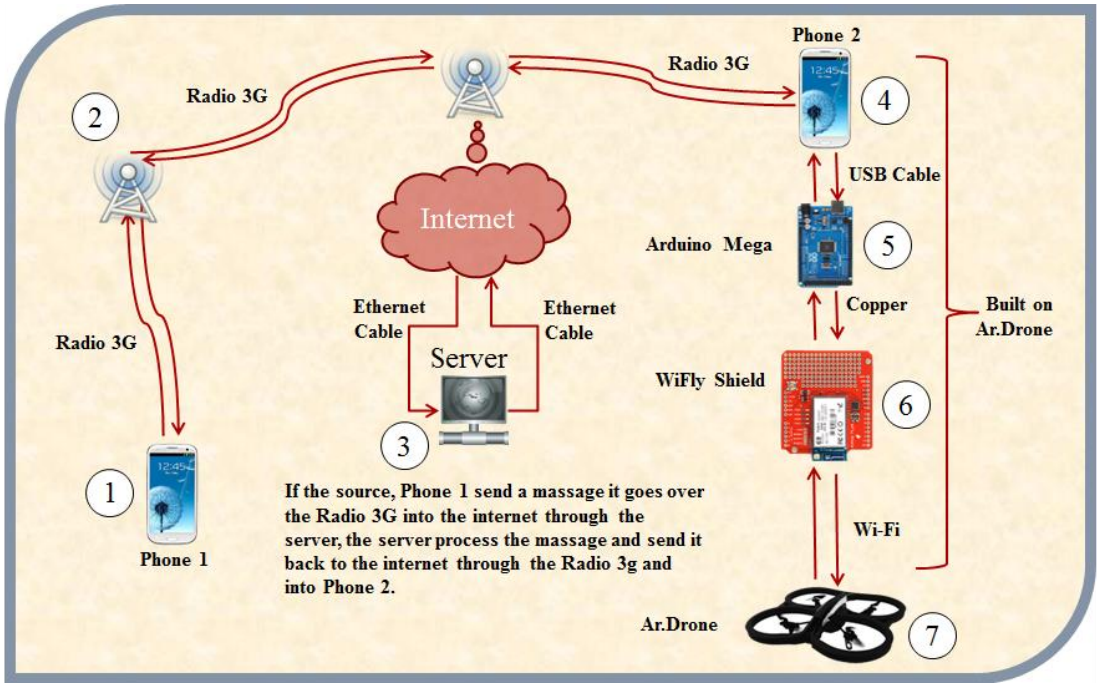
ארכיטקטורת המערכת

לצורך בניית הארכיטקטורה חקרנו את הנושא לעומקו כדי למצוא את הדרך הטובה והפשוטה ביותר לממשו. תוך כדי המחקר גילינו כי שימוש ברשת סלולרית וברשת Wi-Fi בו זמנית אינה אפשרית אלא אם כן מכשיר הטלפון פרוץ. מכיוון שפריצת מכשיר הטלפון מבטלת את אחריותו נאלצנו למצוא דרך חליפית. לאחר מחקר נוסף גילינו כי ישנם רכיבים אלקטרונים המאפשרים חיבור ישיר לטלפון בעזרת USB ובנוסף חיבור Wi-Fi לכיוון הטיסן. רכיבים אלו נבחרו בקפידה לאחר מחקר מעמיק בו נמצא כי ניתן לחברם לשאר רכיבי הארכיטקטורה.

אופן שליחת ההודעות בין המכשירים הסלולריים גם נלקחה בחשבון עקב הצורך באימות הטיסן מול אפליקציית המשתמש והצגת נתוני הטיסה בזמן אמת באתר הפרויקט, על כן החלטנו כי יש צורך במימוש שרת שינהל וישמור את המידע המגיע מהטיסן לבסיס הנתונים משם האתר ישאב את נתוני הטיסה בזמן אמת.

ממשקי חומרה ותקשורת

בשרטוט זה ניתן לראות את ארכיטקטורת המערכת בשלמותה:



טלפון חכם- Phone 1 (1)

טלפון חכם זה הוא טלפון המריץ את אפליקציית המשתמש על פלטפורמת אנדרואיד. אפליקציית המשתמש בודקת תקשורת לרכיבים המקושרים אליה ומאפשרת שליטה ידנית על הטיסן, קבלת נתוני טיסה מהטיסן בזמן אמת והפעלת טייס אוטומטי המאפשר לטיסן לטוס ללא שליטה ידנית של המשתמש.

אנטנת סלולר- Radio 3G (2)

"הדור השלישי" (3G) הוא כינוי לקבוצה של פרוטוקולי תקשורת לקשר רדיו שהוגדרו על ידי איגוד הטלקומוניקציה הבינלאומי (ITU). בקבוצה זו ישנם מספר תקנים, כגון EDGE ו-UMTS, המיועדים לאפשר תקשורת אלחוטית של שיחות טלפוניה, שיחות וידיאו, תקשורת נתונים - וכולם מתוך המכשירים הנמצאים בתנועה. קצב הנתונים בדור השלישי מיועד להיות עד 14.4Mbps בכיוון היורד, ועד ל 5.8Mbps בכיוון העולה. אנטנות הסלולר מאפשרות שידור ברדיוס של עד 32.8 קילומטר. אנטנות אלו פרוסות כמעט בכל העולם על מנת לאפשר תקשורת טלפונית וסלולרית בין המשתמשים הרבים, על כן בחרנו בתשתית תקשורת זו בארכיטקטורה שלנו על מנת לאפשר שליחת הודעות ממכשיר המשתמש למכשיר הטיסן ובכך לגרום למצב שבו האזורים שבהם ניתן להטיס את הטיסן לא יהיו מועטים (למעט אזורים שלא ניתן לשים בהן אנטנה כגון האוקיינוס).

השרת- Server (3)

השרת מטרתו לנהל את ההודעות העוברות בין המכשירים הסלולריים. במידה והשרת נופל, לא יהיה ניתן לשלוח הודעות והמערכת כולה לא תעבוד ותאפשר הטסה של הטיסן.

טלפון חכם- Phone 2 (4)

טלפון חכם זה הוא טלפון המריץ את אפליקציית הטיסן על פלטפורמת אנדרואיד. אפליקציית הטיסן בודקת תקשורת לרכיבים המקושרים אליה ומייצרת מספר רנדומלי בין 10 ספרות המיועד לאימות הטיסן מול אפליקציית המשתמש. בנוסף, אפליקציה זו מאפשרת את שליחת נתוני הטיסה למשתמש ושליחת פקודות השליטה לרכיב Arduino Mega 250.

רכיב תקשורת Arduino Mega 250 (5)

רכיב זה מאפשר לחבר אליו את הטלפון החכם ע"י חיבור USB וחיבור רכיב ה Wi-Fly ע"י כבלי נחושת. אופציה זו מאפשרת לנו לתקשר מצד אחד עם הטלפון החכם ומצד שני עם הטיסן ע"י רכיב ה Wi-Fly. לרכיב זה ישנו זיכרון שבו ניתן לאחסן את קוד התוכנה המאפשרת את מימוש התקשורת וניהול ההודעות העוברות בין הטלפון שמריץ את אפליקציית הטיסן לבין רכיב ה Wi-Fly.

רכיב תקשורת Wi-Fly (6)

רכיב זה משמש ככרטיס רשת אלחוטי נייד המאפשר לנו להתחבר לרשת המופצת מהטיסן ולקבל כתובת IP, לכרטיס זה ניתן להגדיר אפשרויות תקשורת נוספות. לצורך הפרויקט נאלצנו לשנות את פרוטוקול התקשורת לפרוטוקול UDP – User Datagram Protocol המאפשר העברת נתונים לא אמינה, כלומר במידה ונשלחה הודעה אין צורך לבדוק האם הודעה זו הגיעה ליעד שלה. אנו משתמשים בפרוטוקול זה מפני שהוא מהיר יותר.
בנוסף, שינינו את מספר ה- Port שדרכו אנו מעבירים את ההודעות על מנת שיהיה זהה לזה של הטיסן.

הטיסן Ar.Drone (7)

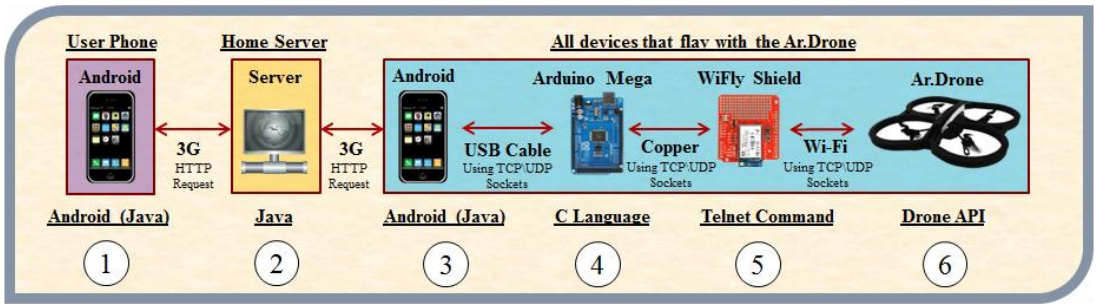
ה- AR.Drone שאנו מכנים כטיסן הוא מסוק זעיר הנשלט באמצעות טלפונים חכמים המריצים IOS/Android. מערכת ה- Wi-Fi המותקנת בגוף המסוק מאפשרת לשלוט בו באמצעות מכשירי הסלולר השונים.

המסוק מצויד בטכנולוגיית חיישנים מתקדמת להטסה פשוטה: חיישני תנועה, מצלמה אחת מותקנת בחזית , מצלמה שנייה מותקנת בתחתית לצילום כולל וכיסוי כל זוויות הטיסה, בנוסף מערכת אלקטרו-מכנית (MEMS), מד-תאוצה, חיישני ג'ירוסקופ וחיישן אולטרסאונד. כולם מחוברים למעבד העוצמתי שעל גוף המסוק והופכים את הטסתו לנוחה, יציבה ובטוחה.

בנוסף מותקן על המסוק "טייס אוטומטי" המאפשר המראה ונחיתה קלה. כאשר המסוק מופעל ונמצא על הרצפה, על מנת להמריא כל שנותר לעשות הוא ללחוץ על כפתור "take off"שנמצא על vאפליקציה באמצעותו מטיסים והמסוק ממריא באופן אוטומטי ויציב. לאחר ההמראה, ה- AR.Droneמתייצב בגובה של 80 ס"מ ונשאר באוויר באותה נקודה ללא תזוזה עד לפקודה ישירה מהמשתמש. במידה ומאבדים את הקשר עם המסוק- הטייס האוטומטי תופס פיקוד ואת השליטה ומייצב את המכשיר לנחיתה עדינה ובטוחה.

בחרנו להשתמש בארכיטקטורה שלנו בטיסן/מסוק זה מפני שהוא מבוסס על תקשורת Wi-Fi וקוד פתוח ומאפשר לנו להתממשק אליו בקלות.

רכיבי וממשקי תוכנה



רכיבי וממשקי התוכנה מתחלקים לשלושה חלקים:

* טלפון המשתמש בו מותקנת אפליקציית המשתמש המאפשרת שליטה על הטיסן.

* השרת שמנהל את שליחה וקבלת ההודעות ושמירת המידע הרלוונטי בבסיס נתונים.

*הטיסן אשר טס ביחד עם הרכיבים האלקטרונים והמכשיר הסלולרי בו מותקנת אפליקציית הטיסן המאפשרת קבלת הודעות מהמשתמש ושליחתן לטיסן דרך רכיב ה Wi-Fly.

שלושת החלקים נכתבו בשפת Java בסביבת Eclipse פרט לרכיב Arduino Mega 250 אשר נכתב בשפת C בסביבת Arduino.

טלפון המשתמש- User Phone (1)

טלפון המשתמש מכיל את אפליקציית המשתמש. אפליקציה זו בנויה בצורה וויזואלית ונוחה למשתמש ומבצעת את הפעולות הבאות: שליחה וקבלה של הודעות שליטה ונתונים על גבי הרשת הסלולרית באמצעות Socket- ים הנפתחים מול השרת על גבי פורט 5555 הייחודי למערכת זו, התממשקות לגוגל מפות והצגת מיקום הטיסן על המפה, הצגת נתוני הטיסה.

אפליקציה זו נכתבה בשפת Java בסביבת Eclipse Juno ובשימוש של ADT – Android Development Tools.

השרת- Home Server (2)

תוכנת השרת בנויה בצורה וויזואלית ונוחה למשתמש. אפליקציה זו מכילה תפריט המאפשר להתחיל או להפסיק את פעולת השרת, חלון טקסטואלי בו ניתן לראות את ההודעות העוברות דרך השרת וסרגל ימני בו ניתן לראות את נתוני הטיסה.

השרת הוא המגשר בין אפליקציית המשתמש לאפליקציית הטיסן ולכן מכיל שירות הנקרא Controller. שירות זה יודע להחליט האם צריך לשלוח את ההודעות לטיסן, למשתמש או לבסיס הנתונים. שרת זה נבנה כשרת Multithreading מפני שעבור כל הודעה מתקבלת בשרת Socket שונה, ויוצרת מצב בו כל הודעה צריכה טיפול נפרד. שירות נוסף שמאפשר השרת הוא גישה לבסיס הנתונים MySQL וממשש את הפעולות: הוספה, עדכון וקבלה. פעולות אלו נכתבו בשילוב של Java ו- SQL.

שרת זה נכתב בשפת Java בסביבת Eclipse Indigo ובשימוש של Window Builder Pro ו- MySQL connection deriver.

טלפון הטיסן- Plane Phone (3)

טלפון הטיסן מכיל את אפליקציית הטיסן. אפליקציה זו בנויה בצורה וויזואלית ונוחה למשתמש, אפליקציה זו מבצעת את הפעולות הבאות: שליחה וקבלה של הודעות שליטה ונתונים על גבי הרשת הסלולרית באמצעות Socket- ים הנפתחים מול השרת על גבי פורט 5555 הייחודי למערכת זו, מימוש Client היוצר קשר עם רכיב Arduino Mega 250 על גבי כבל USB באמצעות Socket- ים הנפתחים מול שרת היושב על הרכיב בפורט 4568.

אפליקציה זו נכתבה בשפת Java בסביבת Eclipse Juno ובשימוש של ADT – Android Development Tools.

רכיב אלקטרוני Arduino Mega ADK (4)

תוכנת הרכיב מיישמת שרת המקבל את הודעות השליטה מאפליקציית הטיסן ומעבירה אותן באופן סריאלי לרכיב ה- Wi-Fly.

תוכנה זו נכתב בשפת C בסביבת עבודה Arduino ובשימוש של ספריית התקשורת ADB.

רכיב אלקטרוני Wi-Fly Shilled (5)

ברכיב זה הגדרנו את אופן קבלת כתובת הרשת כ- Auto, את פרוטוקול הנתונים UDP ואת הפורט גישה לטיסן 5556.

רכיב זה הגרנו באופן סריאלי ע"י סביבת העבודה Arduino.

הטיסן Ar.Drone (6)

לטיסן זה לא היה צורך בכתיבת קוד או תוכנה מפני שהטיסן מגיע עם רכיב המכיל בתוכו תוכנה מוכנה בקוד פתוח המאפשרת ביצוע כל פעולות הטיסן ע"י קבלת פקודות AT המוגדרות מראש. פקודות אלו אנו שולחים מאפליקציית המשתמש עד אשר היא מגיעה לטיסן וכאשר הטיסן מקבל את הפקודה הוא יודע איזה קוד לבצע על מנת לבצע את הפעולה המבוקשת. לטיסן יש גם פקודות המבוצעות אוטומטית במצבי סכנה כגון נחיתה לפני שהסוללה נגמרת או הפסקת פעולות הטיסן כאשר אחד המנועים נתקע כתוצאה מגורם חיצוני. יש לציין שאת כל תוכנת הטיסן המובנת בתוכו ניתן לשנות מפני שזהו קוד פתוח ולכן אנו יכולים לגשת לקבצים המכילים את תוכנת הטיסן ע"י קישור FTP ולשנות אותם כרצוננו.

סקירת ספרות

סקירת פתרונות קיימים

מחיפוש מהיר באינטרנט לא נמצאו פתרונות זהים לרעיון הפרויקט אלא פתרונות חלקיים שהגיעו לידי מימוש ובדיקה בלבד.

אפליקציות אנדרואיד דומות

כיום קיימות אפליקציות דומות שמשתמשות ברשתות תקשורת אחרות (RC, WIFI). בדקנו מה הם היתרונות והחסרונות של שימוש בכל סוג תקשורת, אילו אפשרויות קיימות בכל אפליקציה (כגון: צילום טיסה, נתוני טיסה בזמן אמת ועוד.) והשווינו בין הרעיון שלנו לפתרונות הקיימים.

הבעיות שגילינו כאשר משתמשים בתקשורת שאינה סלולרית:

- טווח טיסה מוגבל – עד כמה מאות מטרים.
- נתוני טיסה בזמן אמת – נתונים מינימליים.
- יכולות טיסה מוגבלות – הטיסן טס ע"י שליטה של המטיס בלבד.
- תקשורת מוגבלת – תקשורת RC מחייבת לתת לכל טיסן תדר משלו על מנת שלא נעלה בטעות על תדר של מישהו אחר.
- עלות – יקרה יחסית לפונקציונאליות של הטיסן.

האפליקציות שקיימות משתמשות בסוגי תקשורת הבאים:

- RC Drone - אפליקציית השתלטות על טיסן באמצעות רשת RC.

במהלך סקירה זו נעזרנו במספר מקורות:

1. מידע לגבי הטסת טיסנים:
http://www.tisan.co.il/Articles/FlyCourse/FlyCourse.html
2. סימולטור: http://www.realflight.com
3. ספר המסביר כיצד להטיס טיסן המבוסס על תקשורת זו:

http://www.rc-airplane-advisor.com/support-files/learn-to-fly-rc-airplanes-rev1.2.pdf

- Wi-Fi AR.Drone - אפליקציית השתלטות על טיסן באמצעות רשת אלחוטית.
מקורות מידע בהם נעזרנו:

http://ardrone2.parrot.com/

http://ro-botica.com/img/Parrot_AR.Drone/ar.drone_user-guide_sp.pdf

סקירת טכנולוגיה

לאורך הפרויקט השתמשנו בטכנולוגיות רבות כגון: שפות תכנות, סביבות עבודה, שרתים, רכיבים אלקטרונים ועוד.

רכיבים אלקטרונים

AR.Drone

AR.Drone הוא מסוק פורץ דרך שמשלב את הטוב מכמה עולמות– עולם משחקי הווידאו, המודלים, ועולם המציאות המדומה. ה- AR.Drone נשלט מרחוק ע"י iPod touch, iPhone או iPad. המסוק מצויד במספר חיישנים וכולל מצלמה קדמית, מצלמה תחתית ומד-גובה שפועל בטכנולוגיית אולטרסאונד. ניתן להשתמש ב AR.Drone גם במשחקי וידאו, לדוגמה AR.FlyingAce- קרב אויר בין שני מסוקי AR.Drone. משחק זה כולל טכנולוגיות של מציאות מדומה הכוללת מכונות יירייה וירי טילים.

ה- AR.Drone יוצר רשת Wi-Fi משל עצמו בתצורת אד-הוק. משנוצרה, ה- iPhone יכול פשוט להתחבר אליה, כמו לכל רשת Wi-Fi. הטווח של ה- AR.Drone תלוי בסביבה שבה הוא מופעל. הטווח יהיה גדול יותר אם ה- AR.Drone מופעל בשטח פתוח לרווחה ללא הפרעות תקשורת היכולות להיווצר ממכשירים אחרים. טווח התקשורת בין ה- Phone והמסוק הוא בממוצע כ- 50 מטר.

Arduino

זו פלטפורמת קוד פתוח לפיתוח מוצרים פיזיים בעזרת לוח I/O פשוט וסביבת פיתוח הממשת שפות Processing ו -Wiring. כרטיסי Arduino זוכים לפופולריות רבה בכל העולם בזכות הפשטות, המחיר, היכולות, ההרחבות הרבות שנקראות "מגנים" (Shields) וסביבת הפיתוח החינמית.

ADK

ה-Arduino Mega ADK משתמש ברכיב ATmega8U2 לתקשורת USB. רכיב ה-ATmega8U2 הוא בעצם מעבד נוסף על הכרטיס ואפשר לתכנת אותו בעזרת חיבור ICSP נוסף. כרטיס זה מאפשר גם חיבור ישיר של מכשירי Android לתקשורת עם הכרטיס. חיבור ה-USB Host נעשה ע"י רכיב MAX3421e. מטרתו של הכרטיס לאפשר שליטה על חומרה חיצונית ע"י מכשיר Android תקני, מכאן השם שלו - ADK - Android Development Kit.

Wi-Fly

מודול RN-XV Wi-Fly מבוסס על מודול רשתות Wi-Fi - Roving RN-171 ומשלב b/g 802.11 radio, מעבד 32 ביט, מחסנית TCP / IP, שעון זמן אמת, חיישן Accelerator, יחידת ניהול צריכת חשמל וחיישן אנלוגי מודול טעון מראש עם הקושחה Roving כדי לפשט אינטגרציה ולצמצם את זמן פיתוח של היישום. בתצורה הפשוטה ביותר, החומרה דורשת רק ארבעה חיבורים (PWR, TX, RX והארקה) כדי ליצור חיבור נתונים אלחוטי.

הגדרת רכיב זה נעשית ע"י Telnet או גישה לרכיב באופן סריאלי ובעזרת פקודות פשוטות הנתונות בספר ההדרכה שמגיע עם הרכיב.

שרת

MySQL

MySQL הוא מסד נתונים רב משתמשים מבוסס שפת SQL (Structured Query Language) . נקרא ע"ש My אחד מהכותבים המקוריים. התוכנה פותחה במקור על ידי החברה השבדית MySQL AB. כיום היא בבעלות חברת אורקל. התוכנה היא חלק מ-LAMP - אוסף תוכנות תשתית פופולריות שעומדות בבסיסם של אתרים חשובים רבים, כגון גוגל וויקיפדיה. תוכנות רבות,כגון וורדפרס ודרופל, משתמשות בה כבסיס נתונים.

PHP

PHP (ראשי תיבות של "PHP Hypertext Preprocessor") היא שפת תסריט המיועדת בעיקר לתכנות יישומי אינטרנט בצד השרת, אך יכולה לרוץ על המחשב האישי באמצעות מפרש. התחביר של השפה דומה לזו של C והסמנטיקה דומה לזו של Perl. שפת PHP נוצרה במקור על ידי רזמוס לרדורף.

לשפת PHP יש מפרש (Interpreter) בשם זה המותקן בשרת ותפקידו להריץ תסריטים (Scripts) ב- PHP, תוך שימוש במשאבים של מחשב השרת (למשל: מערכת הקבצים ובסיסי נתונים). בדומה לטכנולוגיות צד-שרת נוספות, באמצעות PHP ניתן ליצור דפי Web דינמיים בשילוב נתונים מבסיסי נתונים, וכך לטפל בטפסים ובמידע שנשלחים מהמשתמשים (clients). התפקיד של PHP הוא לנתח את המידע וליצור פלט בהתאם.

אתר

JavaScript

JavaScript היא שפת תסריט מבוססת "תכנות מונחה דגמי אב" (Prototypes), שהוא סוג (לא נפוץ) של תכנות מונחה-עצמים. היא ידועה בעיקר כשפה המוטבעת בדפי HTML על מנת להציג דפים דינמיים, שמשולבת בהם תוכנה. קוד ה- JavaScript שמשולב בדף HTML מבוצע על ידי הדפדפן.

HTML5

HTML (ראשי תיבות של HyperText Markup Language) היא שפת תגיות לתצוגה ועיצוב דפי אינטרנט ותוכן לתצוגה בדפדפן. HTML תוכננה לעבוד על כל מחשב, מכל סוג והיא סלחנית מאד לגבי פרטים קטנים. היא מותרת לשימוש על ידי כל מפתח אתרים, ללא צורך ברכישת זכויות יוצרים מחברה כלשהי, והיא ניתנת לקריאה בכל סוגי המערכות.

HTML 5 הוא תקן חדש הכולל שפע של שינויים וחידושים. מי שעומד מאחורי התקן החדש הוא הקונסורציום העולמי (W3C). גוף בינלאומי שמסדיר את כל נושאי התקינה ברשת. התקן החדש מאפשר לנו לעשות דברים שעד כה היה מאד קשה לעשות אותם. כך למשל אנו מסוגלים ליצור גרפיקה באמצעות Canvas בדומה לטכנולוגיות Silverlight או פלאש. רק שבניגוד לאלו, ב- HTML 5 אין צורך בהתקנת תוכנה חיצונית על מנת לראות או ליצור את הגרפיקה הזו. אנו מסוגלים להטמיע וידאו או אודיו בעמוד שלנו בקלות מרובה, בדיוק כמו להציב תמונה. אלמנטים סמנטיים נותנים לנו יכולת לשלוט על מבנה העמוד שלנו בדרכים חדשות ויש עוד שפע של שיפורים שמאפשרים פיתוח אתרים ואפליקציות בעידן המודרני בקלות.

CSS

CSS גיליונות סגנון מדורגים (ראשי תיבות של Cascading Style Sheets) הם פורמט לעיצוב דפי אינטרנט. הגיליונות קובעים את עיצובם של תגיות ב HTML, XHTML וכל שפה דומה ל- XML לבניית אתרי אינטרנט. CSS נוצר במטרה להפריד בין תוכן ומבנה דפי האינטרנט לבין עיצובם. ל- CSS יש מספר גרסאות, האחרונה מבניהן היא CSS3.

JSON

JSON, קיצור של JavaScript Object Notation, הוא פורמט מחשב קל משקל להעברת מידע. פורמט זה הוא פורמט טקסטואלי, הקריא לאדם, המיועד לייצוג מבני מידע פשוטים ושכבות אסוציאטיביות (אובייקטים). תיאורו הרשמי של פורמט JSON מוגדר ב RFC-4627 שפותח על ידי דאגלס קרוקפורד. השם הרשמי של סוג מדיה זו הוא application/json וקבצים בפורמט זה הם בעלי סיומת json.

ב -JSON נעשה שימוש נרחב להעברת מבני מידע ברשת בתהליך הקרוי סריאליזציה. השימוש העיקרי בפורמט הוא במסגרת תכנות AJAX, שם משמש JSON תחליף אפשרי לפורמט XML.

Ajax

Ajax - Asynchronous JavaScript and XML- היא טכניקה ליצירת יישומי דפדפן אינטראקטיביים המבוססים על קוד המורץ במסגרת דף HTML בודד, ולא כיישום מרובה דפים, כמקובל בסביבת הווב. מטרתה העיקרית של הטכניקה היא שיפור חווית המשתמש והאצת מהירות הטעינה של דפי האינטרנט, מאחר שהיא מאפשרת לעדכן רק חלקים מבוקשים בדף האינטרנט, ללא צורך לטעון את הדף כולו מחדש במחשבו של המשתמש.

מטרה זו מושגת באמצעות יצירת תקשורת והחלפת מידע בין מחשב הלקוח לשרת דפי האינטרנט באמצעות קוד JavaScript.

SQL

SQL - Structured Query Language- היא שפת מחשב הצהרתית לטיפול ועיבוד מידע בבסיסי נתונים יחסיים, שפותחה על ידי IBM, והתבססה במקור על ידי אלגברה רלציונית, השפה מאפשרת תשאול נתונים, עדכונם ויצירת סכימה ושינוייה. שפה זו היא השפה הנפוצה ביותר לתשאול בסיסי נתונים יחסיים.

Google Maps for Mobile

Google Maps הוא יישום רשת של גוגל, המציג תמונות לוויין וצילומי אוויר של העולם כולו. לשירות יש שלושה מצבים עיקריים: לוויין, פני השטח ומפה. בנוסף לאלו אפשר להוסיף שכבות מידע נוספות כמו אתרים לפי סיווג, תמונות, ערכים על מקומות מוויקיפדיה, חיזוי תנועה, מבט מהרחוב (תצוגת רחוב) ובחירת מסלולים אופטימליים. גוגל מציעה אפשרות להוספת המפה גם באתרי צד ג'.

בחודש אוקטובר 2005, גוגל הציגה יישום Java בשם Google Maps for Mobile, המיועד לרוץ על כל טלפון מבוסס ג'אווה או מכשיר נייד. רבים מתכונות האתר מבוסס אינטרנט ניתנים ביישום.

PhpMyAdmin

PhpMyAdmin הוא כלי חופשי וקוד פתוח שנכתב ב- PHP ונועד להתמודד עם הניהול של MySQL עם השימוש בדפדפן אינטרנט. כלי זה יכול לבצע משימות שונות, כגון: יצירה, שינוי או מחיקה של מסדי נתונים, טבלאות, שדות או שורות, ביצוע משפטי SQL או ניהול משתמשים והרשאות.

סביבות עבודה

Eclipse

Eclipse היא פלטפורמת תוכנה פתוחה וחופשית לפיתוח יישומי צד-לקוח עשירים. Eclipse מנוהלת על ידי Eclipse Foundation, מוסד ללא כוונת רווח המתווה את כיווני ההתפתחות של הפלטפורמה. השימוש הרווח ביותר ב- Eclipse הוא ככלי פיתוח ליישומים בשפת Java. Eclipse מספקת סביבת פיתוח מתקדמת לשפה זו, והיא בעצמה משמשת לפיתוח פלטפורמת Eclipse, הכתובה גם היא ב Java. נוסף על כך, ל- Eclipse יש מספר רב של פרויקטים משניים, המספקים סביבות פיתוח לשפות אחרות כגון ++C,פייטון ורבות אחרות.

WampServer

WampServer היא סביבת פיתוח אינטרנט במחשב האישי. היא מאפשרת ליצור יישומי אינטרנט עם Apache2, PHP ו -MySQL-מסד הנתונים.
WampServer מותקנת באופן אוטומטי, ההתקנה והשימוש בו הוא מאוד אינטואיטיבי.

Sublime Text

Sublime Text הוא עורך טקסט מתוחכם לקוד הניתן לעבודה על גבי פלטפורמות Windows, Linux ו- OS X. עורך זה מאפשר עריכת קוד עבור שפות תכנות רבות כגון: HTML, CSS, JavaScript, PHP. מעבר לעריכת הטקסט, ישנם אפשרויות רבות ומגוונות בתוכנה כגון: ניצול גודל המסך לפיצול חלונות העריכה, בחירה ועריכה מרובת טקסט, השלמת מילים, הוספת תוספים המאפשרים הרחבה של פונקציות התוכנה. Sublime Text אינה קוד פתוח ולא חינמית, אך היא לא יקרה.

אפליקציות Mobile

ADT

כלי פיתוח אנדרואיד (ADT) הוא תוסף עבור Eclipse IDE אשר נועד לתת סביבה רב עוצמה משולבת לבניית יישומי אנדרואיד.

ADT מרחיב את היכולות של Eclipse כדי לאפשר להגדיר במהירות פרויקטים חדשים באנדרואיד, ליצור יישום ממשק משתמש, להוסיף חבילות המבוססות על API של אנדרואיד, ניפוי שגיאות ביישומים שלך באמצעות כלי SDK של אנדרואיד, ולייצא קובץ APK חתום (או לא חתום) על מנת להפיץ את היישום (אפליקציה).

Window Builder

WindowBuilder הוא כלי רב עוצמה וקל לשימוש עבור יצירת ממשק משתמש ללא צורך בבזוז זמן רב על כתיבת קוד על מנת להציג טופס פשוט. עם WindowBuilder אפשר ליצור חלון מסובך בתוך דקות. שימוש במעצב החזותי מנפיק את הקוד ב Java. הוספת פקדים באמצעות גרירה ושחרור, הוספת טיפול באירועים לפקדים, שינוי מאפייני הפקדים ועוד.

WindowBuilder נבנתה כתוסף ל- Eclipse ולשאר סביבות העבודה המבוססות על Eclipse כגון: RAD, RSA, MyEclipse, JBuilder. תוסף זה בונה עץ תחביר מופשט (AST) כדי לנווט את קוד המקור ומשתמש ב GEF כדי להציג ולנהל את ההצגה החזותית.

Android

אנדרואיד (Android) היא מערכת הפעלה המיועדת למכשירים ניידים ומבוססת על ליבת לינוקס. היא מופצת על ידי חברת גוגל בשיתוף פעולה עם Open Handset Alliance. מערכת הפעלה זו מעוצבת במיוחד לשימוש בטלפונים חכמים מבוססי מסך מגע ומחשבי לוח (טאבלטים). אנדרואיד פותחה לראשונה בידי חברת Android Inc ובשנת 2005 נרכשה על ידי Google. גוגל מפיצה את הקוד של אנדרואיד כקוד פתוח, תחת רישיון אפאצ'י. פרויקט הקוד הפתוח של אנדרואיד (Android Open Source Project, AOSP) שמובל על ידי Google הוא הפרויקט האחראי לתחזוקה ולפיתוח של מערכת ההפעלה.

Native

Native היא סוג האפליקציה שמימשנו בפרויקט זה, סוג זה של אפליקציה מאפשר למתכנת לגשת לרכיבי המכשיר כגון סנסורים, מצלמה, רמקול ועוד.

מכשיר הסלולר

Accelerometer

מד תאוצה, סנסור שמודד את התאוצה שמופעלת על המכשיר. תאוצה נותנת לנו מדד לכוח שמופעל על המכשיר, זה יכול להיות כוח המשיכה או כוחות שהמשתמש מפעיל על המכשיר (לדוגמא הזזת המכשיר מצד לצד). מד התאוצה מחזיר לנו וקטור שמציין מהו ערך התאוצה בכל אחד מהצירים X, Y, Z.

GPS

ב-1978 שוגר לוויין ה- GPS הראשון לחלל, עשרת הראשונים היו ניסיוניים ובין השנים 1989 -1994 שוגרו 24 לוויינים שהם המינימום הנדרש עד היום לכיסוי מלא של כדור הארץ. לווייני ה -GPS משדרים גלי רדיו (כל הזמן) שמציינים את השעה והמיקום שלהם בחלל.

חיישן ה- GPS קולט את שדרי הלוויין וכך יודע את מיקום הלוויין בשמים במדויק. כדי לברר היכן נמצא הוא צריך לקלוט שדרים של 4 לוויינים ובאמצעות המרה של הזמן למרחק הוא מגלה את מיקומו שלו.

הדיוק של השעון הפנימי של הלוויין הוא חיוני לפעולת ה- GPS אחרת החיישן יטעה בחישוב שמבוסס על מדידת זמן וטעות של שניות תתורגם לסטייה של מספר קילומטרים.

Compass

המצפנים הינם מכשירים למדידת זוויות במישור האופק . סוג החיישן שעל מכשיר הסלולר הוא חיישן מגנטי. עקרון הפעולה של חיישן מצפן מגנטי מבוסס על כך שלכדור הארץ יש שדה מגנטי שהקטבים שלו קרובים לקטבים הגאוגרפיים. במצפן המגנטי מותקנים מוטות (מחטים) מגנטיים אל לוח, שעליו יש חלוקה של מעגל האופק, והוא מורכב על ציר אנכי. המוטות המגנטיים, בכל מקום על כדור הארץ, מושפעים מהכוחות המגנטיים המקומיים, ומסובבים את הלוח המסומן . הלוח המסומן מראה בנקודת האפס שלו את צפון המצפן.

היישום

אפליקציית הטיסן

אפליקציית הטיסן בנויה מחבילות המכילות מספר רב של מחלקות ומתודות המהוות חלק חשוב בפעולותיה של האפליקציה ועל כן נפרט.

חבילות

Com.example.sensors_listener

חבילה זו מכילה מחלקות המאפשרות לנו גישה לערכי הסנסורים שעל מכשיר האנדרואיד.

Com.example.socket_adk

חבילה זו מכילה מחלקות המאפשרות יצירה של שרת לקוח בעזרת פרוטוקול TCP עבור יצירת תקשורת עם רכיב ה ADK.

Com.example.socket_server

חבילה זו מכילה מחלקות המבצעות את הפעולות המרכזיות של כל מסך ומסך באפליקציית הטיסן.

מחלקות חבילת Com.example.sensors_listener

MyAccelerometerListener

מחלקה זו מיישמת מתודות המקשיבות לאירועי חיישן מד תאוצה ובכך מאפשרת לנו גישה לערכיו בכל אחד מהצירים X, Y, Z המייצגים את וקטור התאוצה.

MyCompassListener

מחלקה זו מיישמת מתודות המקשיבות לאירועי חיישן המצפן ובכך מאפשרת לנו גישה לערכיו המציינים סיבוב סביב הציר (Roll) – X, סיבוב סביב הציר (Pitch) – Y וסיבוב סביב הציר (Yaw) – Z.

MyLocationListener

מחלקה זו מיישמת מתודות המקשיבות לאירועי חיישן ה GPS ובכך מאפשרת לנו גישה לערכיו המציינים קו רוחב (Latitude), קו אורך (Longitude) וגובה (Altitude). בעזרת ערכים אלו מחלקה זו מיישמת מתודות נוספות המאפשרות לנו לקבל את המיקום (שם העיר) והמהירות בקמ"ש בה אנו נמצאים בכל זמן נתון.

מחלקות חבילת Com.example.socket_adk

AbstractServerListener

מחלקה זו מיישמת מתודות המקשיבות לאירועי השרת ומאפשרת לנו לדעת מתי השרת הופעל או הופסק, ומתי משתמש התחבר, התנתק או שלח הודעה.

Client

מחלקה זו מיישמת מתודות לקוח המאפשרות לנו את היכולת להתחבר, לשלוח ולקבל הודעות מהשרת באמצעות פרוטוקול TCP.

Server

מחלקה זו מיישמת מתודות שרת בפרוטוקול TCP קל משקל התומך במספר רב של לקוחות המחוברים על פורט קבוע מראש.

מחלקות חבילת Com.example.socket_server

SplashScreen

מחלקה זו מיישמת מתודות המציגות למשתמש את מסך הפתיחה של האפליקציה על ידי אפקט שינוי צבע הרקע והזזת סמל הפרויקט.

ConnectionTest

מחלקה זו היא מחלקה המציגה לנו באופן גרפי את מצב חיבורי התקשורת לרכיבים השונים ומציגה לנו כפתור לצורך בדיקה חוזרת במידה וישנה בעיה באחד החיבורים. מחלקה זו מיישמת מתודות הבודקות את החיבורים לכלל המרכיבים הדרושים (ADK, Server, GPS, Internet) על מנת להשתמש באפליקציית הטיסן.

MainActivity

מחלקה זו היא המחלקה הראשית המציגה למשתמש את סמל הפרויקט בשילוב של מספר סריאלי בן 10 ספרות המשמש את אפליקציית המשתמש לאימות הטיסן. מחלקה זו מיישמת מתודות המבצעות קבלה של הודעות המתקבלות מהמשתמש דרך השרת והעברתן לרכיב ה- ADK על גבי כבל USB בתקשורת TCP. בנוסף, מחלקה זו מבצעת מתודה השולחת לאפליקציית המשתמש ולשרת את נתוני הסנסורים של מכשיר הטלפון.

אפליקציית המשתמש

אפליקציית המשתמש בנויה מחבילות המכילות מספר רב של מחלקות ומתודות המהוות חלק חשוב בפעולותיה של האפליקציה ועל כן נפרט.

חבילות

Com.example.components

חבילה זו מכילה מחלקה המאפשרת הצגה של רכיב דינמי וגרפי הניתן להוספה במסכי האפליקציה. רכיב זה הוא סרגל התקדמות אנכי שיכול לשמש עבור סוגי מידע כגון מהירות, גובה, כמות ועוד.

Com.example.myJoyStick

חבילה זו מכילה מחלקות המאפשרות יצירה של שני רכיבי ג'ויסטיק דינמי וגרפי יחיד או כפול הניתנים להוספה במסכי האפליקציה. מחלקות אלו מטפלות גם באירועי הג'ויסטיק.

Com.example.socket_client

חבילה זו מכילה מחלקות המבצעות את הפעולות המרכזיות של כל מסך ומסך באפליקציית המשתמש.

מחלקות חבילת Com.example.components

VerticalProgressBar

מחלקה זו מיישמת מתודות המציגות באופן גרפי סרגל התקדמות אנכי הניתן להוספה במסכי האפליקציה. מתודות אלו מטפלות גם בקביעת הטווח המקסימלי, לקבוע ולקבל את הערך המוצג ולקבוע האם סרגל זה יוצג או לא על גבי מסך האפליקציה.

מחלקות חבילת Com.example.myJoyStick

JoystickView

מחלקה זו מיישמת מתודות המציגות באופן גרפי ודינמי ג'ויסטיק יחיד הניתן להוספה במסכי האפליקציה. מתודות אלו מטפלות גם באירועי הג'ויסטיק, אירועים כגון מה קורה בעת הזזה של הג'ויסטיק, עזיבה של הג'ויסטיק היחיד, חזרה של הג'ויסטיק למרכז ולחיצה על הג'ויסטיק.

DualJoystickView

מחלקה זו מיישמת מתודות המציגות באופן גרפי ודינמי ג'ויסטיק כפול המיוצג על ידי שני ג'ויסטיקים יחידים מהמחלקה JoystickView, וניתן להוספה במסכי האפליקציה. מתודות אלו מטפלות גם באירועי הג'ויסטיק הכפול, אירועים כגון מה קורה בעת הזזה של הג'ויסטיק, עזיבה של הג'ויסטיק, חזרה של הג'ויסטיק למרכז ולחיצה על הג'ויסטיק.

מחלקות חבילת Com.example.socket_client

SplashScreen

מחלקה זו מיישמת מתודות המציגות למשתמש את מסך הפתיחה של האפליקציה על ידי אפקט שינוי צבע הרקע והזזת סמל הפרויקט.

ConnectionTest

מחלקה זו היא מחלקה המציגה לנו באופן גרפי את מצב חיבורי התקשורת לרכיבים השונים ומציגה לנו כפתור לצורך בדיקה חוזרת במידה וישנה בעיה באחד החיבורים. מחלקה זו מיישמת מתודות הבודקות את החיבורים לכלל המרכיבים הדרושים (Server, Internet) על מנת להשתמש באפליקציית המשתמש.

ControllerScreen

מחלקה זו היא המחלקה הראשית המציגה לנו את המסך הראשי של האפליקציה בו ניתן לראות את שני הג'ויסטיקים המיועדים לתפעול הטיסן, ותפריט עליון בו ישנם שלושה פקדים המאפשרים להמריא / לנחות, להציג את נתוני הטיסה, לעבור למסך המפה בו ניתן לקבוע טייס אוטומטי. מחלקה זו מיישמת מתודות המבצעות קבלה של הודעות נתוני הטיסה המתקבלות מאפליקציית הטיסן דרך השרת. בנוסף, מחלקה זו מבצעת שליחה של פקודות הטיסה התלויות במצב הג'ויסטיק אל אפליקציית הטיסן דרך השרת.

SerialVerification

מחלקה זו היא מחלקה המציגה לנו באופן גרפי כפתור ותיבת טקסט שבה אנו מכניסים את קוד האימות שקיבלנו מאפליקציית הטיסן וע"י הכפתור מבצעים אימות למספר זה. מחלקה זו מיישמת את המודות אשר שולחות את קוד האימות לשרת ומציגות את ההודעה המתאימה למשתמש בעת קבלת תוצאת האימות.

MapScreen

מחלקה זו היא מחלקה המציגה לנו את מסך המפה של האפליקציה בו ניתן לראות את המפה של גוגל. על מפה זו מוצג לנו סמן עם אייקון של טיסן המייצג את מיקום הטיסן בזמן אמת על גבי המפה. מחלקה זו מיישמת מתודות המבצעות קבלה והצגה של נתוני הטיסה, לחיצה על מיקום במפה לקביעת יעד הטיסה והפעלת הטיסה האוטומטית ע"י שליחת הודעה לאפליקציית השרת דרך השרת.

תוכנת השרת

תוכנת השרת בנויה מחבילות המכילות מספר רב של מחלקות ומתודות המהוות חלק חשוב בפעולותיה של התוכנית ועל כן נפרט.

ActivityDBManager

מחלקה זו מיישמת מתודות המבצעות את הפעולות השונות מול בסיס הנתונים, כגון: הוספת רשומה חדשה, עדכון רשומה קיימת או קריאת רשומה.

ConnectionManagerST

מחלקה זו מיישמת מתודות המטפלות ביצירת התנתקות והתקשרות לבסיס הנתונים MySQL ע"י תבנית Singletone אשר נועדה למקרים בהם מעוניינים להגביל את יצירת המופעים של מחלקה מסוימת למופע יחיד. כלומר, בתוכנת השרת אנחנו יוצרים מופע יחיד המתקשר עם בסיס הנתונים ואיתו אנו עובדים לאורך כל פעולת השרת.

Controller

מחלקה זו מיישמת מתודות המטפלות בהודעות המתקבלות מאפליקציית הטיסן ואפליקציית המשתמש. המחלקה מאפשרת את שליחת ההודעה הנכונה לצד המבקש את המידע. בנוסף, ההודעות הרלוונטיות לנתוני הטיסה של הטיסן מועברות ונשמרות בבסיס הנתונים ומוצגות באופן דינאמי בממשק המשתמש.

ClientThread

מחלקה זו יורשת ממחלקת Threadכדי לאפשר למחלקה ליישם מתודות המאפשרות יצירה של Thread ולטפל בכל הודעה שהשרת מקבל מאפליקציות הטיסן והמשתמש. במקרה שה- Thread סיים את פעילות ההודעה שהתקבלה בשרת, ה- Thread מסיים את ריצתו. בנוסף, מחלקה זו מציגה בממשק המשתמש את ההודעה שכרגע נמצאת בטיפול.

Server

מחלקה זו יורשת ממחלקת Thread, זאת על מנת לאפשר למחלקה ליישם מתודות המאפשרות יצירה של Thread עבור השרת שמטרתו לעבוד כל עוד לא הפסקנו את פעולתו ע"י הפקד Stop Server. שרת נקרא גם שרת Multithread מפני שעבור כל הודעה המתקבלת בשרת, השרת יוצר ClientThread חדש. פעילות בצורה זו מאפשרת לעבוד במקביל על כל ההודעות המתקבלות בשרת במקום לעבוד על כל הודעה בנפרד, דבר שמאט את פעילות המערכת.

ServerGui

מחלקה זו היא מחלקה המיישמת את כל המתודות המציגות לנו באופן גרפי את ממשק המשתמש של השרת המציג לנו חלון שבו ניתן לראות את ההודעות העוברות דרך השרת, תפריט שבו ניתן להפעיל ולהפסיק את פעילות השרת וסרגל ימני בו ניתן לראות את כל נתוני הסנסורים של הטלפון המפעיל את אפליקציית הטיסן.

רכיב ה ADK

קוד ה ADK בנוי מקובץ יחיד בשם adkUsbConnection הכתוב בשפת C ומיישם את המתודות הבאות:

AdbEventHandler

מתודה זו מיישמת את פעולות השרת המקבל את ההודעה מאפליקציית הטיסן דרך התקן ה- USB ובהתאם להודעה שהתקבלה מחליט איזו פקודה לשלוח לטיסן דרך רכיב ה- Wi-Fly. בנוסף, כאשר ה- USB מחובר למכשיר הטלפון ונוצר קשר בין הרכיב (השרת) למכשיר הטלפון, מציג השרת את ההודעה ADB_CONNECTIO_OPEN ובעת ניתוק מחבר ה- USB, מציג השרת את ההודעה ADB_CONNECTION_CLOSE.

DataLed

מתודה זו מיישמת הדלקה וכיבוי של לד הנמצא על גבי רכיב ה- ADK בהשעיה של 300ms על כל הודעה הנשלחת מרכיב ה ADK למכשיר הטלפון או לטיסן דרך רכיב ה Wi-Fly. אופציה זו מאפשרת לנו בדיקה וויזואלית כי ההודעות אכן נשלחות לרכיבים השונים.

SendAndroid

מתודה זו מיישמת שליחת הודעה למכשיר הטלפון בגודל 2 בתים המכיל את המספר 30 ומייצג כי בוצע אימות תקשורת בין ה- ADK למכשיר הטלפון והתקשורת תקינה.

Setup

מתודה זו היא מתודה ברירת מחדל החייבת במימוש. במתודה זו מבוצע אתחול השרת לפרוטוקול TCP על פורט 4568 ואתחול התקשורת מול רכיב ה- Wi-Fly בעזרת תקשורת סריאלית בקצב 9600.

Loop

מתודה זו היא מתודה ברירת מחדל החייבת במימוש. במתודה זו מבוצעת הפעלת השרת ובדיקה האם התקבלה הודעה מרכיב הטלפון לאימות התקשורת עם הרכיב.

רכיב ה- Wi-Fly

קוד ה- Wi-Fly בנוי מקובץ יחיד בשם wifly הכתוב בשפת C ונבנה במטרה לאפשר למשתמש להגדיר את רכיב ה- Wi-Fly באופן ראשוני ע"י פקודות command line דרך serial monitor של סביבת העבודה Arduino. קוד זה מיישם את המתודות הבאות:

בנוסף, **בקובץ זה מוסבר אופן החיבור של רכיב ה- Wi-Fly לרכיב ה- ADK.**

Setup

מתודה זו היא מתודה ברירת מחדל החייבת במימוש. במתודה זו מבוצע אתחול התקשורת מול רכיב ה- Wi-Fly בעזרת תקשורת סריאלית בקצב 9600 ואתחול התקשורת מול המחשב בעזרת תקשורת סריאלית בקצב 9600. עבור כל תקשורת ישנו רכיב סריאלי שונה.

Loop

מתודה זו היא מתודה ברירת מחדל החייבת במימוש. במתודה זו מבוצעת כתיבת וקריאה של ההודעות מהרכיבים הסריאליים המאפשרות את הגדרת רכיב ה- Wi-Fly.

אופן החיבור של רכיב ה Wi-Fly לרכיב ה ADK:

Wi-Fly	ADK
Pin 1 – 3.3V	3.3V
Pin 2 – TX	Pin 15 – RX
Pin 3 – RX	Pin 14 – TX
Pin 10 – GND	GND

- ❖ החיבור מבוצע ע"י כבלי נחושת פשוטים.

אתר הפרויקט

אתר הפרויקט בנוי ממספר קבצים ומהשפות HTML5, PHP, JavaScript, Ajax, CSS3. בדף זה נסביר מה מטרת כל קובץ באתר הפרויקט ובאיזה שפה הוא נכתב. באתר הפרויקט ישנן גם שתי תיקיות נוספות: תיקיית images המכילה את תמונות האתר ותיקיית api המכילה שלושה אתרים שנוצרו באופן אוטומטי ע"י JavaDoc ומציגות את ה- api של אפליקציית המשתמש, אפליקציית הטיסן ותוכנת השרת.

GetStatus.php

קובץ זה יוצר קשר עם בסיס הנתונים MySQL ומושך מהטבלה את כל נתוני הטיסה של הטיסנים הנמצאים כרגע על גבי המפה של גוגל. הנתונים שהתקבלו מבסיס הנתונים נשמרים במערך ומועברים לטיפול קובץ mapScript.js ע"י הדפסת המערך בייצוג של קובץ json. קובץ זה נכתב בשפת PHP וניתן למצוא אותו בתיקיית php הנמצאת באתר הפרויקט.

MapScript.js

קובץ זה מכיל מספר פונקציות המאפשרות את הצגת המפה של גוגל על גבי canvas וביצוע קריאה לקובץ getStatus.php בכל 10 שניות על מנת לקבל את הנתונים העדכניים של כל הטיסנים בכדי שניתן יהיה להציג כל טיסן ע"י מרקר על גבי המפה וכאשר נלחץ על הטיסן (מרקר) הרצוי, יוצגו לנו נתוני הטיסה של הטיסן אותו בחרנו. קובץ זה נכתב בשפת JavaScript , למעט הצגת נתוני הטיסה המוצגים ע"י שימוש בשפת Ajax. קובץ זה ניתן למצוא בתיקיית JavaScript הנמצאת באתר הפרויקט.

בתיקייה JavaScript ישנה תיקייה נוספת בשם anchor המכילה תוסף קוד פתוח הכתוב גם הוא בשפת javascript ומאפשר לנו גלילה חלקה של האתר למקום רצוי ע"י לחיצה על כפתור.

Style.css

קובץ זה מכיל את עיצוב האתר. עיצוב האתר כולל את העיצוב של הטקסט, הכפתורים, הכותרות, הקישורים, התמונות, הרקעים, התפריט, מפת גוגל והצגת נתוני הטיסה. קובץ זה נכתב בשפת CSS3 וניתן למצוא אותו בתיקיית Style הנמצאת באתר הפרויקט.

Index.html

קובץ זה מכיל את הקריאות לקבצים השונים על מנת שיבצעו את פעולתם באתר. כמו כן קובץ זה מכיל את הגדרת הכפתורים, הקישורים, חלוקת הדף למקטעים, התפריט, הצגת המפה של גוגל ואת התוכן המוצג באתר. קובץ זה נכתב בשפת HTML5 וניתן למצוא אותו בתיקייה AA-UAV התיקייה של אתר הפרויקט.

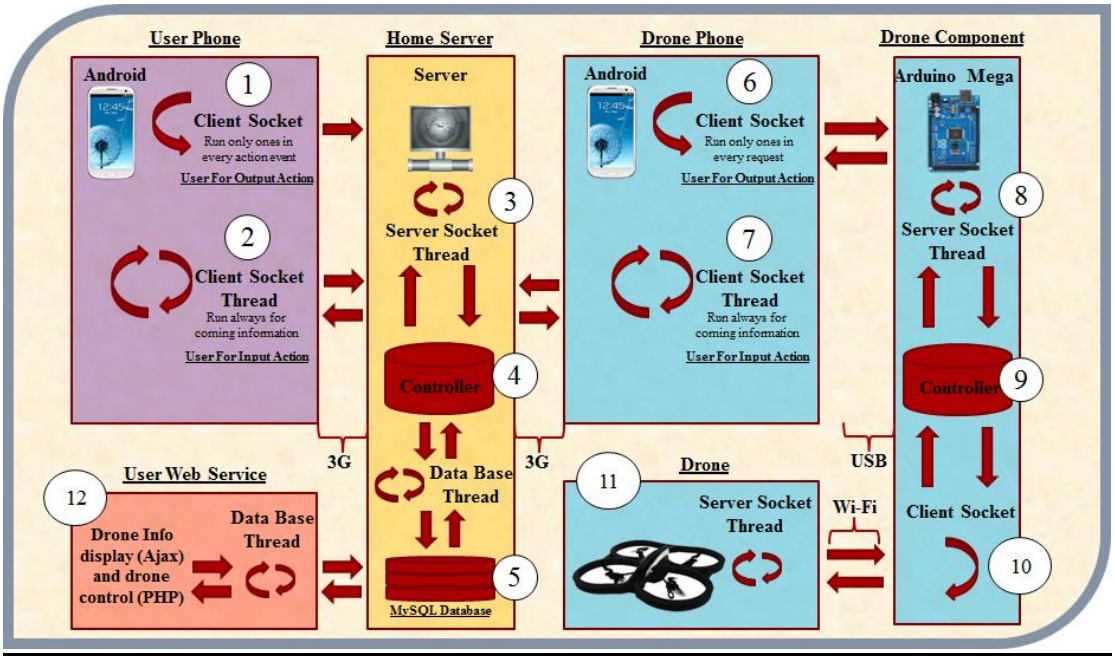
מבנה ה- DataBase

בסיס הנתונים aauavdb בנוי מטבלה אחת בשם droneinfo וטבלה זו מכילה שדות המכילים את נתוני הטיסה של כל הטיסנים. שדות אלה מתעדכנים בזמן אמת כל 10 שניות, ובכך מאפשרות לנו להציג את הנתונים הכי עדכניים והמיקום הנוכחי של כל טיסן על גבי המפה של גוגל הנמצאת באתר הפרויקט.

פירוט השדות של טבלת droneinfo

שם השדה	הסבר
Id	מספור אוטומטי הגדל ב-1 ומציין את מספר השורה.
Droneid	מספר רנדומלי בן 4 ספרות המיוצר באפליקציית הטיסן ומייצג את המספר שאיתו יש לאמת את הטיסן באפליקציית המשתמש.
City	שם העיר בה נמצא הטיסן
Height	גובה הטיסן
Speed	מהירות הטיסן
Startpointx	נקודת ציון התחלתית של הטיסן בציר ה X
Startpointy	נקודת ציון התחלתית של הטיסן בציר ה Y
Currentpointx	נקודת ציון נוכחית של הטיסן בציר ה X
Currentpointy	נקודת ציון נוכחית של הטיסן בציר ה Y
Endpointx	נקודת ציון סופית של הטיסן בציר ה- X מייצג את נקודת הציון אליה צריך להגיע הטיסן במצב של טייס אוטומטי.
Endpointy	נקודת ציון סופית של הטיסן בציר ה-Y מייצג את נקודת הציון אליה צריך להגיע הטיסן במצב של טייס אוטומטי.
Distance	מרחק הטיסה בין נקודת הציון ההתחלתית של הטיסן לבין נקודת הציון הסופית של הטיסן (מתעדכן רק במצב טייס אוטומטי בו נקבעה נקודת ציון סופית)
Distancemade	מרחק הטיסה בין נקודת הציון ההתחלתית של הטיסן לבין נקודת הציון הנוכחית של הטיסן
Dbattary	מצב סוללת הטיסן באחוזים
Pbattary	מצב סוללת מכשיר הטלפון הנמצא על הטיסן באחוזים
Dtype	סוג הטיסן (משחק, משטרה, העברה). כל סוג מאפשר תכונות שונות, למשל טיסן משטרה לא מוצג על גבי המפה של גוגל. בפרויקט זה מומש רק סוג הטיסן "משחק", כל שאר הסוגים הם עתידיים.
Status	מצב הטיסן כלומר, האם הטיסן פעיל או מכובה.

תהליכי המערכת



בפרויקט זה תהליכי המערכת תלויים ברכיבים השונים המרכיבים אותה. באיור זה ניתן לראות את תהליכי המערכת העיקריים בכל רכיב ורכיב על מנת שהמערכת כולה תעבוד. תהליכי המערכת המוצגים פה הם בעיקר תהליכים הקשורים לתקשורת בין הרכיבים השונים. כעת נסביר את התהליך של כל רכיב במערכת הפרויקט.

User Phone

(1) Client Socket

תהליך זה הוא לקוח השולח לשרת הבית את הודעות הפעולה השונות (כגון: Take off, Land) ובכל פעולה שהמשתמש מבקש לעשות, התהליך מתבצע פעם אחת על מנת לשלוח את ההודעה לשרת ולאחר מכן מפסיק את פעולתו.

(2) Client Socket Thread

תהליך זה הוא לקוח השולח לשרת הבית הודעות המבקשות לקבל את נתוני הטיסה העדכניים של הטיסן. תהליך זה מתבצע ללא הפסקה ושולח הודעה בקשה כל 10 שניות.

Home Server

(3) Socket Server Thread

תהליך זה הוא שרת הבית, שרת רב תהליכים כלומר, תהליך השרת מתבצע ללא הפסקה ומחכה לבקשות הלקוחות (אפליקציית המשתמש והטיסן), בכל פעם שהשרת מקבל בקשה מלקוח הוא יוצר תהליך חדש עבור אותו לקוח, תהליך הלקוח מתבצע פעם אחת בלבד.

(4) Controller

לאחר שהבקשה של הלקוח מגיעה לשרת הבית, השרת רוצה לדעת איזו תשובה להחזיר לאותו לקוח ולכן השרת שולח את הבקשה ל- controller. ה- controller לוקח את הבקשה ובודק מאיזו אפליקציה היא הגיעה (משתמש או טיסן). לאחר מכן בודק מה תוכן הבקשה ובהתאם מחזיר את התשובה הרצויה לשרת והשרת שולח את התשובה הרצויה ללקוח.

(5) Data Base Thread

בהרצה הראשונית של שרת הבית, נוצר מופע מהמחלקה ConnectionManagerST המבצע קישור לבסיס הנתונים. מרגע זה הקישור פעיל לאורך כל פעולתו של השרת ולכן הוגדר על ידינו כתהליך. תהליך זה בא לשימוש כאשר מתבצע אימות בין אפליקציית הטיסן לאפליקציית המשתמש ויש צורך לגשת לבסיס הנתונים על מנת לאמת את קוד הטיסן.
בנוסף, תהליך זה בא לידי שימוש כאשר מגיעים נתוני הטיסה לשרת אותם אנו רוצים לשמור בבסיס הנתונים על מנת שאתר הפרויקט יוכל להציג אותם.

Drone Phone

(6) Client Socket

תהליך זה הוא לקוח השולח לשרת ה- ADK את הודעות הפעולה השונות (כגון: Take off, Land) אשר התקבלו מאפליקציית המשתמש דרך השרת, התהליך מתבצע פעם אחת על מנת לשלוח את ההודעה לשרת ולאחר מכן מפסיק את פעולתו.

(7) Client Socket Thread

תהליך זה הוא לקוח השולח לשרת הבית הודעות המבקשות לקבל את הפעולות השונות (כגון: Take off, Land) אשר התקבלו מאפליקציית המשתמש דרך השרת, תהליך זה מתבצע ללא הפסקה ושולח הודעה בקשה כל 1 שניה.

Drone Component

(8) Socket Server Thread

תהליך זה הוא שרת ה- ADK. תהליך שרת זה מתבצע ללא הפסקה ומחכה להודעות הפעולה מאפליקציית הטיסן, כל הודעה שמתקבלת בשרת זה מתבצעת פעם אחת בלבד. כאשר הלקוח שולח הודעה הדורשת תשובה, שרת זה מפעיל בעזרת ה- controller מתודה המספקת את התשובה הרצויה. דוגמא להודעה החייבת בתשובה: בדיקת חיבור ה- USB בין מכשיר הטלפון לרכיב ה ADK.

(9) Controller

לאחר קבלת הודעה מאפליקציית הטיסן בשרת ה- ADK, השרת רוצה לדעת איזו פקודה לשלוח לרכיב ה- Wi-Fly והאם זו הודעה הדורשת תשובה ולכן מעביר את ההודעה ל- Controller. ה-controller מבוצע במתודה הנקראת AdbEventHandler והוא יודע לקחת את ההודעה שהתקבלה ובמידה והודעה זו דורשת תשובה ה- controller מפעיל מתודה הנקראת SendAndroid השולחת תשובה למשתמש בייצוג של מספר באורך 2 בתים. במידה וההודעה היא הודעה המכילה פעולה שאותה הטיסן צריך לבצע, ה- Controller יודע להמיר את ההודעה המיוצגת כמספר באורך 2 בתים להודעת AT הרצויה ושליחתה לרכיב ה- Wi-Fly דרך Serial בקצב 9600.

(10) Client Socket

תהליך זה הוא תהליך שמבוצע באופן אוטומטי ברכיב ה- Wi-Fly. ברגע שרכיב ה- Wi-Fly מקבל את הודעת הפעולה מרכיב ה-ADK, הוא יוצר Client Socket מול הטיסן ושולח לו את הודעת הפעולה. תהליך זה מתבצע פעם אחת עבור כל הודעה המתקבלת ברכיב. תהליך זה הוא חלק מתוכנית הרכיב ולכן קוד זה הוא לא שלנו.

Drone

(11) Server Socket Thread

הטיסן עצמו מכיל רכיב תקשורת המפיץ רשת בדומה לנתב ומכיל קוד פתוח המבצע את פעולות הטיסן, חלק מקוד זה מכיל שרת המקשיב לפורטים (5554,5555,5556,5557) בפרוטוקול UDP. כל פורט מיועד לשימוש אחר כגון שליטה על הטיסן והצגת וידיאו. בפרויקט זה השתמשנו בפורט 5556 להעברת הודעות השליטה מרכיב ה- Wi-Fly לטיסן.

User Web Service

(12) Data Base Thread

אתר הפרויקט מכיל את המפה של גוגל ועל מפה זו מוצגים הטיסנים הפעילים וניתן לראות את התקדמותם על המפה בזמן אמת. לחיצה על טיסן מציגה לנו את נתוני הטיסה שלו בזמן אמת. פעולות אלו מתאפשרות לנו בזכות קוד המבוצע בקובץ GetStatus.php המדמה תהליך שרץ באופן קבוע כל 1 שניה. תהליך זה ניגש לבסיס הנתונים ומושך ממנו את נתוני הטיסה של כל הטיסנים הפעילים.

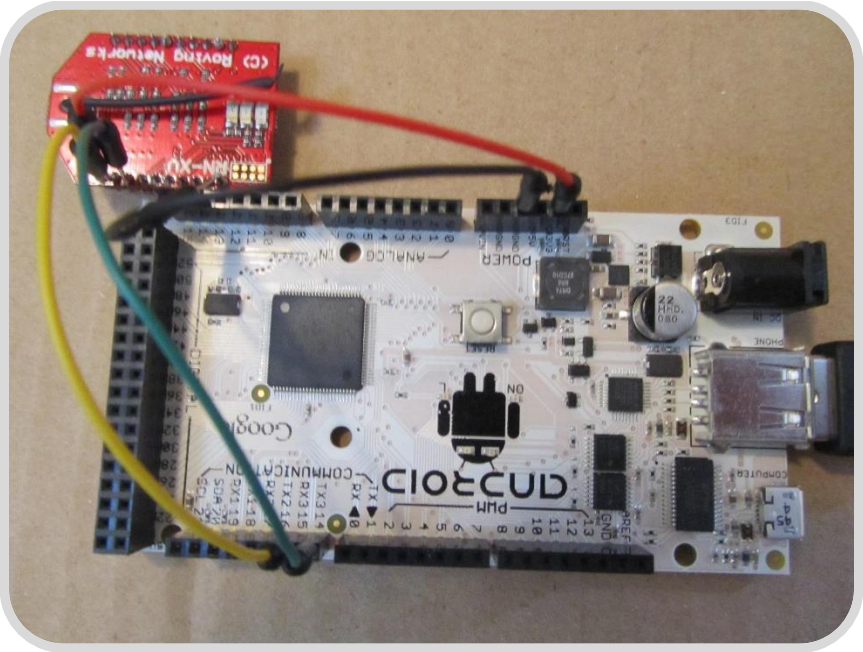
מדריך התקנה

התקנת המערכת מתחלקת לשני חלקים. בחלק הראשון נסביר כיצד לבצע את החיבורים הפיזיים בין כל הרכיבים, בחלק השני נסביר כיצד להתקין את האפליקציות של המשתמש והטיסן וכיצד להתקין את בסיס הנתונים ולהפעיל את תכנית השרת. את שני חלקים אלו נסביר בעזרת תמונות וצילומי מסך הממחישים את אופן החיבורים, התקנות האפליקציות, התקנת בסיס הנתונים והפעלת תוכנת השרת.

אופן חיבור הרכיבים

חיבור רכיב ה-Wi-Fly לרכיב ה-ADK

ראשית יש לחבר את רכיב ה-Wi-Fly לרכיב ה-ADK ע"י ארבעה כבלי נחושת פשוטים, רצוי שיהיו בצבעים שונים על מנת להבחין מה פעולתו של כל כבל. בתמונה זו ניתן לראות את רכיב ה-ADK בצע הלבן ואת רכיב ה-Wi-Fly בצבע האדום. משמעות צבעי הכבלים המחברים את הרכיבים: הצבע האדום מסמל "מתח", הצבע השחור מסמל "אדמה" והצבעים ירוק וצהוב מסמלים "TX" ו- "RX". מתחת לתמונה ניתן לראות את הטבלה המסבירה איפה כל כבר מתחבר ברכיב ה- ADK ואיפה כל כבל מתחבר ברכיב ה- Wi-Fly.

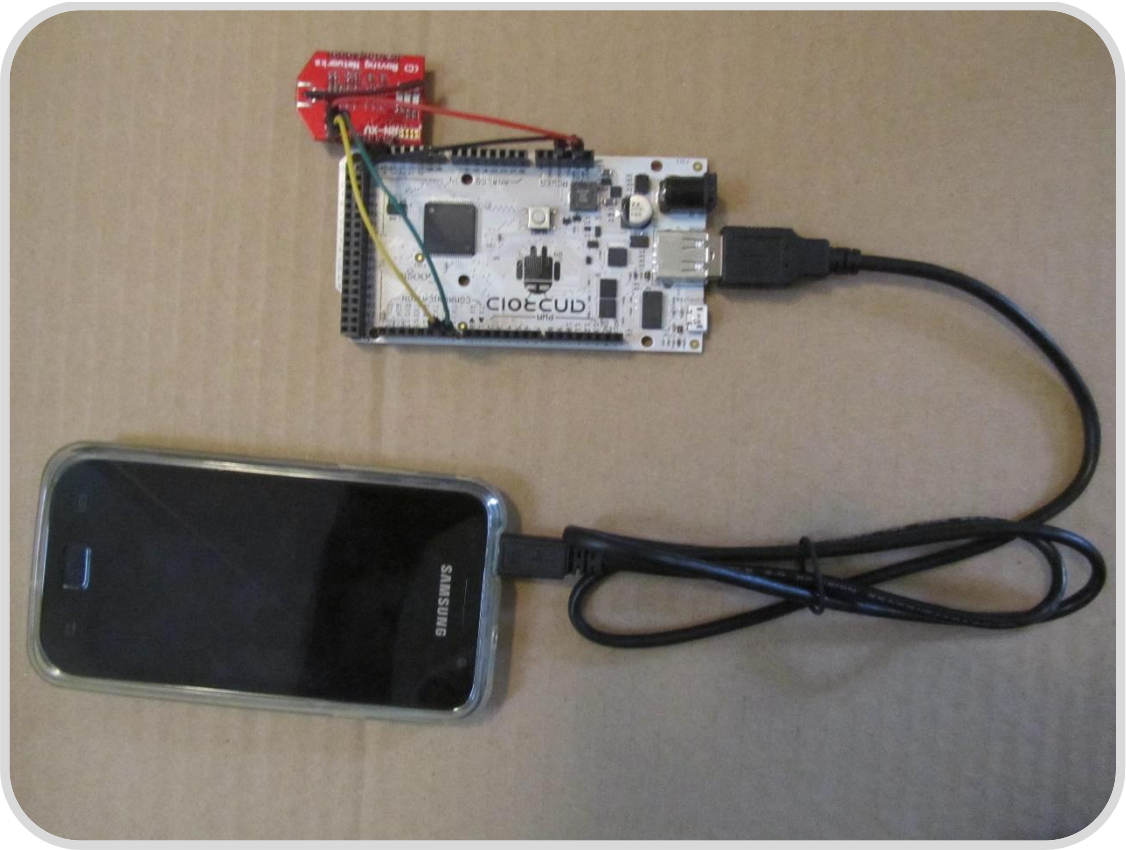


ADK	Wi-Fly
3.3V	Pin 1 – 3.3V
Pin 15 – RX	Pin 2 – TX
Pin 14 – TX	Pin 3 – RX
GND	Pin 10 – GND

כמו כן, ניתן לראות בתמונה זו את רכיב ה-ADK ועליו את המחברים השונים בקצה הימני. המחבר העליון הוא מחבר המספק מתח למכשיר ה- ADK, המחבר האמצעי הוא מחבר המאפשר תקשורת למכשיר אנדרואיד והמחבר התחתון הוא מחבר המתחבר למחשב המאפשר טעינה של קוד התוכנה לרכיב ה-ADK.

חיבור רכיב ה-ADK למכשיר הטלפון

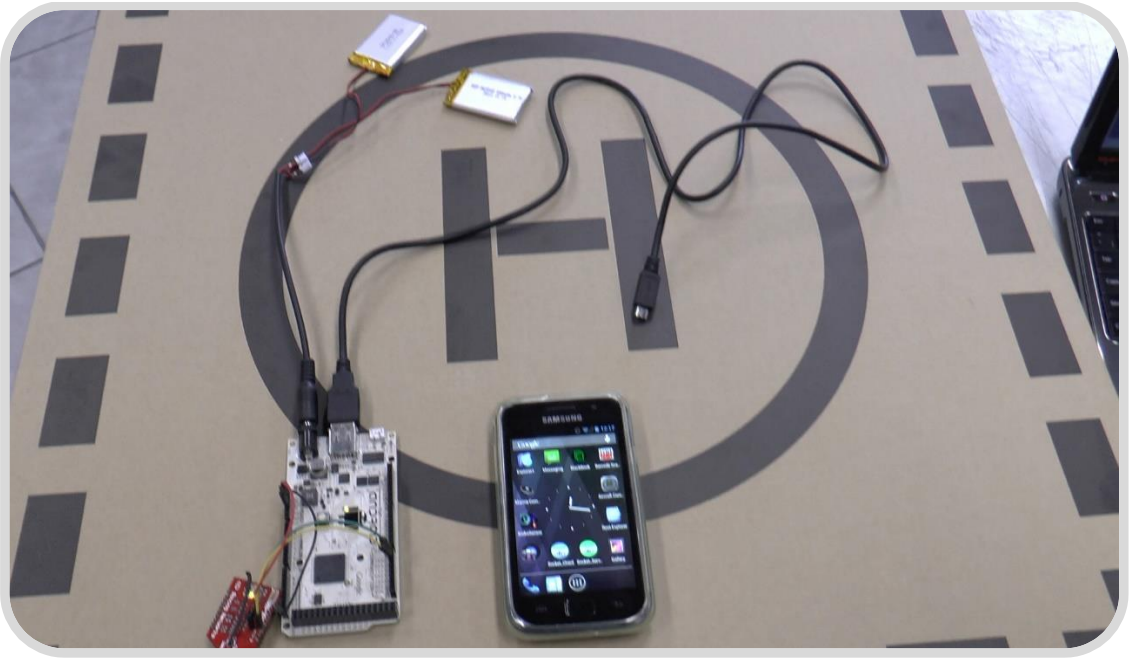
לאחר חיבור רכיב ה-Wi-Fly לרכיב ה-ADK, יש לחבר את רכיב ה-ADK למכשיר הטלפון ע"י כבל טעינה רגיל של מכשיר. יש לקחת את מחבר ה-USB הגדול של הכבל ולחבר אותו למחבר האמצעי של רכיב ה-ADK ולאחר מכן לקחת את מחבר ה-USB הקטן של הכבל ולחבר אותו למחבר הטעינה של מכשיר הטלפון. בתמונה זו ניתן לראות את אופן החיבור.



חיבור מתח חיצוני לרכיב ה-ADK

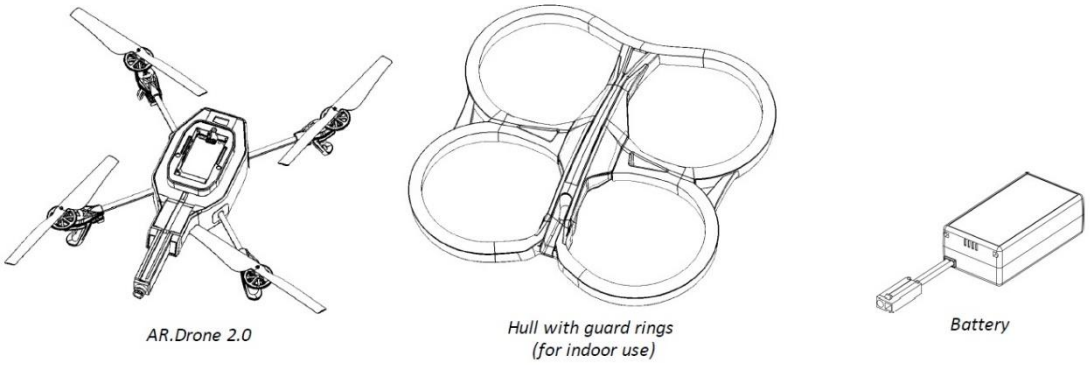
באופן עקרוני הרכיבים ADK, Wi-Fly והמכשיר הסלולרי צריכים להיות מותקנים בתוך הטיסן ולטוס אתו בזמן אמת. מכאן שרכיב ה-ADK זקוק למקור מתח ולכן בתמונה זו ניתן לראות כיצד חיברנו מתח חיצוני לרכיב ה-ADK.

על מנת לחבר מתח חיצוני לרכיב ה-ADK יש לקחת שני סוללות של 3.5V ולחבר אותן באופן טורי, כלומר "+" ל "-". את ה "+" וה- "-" הנותרים יש לחבר לכבל המספק את המתח למכשיר באופן הבא: "+" ל "+" ו "-" ל "-". בתמונה זו ניתן לראות כיצד כל זה מחובר וניתן לראות כי לקחנו כבל מתח עם מחבר המתאים למחבר הטעינה של רכיב ה -ADK, חתכנו אותו באמצע ואליו חיברנו את הסוללות.



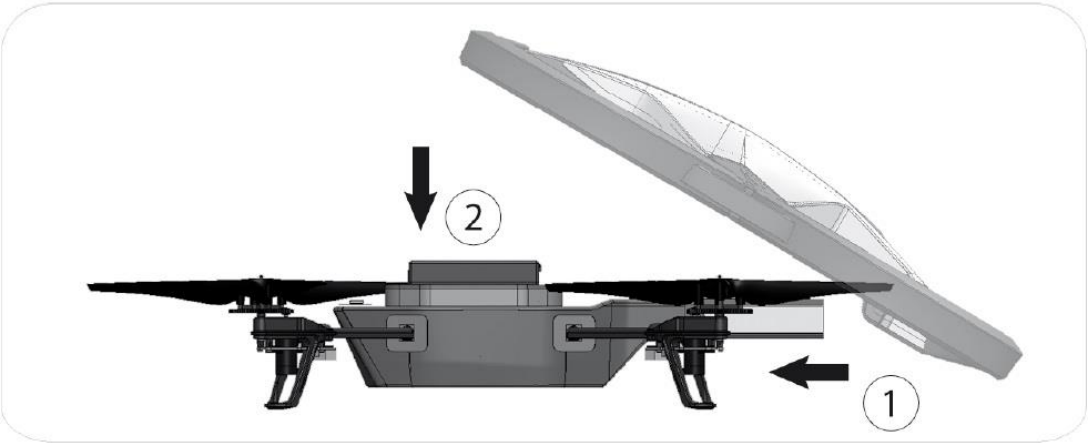
חיבור חלקי הטיסן

הטיסן מורכב משלושה חלקים. בתמונה זה ניתן לראות את הטיסן מצד שמאל, את מגן הטיסן באמצע ואת הבטרייה מצד ימין. את שלושת חלקים אלו יש לחבר ביחד למקשה אחת.



תחילה נחבר את הסוללה לטיסן על פי התמונה הבאה:

לבסוף נחבר את מגן הטיסן לטיסן על פי התמונה הבאה:



תוצאה סופית

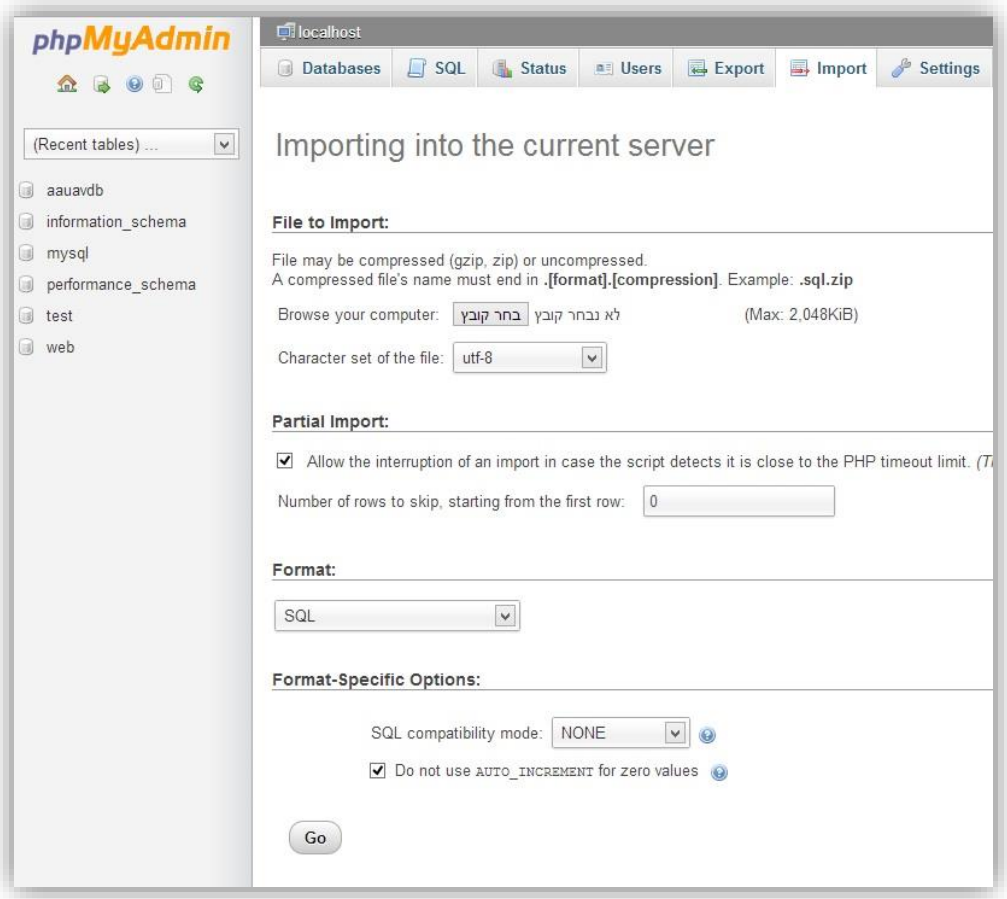
בתמונה זה ניתן לראות את כל החיבוריים הפיזיים בשלמותם:

אופן התקנת בסיס הנתונים

פרויקט זה מכיל בסיס נתונים. את בסיס הנתונים ניתן למצוא בתיקיית Database המצויה בדיסק המצורף לספר הפרויקט.

את בסיס נתונים זה ניתן להתקין בכל שרת. בפרויקט זה השתמשנו בשרת הביתי WampServer, שרת זה מכיל פלטפורמה בשם phpMyAdmin המאפשרת ניהול בסיסי נתונים. כעת נסביר כיצד להתקין את בסיס הנתונים בעזרת פלטפורמה זו.

על מנת להתקין את בסיס הנתונים יש להיכנס ל phpMyAdmin , לבחור בתפריט העליון את חוצץ "Import" ובחוצץ זה ללחוץ על כפתור "בחר קובץ" ולנתב לקובץ בסיס הנתונים בשם aauav.sql הנמצא בתיקיית Database המצויה בדיסק המצורף לספר הפרויקט. לאחר שבחרנו את הקובץ יש ללחוץ על הכפתור "Go", לאחר לחיצה על כפתור זה בסיס הנתונים של פרויקט זה יותקן ויוצג בסרגל השמאלי כבסיס נתונים בשם "aauavdb".



אופן התקנת האפליקציות

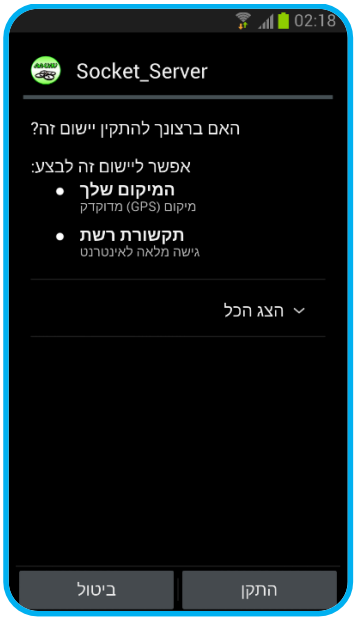
פרויקט זה מכיל שתי אפליקציות: אפליקציית משתמש ואפליקציית הטיסן. את שתי האפליקציות ניתן למצוא בתיקיית Applications המצויה בדיסק המצורף לספר הפרויקט.

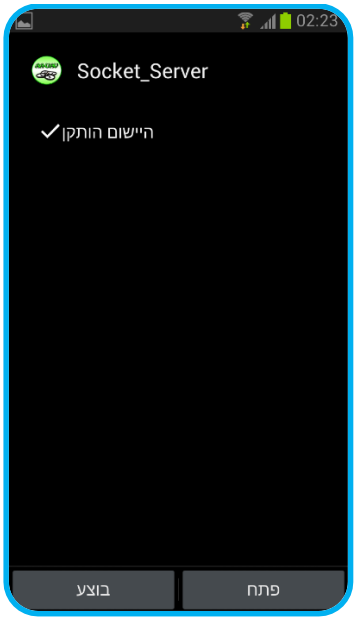
על מנת להתקין את אפליקציית המשתמש יש לקחת את הקובץ בשם UserApp.apk ולהעביר אותו למכשיר הטלפון דרך USB לתיקייה שאתם בחרתם הנמצאת במכשיר הטלפון. לאחר שהעברנו את הקובץ, יש לגשת לתיקייה שאליה העתקנו את קובץ האפליקציה דרך מכשיר הטלפון וברגע שהגענו לתיקייה וראינו את קובץ האפליקציה, יש ללחוץ עליו לצורך התקנה. לאחר הלחיצה יופיע מסך שישאל אותנו האם להתקין את האפליקציה, יש לבחור "כן" והמכשיר יתקין אוטומטית את האפליקציה. בסיום ההתקנה המכשיר ישאל אותנו האם נרצה להריץ את האפליקציה או לא. יש לבחור "כן" ונראה שהיא אכן עובדת.

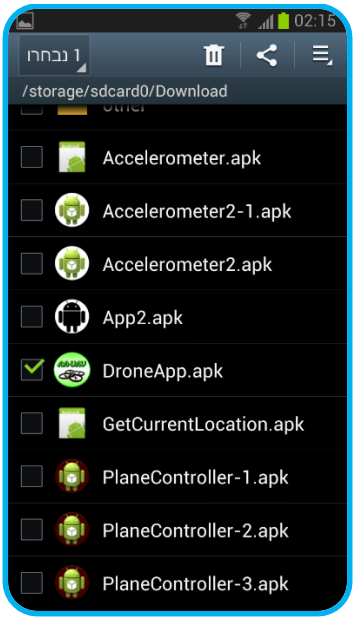
על מנת להתקין את אפליקציית הטיסן יש לבצע את אותן פעולות כמו שביצענו עבור אפליקציית המשתמש רק שהפעם נבחר בקובץ בשם DroneApp.apk.

לאפליקציות אלה יש אייקון זהה. על מנת להבדיל בין האפליקציות בחרנו צבע שונה עבור כל אפליקציה: צבע ירוק עבור אפליקציית הטיסן וצבע כחול עבור אפליקציית המשתמש.

את אופן התקנת האפליקציות על מכשיר הטלפון ניתן לראות בתמונות הבאות לפי סדר הצגתן משמאל לימין.





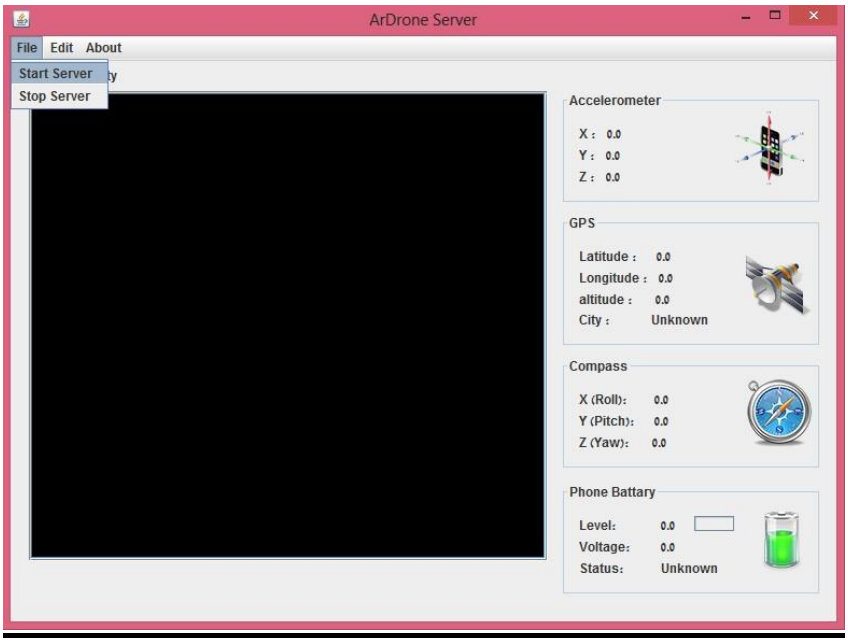


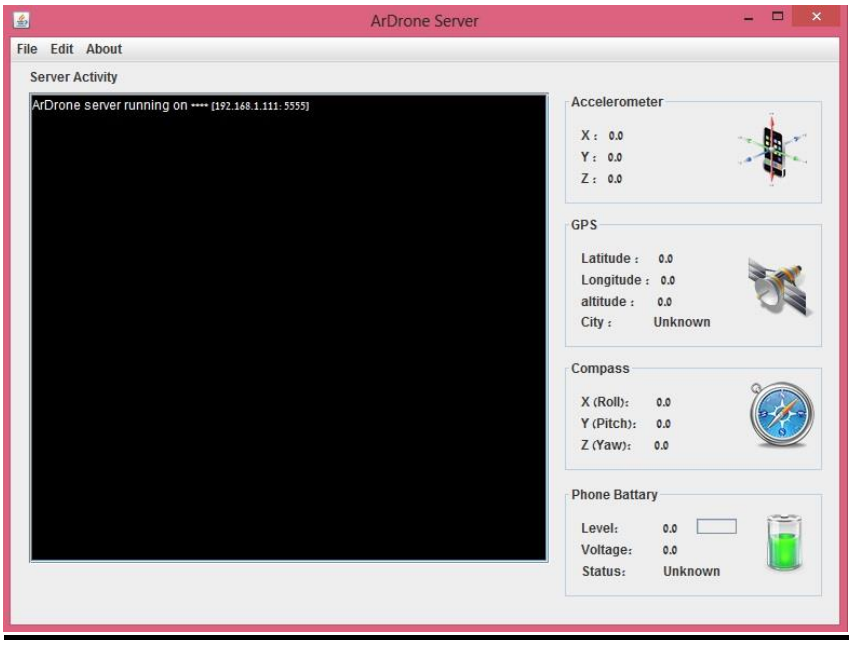
אייקון המשתמש

אייקון הטיסן

אופן הפעלת תוכנת השרת

פרויקט זה מכיל תוכנת שרת אותה ניתן למצוא בתיקיית ServerSoftware. על מנת להפעיל את תוכנת שרת זה, יש להיכנס לתיקייה וללחוץ לחיצה כפולה על קובץ בשם DroneServer.jar. לאחר הלחיצה הכפולה תופיע תוכנת השרת, בתוכנה זו ניתן לראות תפריט הנמצא בחלק העליון. על מנת להפעיל את השרת יש ללחוץ בתפריט זה על File ואז על StartServer. במידה ונרצה להפסיק את השרת נלחץ על File ואז על StopServer. לאחר לחיצה על StartServer אנחנו צריכים לראות הודעה בתיבת הטקסט של השרת שאומרת כי השרת פעיל. את אופן הפעלת תוכנת השרת ניתן לראות בתמונות הבאות:





נקודה חשובה: שרת זה עובד על פורט 5555 ולכן יש צורך לפתוח פורט זה בחומת האש של מערכת ההפעלה הרצה במחשב עליו מריצים את השרת. בנוסף, יש צורך לפתוח פורט זה בנתב הביתי המקשר את המחשב עליו מריצים את השרת לאינטרנט.

מדריך למשתמש

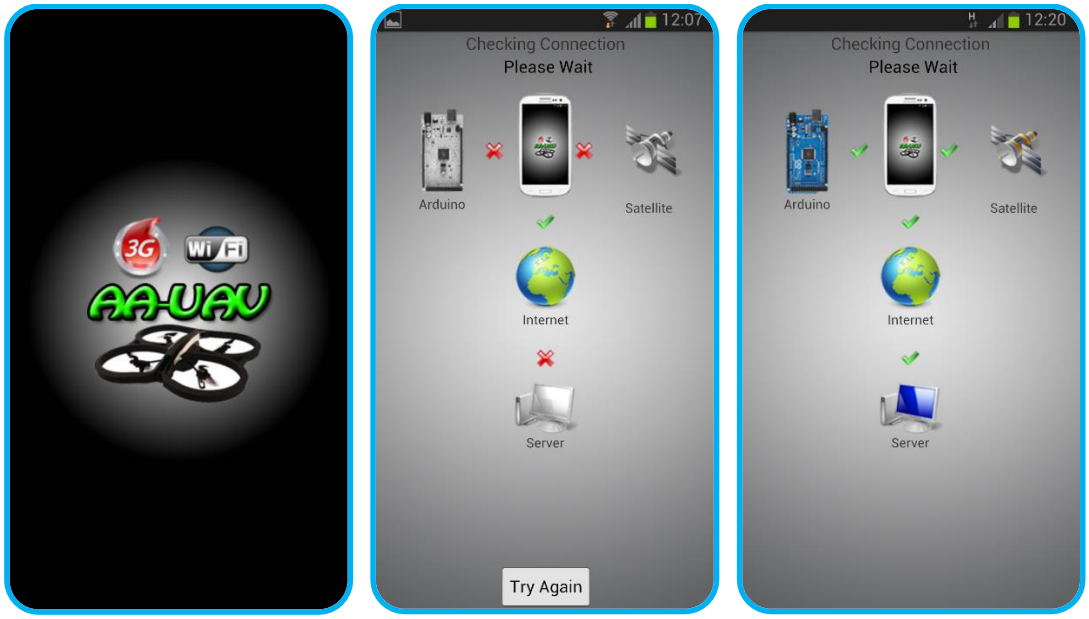
כעת, לאחר שהתקנו את כל הרכיבים הרצויים, כולל האפליקציות והפעלת השרת הראשי, ניתן להתחיל ולהשתמש באפליקציות על מנת להטיס את הטיסן באופן ידני או אוטומטי. מדריך זה יסביר לנו כיצד לעבוד עם אפליקציית המשתמש ואפליקציית הטיסן.

אפליקציית הטיסן

אפליקציית הטיסן נפתחת במסך פתיחה המציג את לוגו הפרויקט. לאחר מכן יש מסך הבודק את התקשורת עם כל הרכיבים איתם האפליקציה צריכה לעבוד. במידה וישנה בעיה בתקשורת לאחד הרכיבים, אנו נראה את הרכיב בצבע אפור. במצב כזה יש לוודא כי כל החיבורים תקינים, השרת פועל ויש מתח לרכיב ה ADK. לאחר מכן יש ללחוץ על הכפתור Try Again המצבע בדיקה חוזרת עבור כל הרכיבים. במידה ועדיין נתקלים בבעיה יש לעבור שוב על מדריך ההתקנה ולראות מה פספסנו.

במידה וכל חיבורי התקשורת תקינים, אנו נראה במסך הבדיקה את כל הרכיבים בצבע ולאחר מספר שניות נעבור למסך הבא (המסך הראשי של האפליקציה).

את אופן ריצת אפליקציית הטיסן על מכשיר הטלפון ניתן לראות בתמונות הבאות לפי סדר הצגתן משמאל לימין:



המסך הראשי של האפליקציה מציג את הלוגו של הפרויקט ואת מספר הזיהוי של הטיסן אותו צריך להכניס באפליקציית המשתמש לצורך אימות הטיסן. מסך זה הוא המסך הראשי ולכן אינו משתנה. במצב זה יש להשאיר את האפליקציה פתוחה ולא לכבות אותה.

לבסוף כאשר נסיים להטיס את הטיסן ונרצה לכבות את האפליקציה, ניתן ללחוץ על לחצן ה- Back במכשיר הטלפון והאפליקציה תציג לנו תיבת טקסט השואלת האם נרצה לצאת מהאפליקציה. במידה ונלחץ "Yes" האפליקציה תיסגר, במידה ונלחץ "No" היא תישאר פתוחה במסך הראשי.

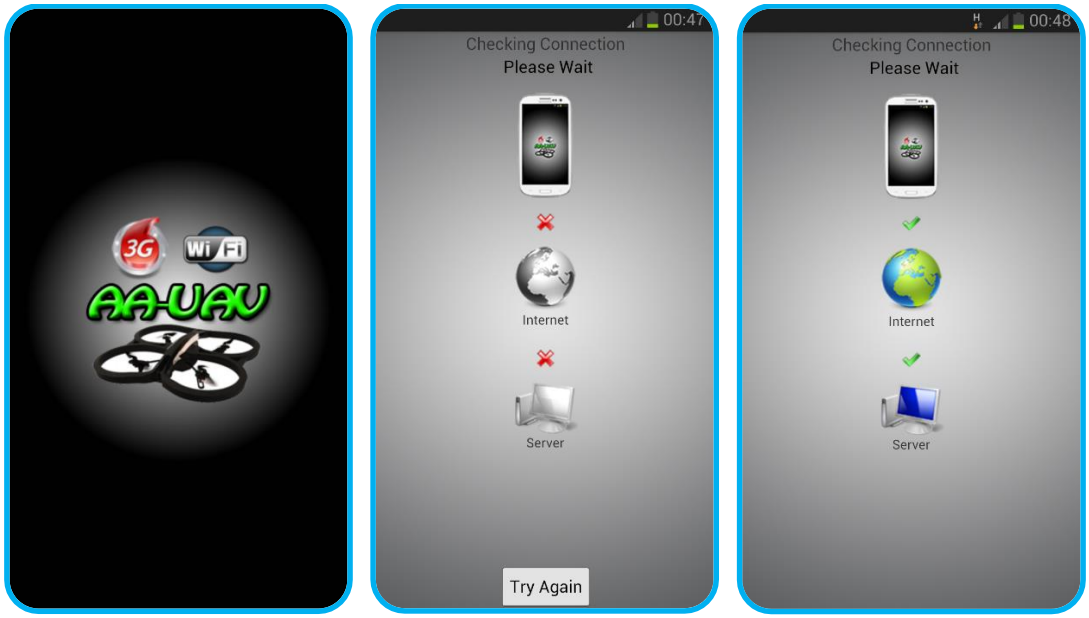
את אופן המשך ריצת אפליקציית הטיסן על מכשיר הטלפון ניתן לראות בתמונות הבאות לפי סדר הצגתן משמאל לימין:

אפליקציית המשתמש

אפליקציית המשתמש נפתחת במסך פתיחה המציג את לוגו הפרויקט. לאחר מכן יש מסך הבודק את התקשורת עם כל הרכיבים איתם האפליקציה צריכה לעבוד. במידה וישנה בעיה בתקשורת לאחד הרכיבים, אנו נראה את הרכיב בצבע אפור. במצב כזה יש לוודא כי כל החיבורים תקינים והשרת פועל, לאחר מכן יש ללחוץ על הכפתור Try Again המצבע בדיקה חוזרת עבור כל הרכיבים. במידה ועדיין נתקלים בבעיה יש לעבור שוב על מדריך ההתקנה ולראות מה פספסנו.

במידה וכל חיבורי התקשורת תקינים, אנו נראה במסך הבדיקה את כל הרכיבים בצבע ולאחר מספר שניות נעבור למסך הבא (מסך אימות הטיסן).

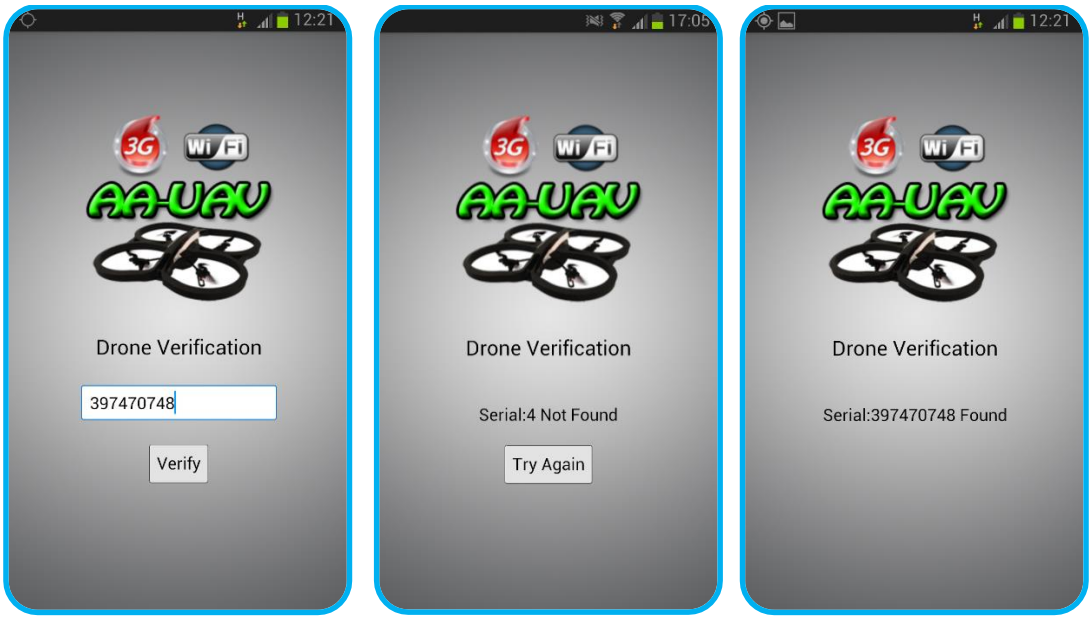
את אופן ריצת אפליקציית המשתמש על מכשיר הטלפון ניתן לראות בתמונות הבאות לפי סדר הצגתן משמאל לימין:



מסך האימות של הטיסן מציג לנו את לוגו הפרויקט ומתחתיו תיבת טקסט המשתמש להכנסת מספר הזיהוי של הטיסן, אותו קיבלנו מאפליקציית הטיסן. לאחר שהכנסנו את המספר יש ללחוץ על כפתור 'Verify' ואז תהליך האימות יעבוד. במידה ונמצא כי המספר שגוי, אנו נקבל מסך עם הודעה שאומרת כי המספר לא נמצא ומתחתיו כפתור 'Try Again' המחזיר אותנו למסך הקודם, שם ניתן להכניס שוב את המספר ולבדוק שנית.

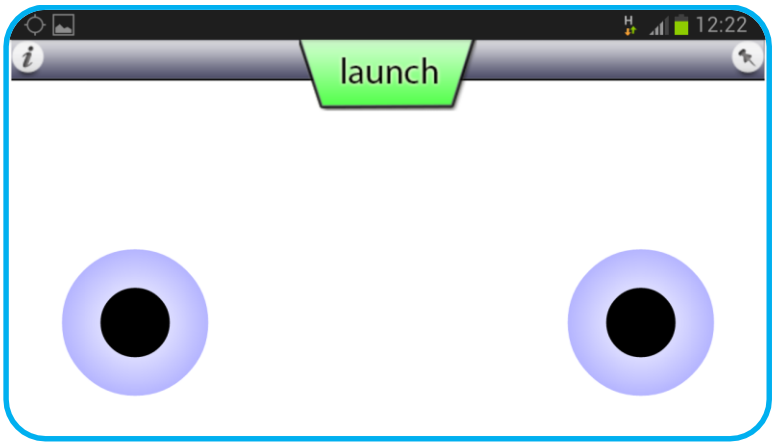
במידה והמספר נמצא תקין והאימות הצליח, תוצג לנו הודעה על גבי המסך כי המספר נמצא במערכת ולאחר מספר שניות נעבור למסך הבא (המסך הראשי של האפליקציה).

את אופן המשך ריצת אפליקציית המשתמש על מכשיר הטלפון ניתן לראות בתמונות הבאות לפי סדר הצגתן משמאל לימין:



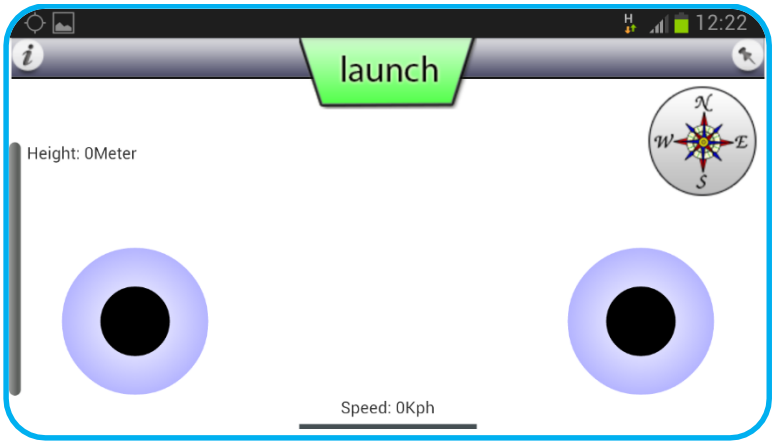
המסך הראשי

המסך הראשי מציג לנו את הג'ויסטיקים איתם אנו מטיסים את הטיסן ואת הסרגל העליון עליו ניתן לראות שלושה כפתורים: הכפתור השמאלי עבור הצגת נתוני הטיסה, הכפתור האמצעי עבור המראה ונחיתה של הטיסן והכפתור הימני עבור מעבר למסך טייס אוטומטי.



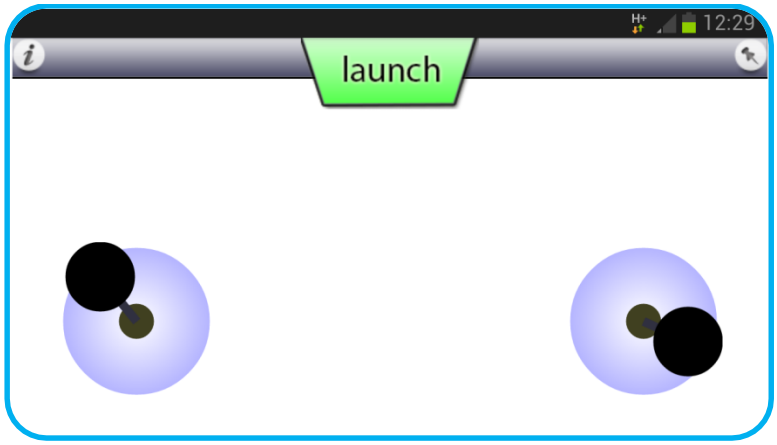
במידה ונלחץ על הכפתור השמאלי פעם אחת, יוצגו לנו נתוני הטיסה של הטיסן. במידה ונרצה להעלים את נתוני הטיסה מהמסך נלחץ שוב על הכפתור השמאלי פעם אחת.
נתוני הטיסה שאנו רואים על גבי המסך הם: מצד שמאל גובה הטיסן מעל פני האדמה במטרים, למטה באמצע מהירות הטיסן בקמ"ש ומצד ימין למעלה ניתן לראות מצפן המראה לנו את כיוון הטיסה של הטיסן. כל נתוני הטיסה מתעדכנים בזמן אמת.

במידה ונלחץ על הכפתור Launch, כפתור זה יחליף את צבעו לאדום וכיתובו ל-Landing, הטיסן ימריא לגובה של מטר ויחכה לפקודת טיסה.

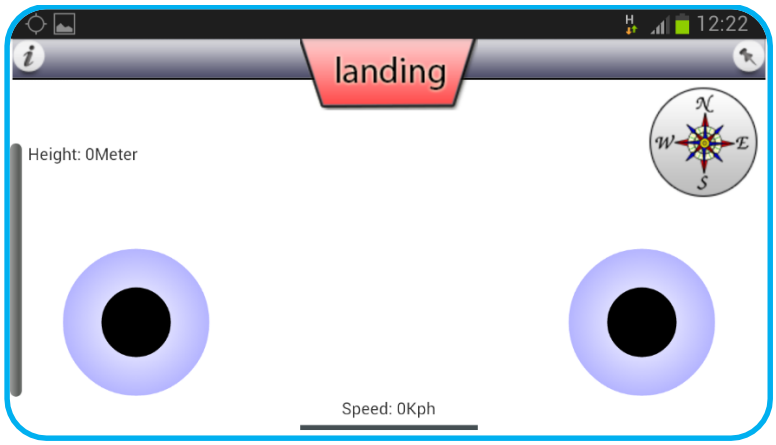


לאחר שגרמנו לטיסן להמריא והוא מחכה לפקודת הטיסה, ניתן להתחיל להטיס אותו ע"י הזזת הג'ויסטיקים. ניתן לראות שניתן להזיז את שני הג'ויסטיקים בו זמנית על מנת לאפשר תנועות (לדוג': להגביר מהירות ובאותו זמן לטוס למעלה). הג'ויסטיקים הם אלו ששולחים את פקודות הטיסה לטיסן ועל כן נסביר מה כל ג'ויסטיק מבצע.

	ג'ויסטיק ימני	ג'ויסטיק שמאלי
למעלה	טוס למעלה	טוס קדימה
למטה	טוס למטה	טוס אחורה
ימינה	טוס ימינה	הסתובב ימינה
שמאלה	טוס שמאלה	הסתובב שמאלה

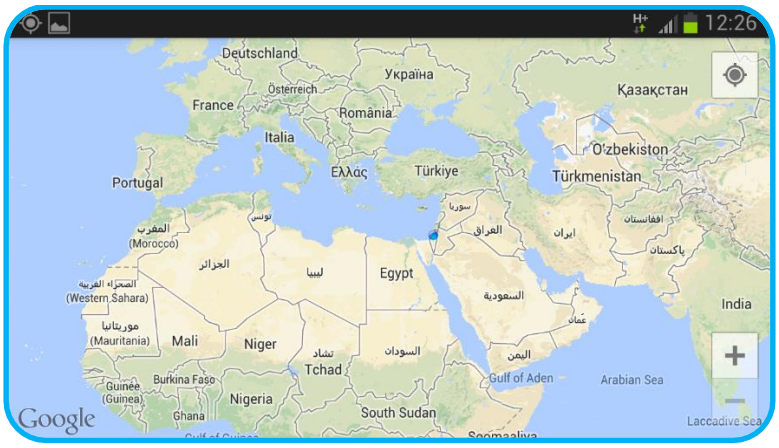


במידה ונלחץ על הכפתור Landing, כפתור זה יחליף את צבעו לירוק וכיתובו ל- Launch והטיסן יבצע נחיתה.

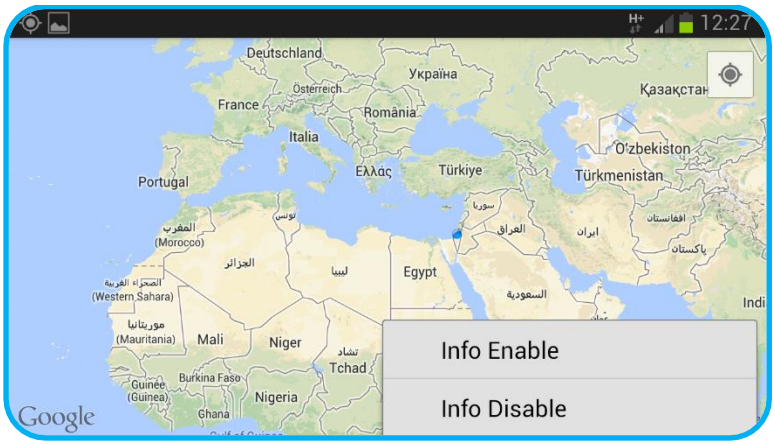


הכפתור האחרון במסך הראשי הוא הכפתור הימני עם סמל של נעץ. לחיצה על כפתור זה תעביר אותנו למסך המפה שם נוכל לבחור נקודת יעד אליו יטוס הטיסן באופן אוטומטי.

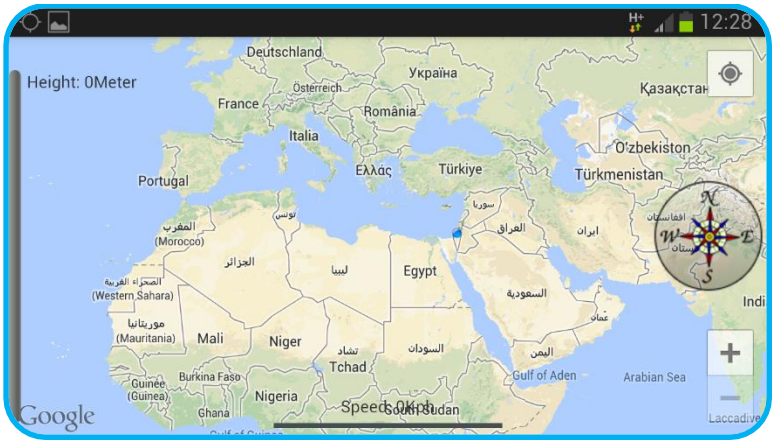
לאחר שלחצנו על הכפתור הימני במסך הראשי אנו נעבור מסך. במסך זה ניתן לראות את מפת גוגל ועליה את המיקום הנוכחי של הטיסן בצורת נקודה כחולה וקטנה.



במסך זה ניתן ללחוץ במכשיר הטלפון על כפתור התפריט. לאחר לחיצה על כפתור זה יוצג לנו תפריט בו נוכל לבחור האם להציג את נתוני הטיסה על המפה או להפסיק את הצגתם.

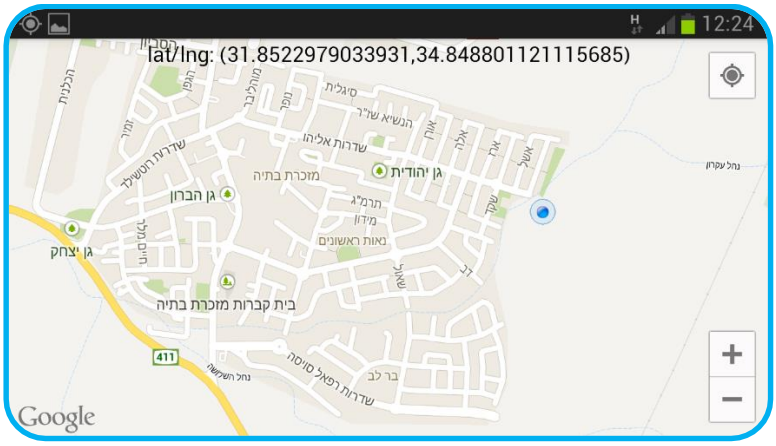


במידה ולחצנו על כפתור Info Enable יוצגו לנו נתוני הטיסה בצורה הבאה.

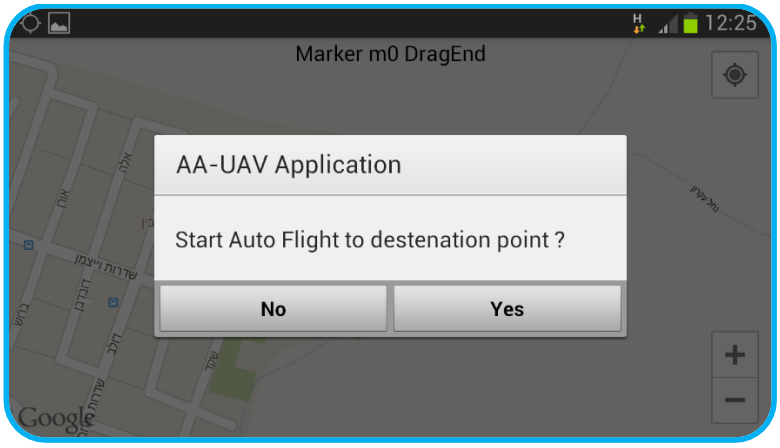


במידה ונלחץ על כל מקום במפה לחיצה אחת, תוצג לנו בחלק העליון של המפה נקודת ציון (נקודת אורך ורוחב) של המיקום עליו לחצנו.

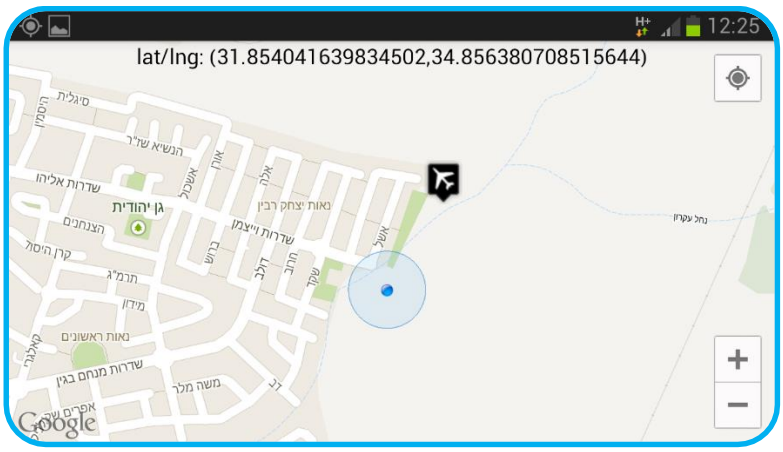
במידה ונלחץ על כפתור הג'י פי אס בצד ימין, האפליקציה תבצע זום אוטומטי למיקום בו נמצא הטיסן (כפי שניתן לראות בתמונה זו). מצד ימין למטה ישנם כפתורים המאפשרים לבצע זום ידני. ניתן ללחוץ על הכפתור "+" לצורך התקרבות או "-" לצורך התרחקות.



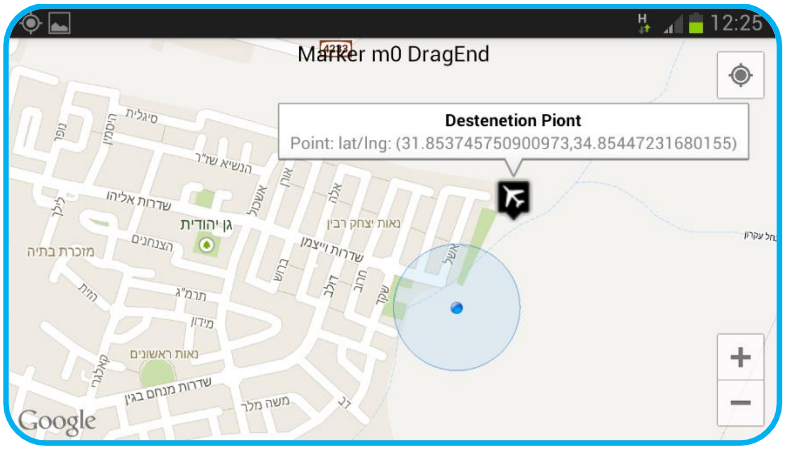
על מנת לקבוע נקודה יעד טיסה עבור הטיסן, נבצע לחיצה קבועה על גבי המפה במיקום בו אנו רוצים שהטיסן יטוס אליו. לאחר הלחיצה תופיע לנו תיבת טקסט שתשאל האם אנו מעוניינים שהטיסן יטוס לנקודת יעד זו. במידה ונלחץ "No", נוכל לבחור שוב נקודת יעד. במידה ונלחץ "Yes" יוצג לנו מיקום היעד שבחרנו עם אייקון של טיסן, במקביל הטיסן יקבל פקודת טיסה אוטומטית ומרגע זה הטיסן טס עצמאית ואין צורך להטיס אותו.



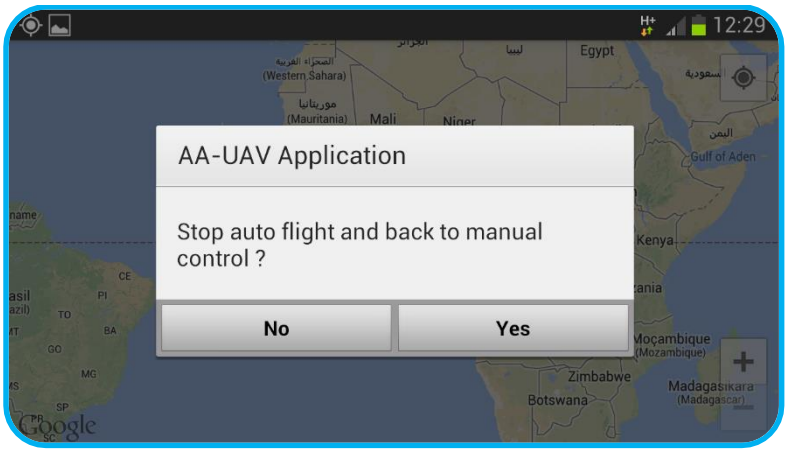
כפי שניתן לראות בחרנו נקודת יעד, אישרנו את היעד הרצוי וכעת רואים את נקודת היעד שלנו כאייקון של טיסן על גבי המפה.



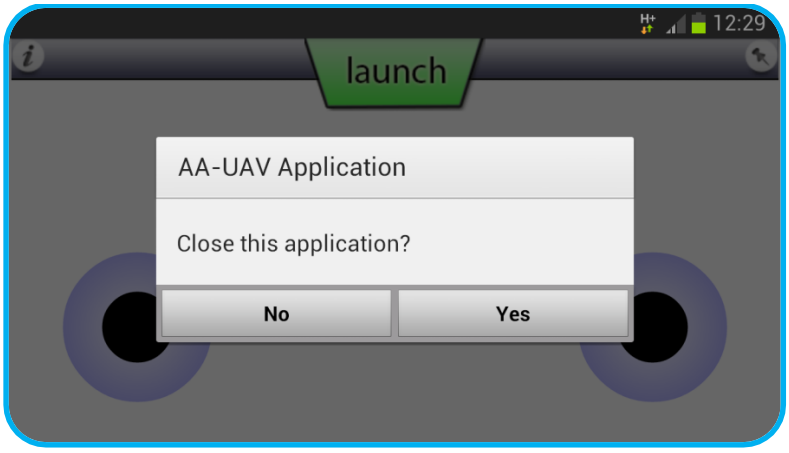
לחיצה על אייקון הטיסן תציג לנו את נקודת הציון של יעד הטיסה.



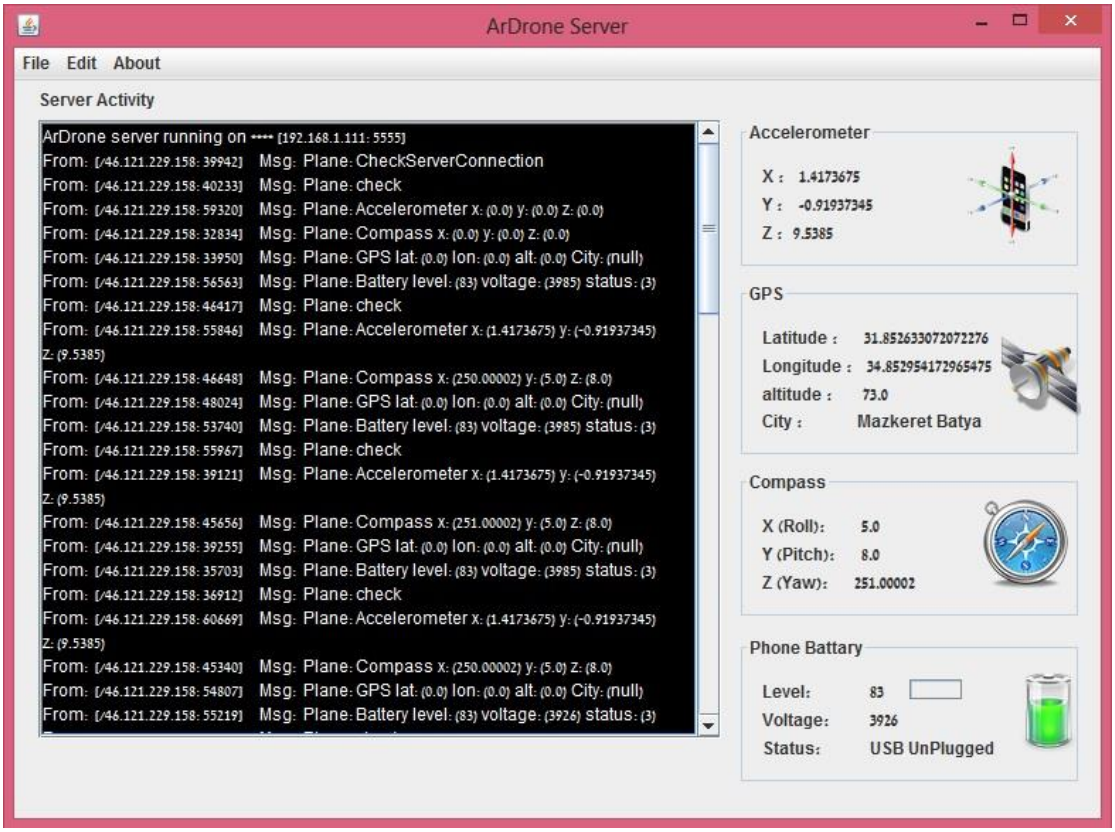
במידה ונלחץ על כפתור Back במכשיר הטלפון תוצג לנו תיבת טקסט שתשאל אותנו האם אנחנו רוצים להפסיק את הטייס האוטומטי ולחזור לניווט ידני. במידה ונבחר לא- נחזור למפת הטייס האוטומטי. במידה ונבחר כן- הטייס האוטומטי יופסק ואנחנו נחזור למסך הראשי, מסך השליטה הידני של הטיסן.



במידה וסיימנו לשחק או לעבוד עם הטיסן ואנחנו רוצים לסגור את האפליקציה נלחץ על כפתור Back במכשיר הטלפון מהמסך הראשי ותוצג תיבת טקסט השואלת את המשתמש האם הוא מעוניין לצאת מהאפליקציה. במידה ונלחץ No- נחזור למסך הראשי, במידה ונלחץ Yes- האפליקציה תיסגר.



במהלך כל הפעולות שנעשות באפליקציות המשתמש והטיסן, נשלחות הודעות לשרת הראשי. חלק מהודעות אלו מכילות את נתוני הסנסורים של מכשיר הטלפון עליו יושבת אפליקציית הטיסן. נתוני הסנסורים מייצגים לנו את נתוני הטיסה. את נתוני הסנסורים וההודעות הנשלחות לשרת הראשי ניתן לראות בתמונה הבאה:



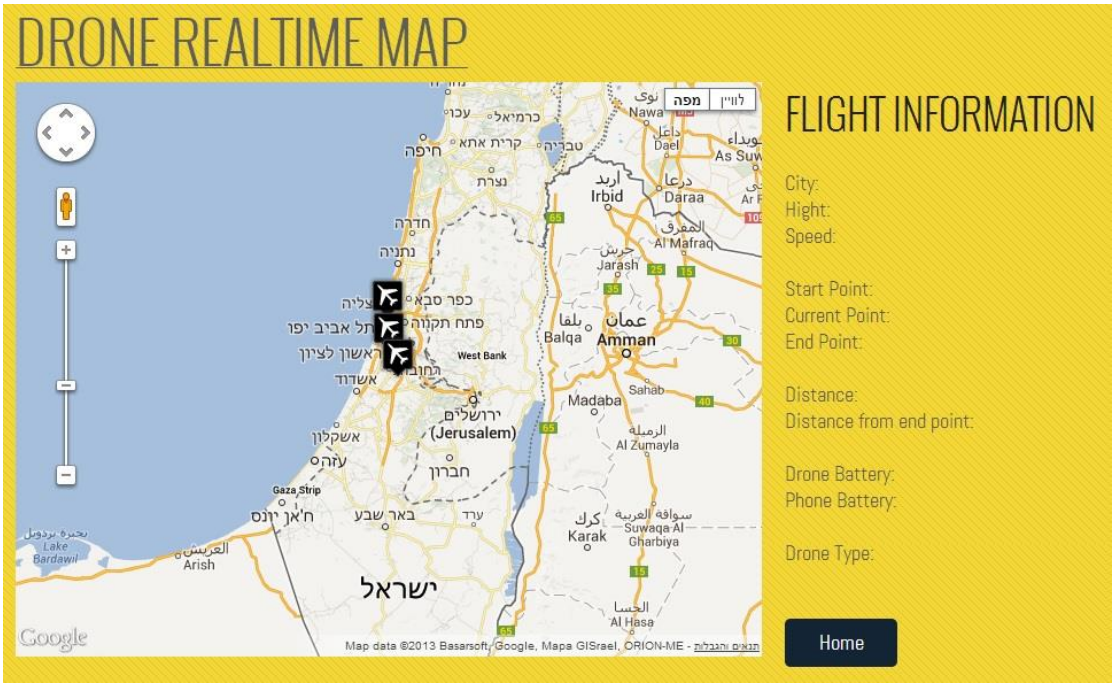
אתר הפרויקט

אתר הפרויקט מציג את תהליך ביצוע הפרויקט בקצרה.
אתר זה כתוב באנגלית ומורכב מדף אחד המחולק לנושאים הבאים :

- Concept – הסבר על קונספט הפרויקט.
- Goals – הצגת מטרות הפרויקט.
- Demo – הצגת דמו הפרויקט ע"י סרטון הדגמה.
- Architecture – הצגת ארכיטקטורת הפרויקט.
- Design – הצגת תכנון הפרויקט.
- Implementation – הצגת יישום הפרויקט.
- Result – הצגת התוצאה הסופית של האפליקציות, תוכנת השרת והחיבורים הפיזיים.
- Drone real time map – הצגת מפה של גוגל המאפשרת הצגה של כלל הטיסנים הפעילים ונתוניהם בזמן אמת.
- About Us – הצגת מפתחי הפרויקט.

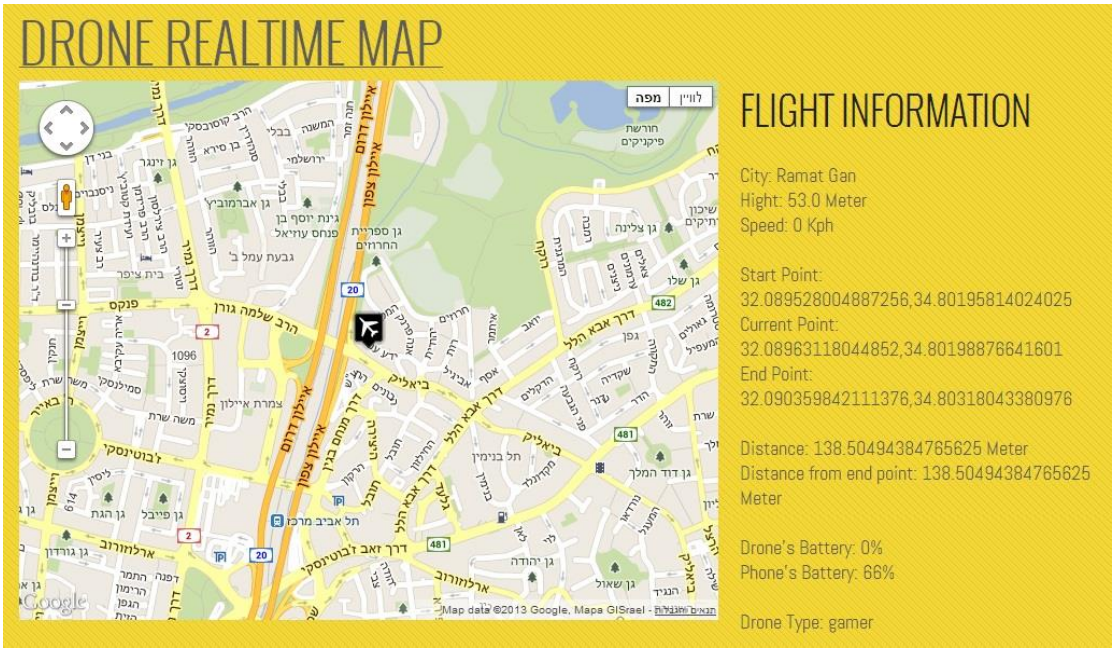
החלק העיקרי באתר הוא החלק בו אנו יכולים לנתר אחר כל הטיסנים הפעילים. אופציה זו הוספנו לפרויקט על מנת שנוכל לעקוב אחר הטיסן לא רק דרך האפליקציה אלא מכל מקום בעולם. אופציה זו שימושית בעיקר כאשר הטיסן במצב של טייס אוטומטי.

בתמונה זו ניתן לראות את החלק באתר בו אנו מציגים את המפה של גוגל ועליה ניתן לראות את הטיסנים הפעילים ע"י אייקונים המצויר עליהם טיסן לבן.



לחיצה על הטיסן הרצוי תעשה זום על הטיסן ותציג את נתוני הטיסה של הטיסן בצד ימין של המפה. נתוני הטיסן משתנים באתר בזמן אמת ולכן ניתן לראות את הנתונים משתנים ואת הטיסן מתקדם על המפה.

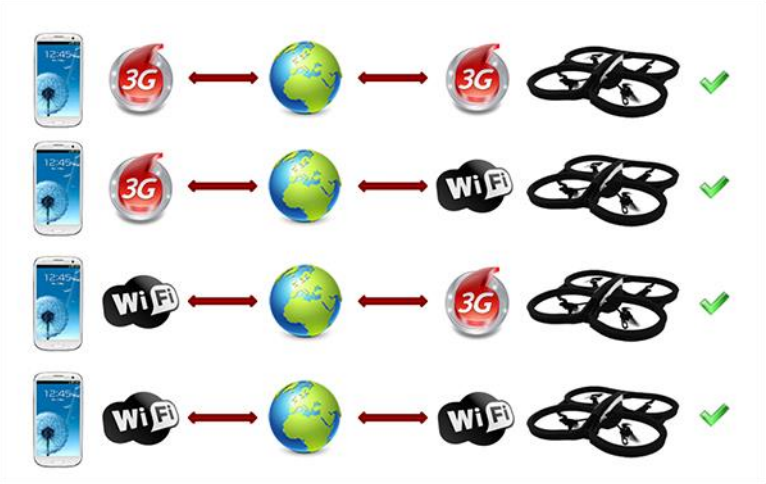
בתמונה זו ניתן לראות כי לחיצה על טיסן ביצעה זום והציגה לנו את נתוני הטיסה:



סיכום

מכיוון שבפרויקט זה אנו משתמשים בשרת ראשי על מנת לתקשר בין שתי האפליקציות, יש צורך בסיסי על מנת שהאפליקציות המותקנות על המכשירים הסלולריים יתקשרו עם השרת. צורך זה הוא גישה לאינטרנט. במידה ויש לנו גישה לאינטרנט במכשירים אז התקשורת תקינה וניתן להטיס את הטיסן. במכשירי הטלפון ניתן לגשת לאינטרנט ע"י הרשת הסלולרית 3G וע"י רשת ביתית Wi-Fi, ולכן לא משנה באיזה סוג תקשורת נשתמש ובאיזה מכשיר, בכל מצב שנקבע התקשורת תעבוד וניתן יהיה להטיס את הטיסן.

בתמונה הבאה ניתן לראות את כל המצבים האפשריים בהם ניתן להטיס את הטיסן.



אפליקציית הטיסן הנוכחית קובעת כי סוג הטיסן הוא טיסן משחק. בעתיד אפליקציית הטיסן תאפשר את בחירת סוג הטיסן (משחק, אבטחה או תעבורה). בחירת סוג הטיסן תשפיע על אופן הצגתו במפת האתר ועל אופן שמירת הנתונים בבסיס הנתונים.

במהלך פרויקט זה ישבנו וחשבנו על כל מסך ומסך באפליקציות המשתמש, הטיסן, תוכנת השרת ואתר האינטרנט על מנת לייצר אתר, תוכנה ואפליקציות מעוצבות, ידידותיות למשתמש וקלות לשימוש. עיצוב האתר, תוכנת השרת והאפליקציות מורכבות מאייקונים שנמצאו באתרים שונים, ולכן חלק מהאייקונים מותרים לשימוש באופן מסחרי וחלקם לא. מפני שפרויקט זה אינו מסחרי בשלב זה, לקחנו את הרשות להשתמש באייקונים אלה.

ביבליוגרפיה

מדריכים

מדריך של חברת parrot להתקנה והפעלת הטיסן – Ar.Drone User Guide
http://ardrone2.parrot.com/media/uploads/support_ardrone_1/ar.drone_user-guide_uk.pdf

מדריך המסביר כיצד בנוי ומורכב הטיסן – Ar.Drone Developer Guide
http://www.msh-tools.com/ardrone/ARDrone_Developer_Guide.pdf

מדריך שימוש ברכיב ה Wi-Fly – Wi-Fly User Manual
http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Wireless/WiFi/WiFly-RN-UM.pdf

פיתוח

אתר מפתחי אנדרואיד – Android Developer API:
developer.android.com/index.html

אתר מפתחי גוגל מפות – Google Maps API: developers.google.com/maps

אתר מפתחי Arduino – Arduino API : www.arduino.cc

אתר מפתחי Drone – Ar.Drone API: projects.ardrone.org

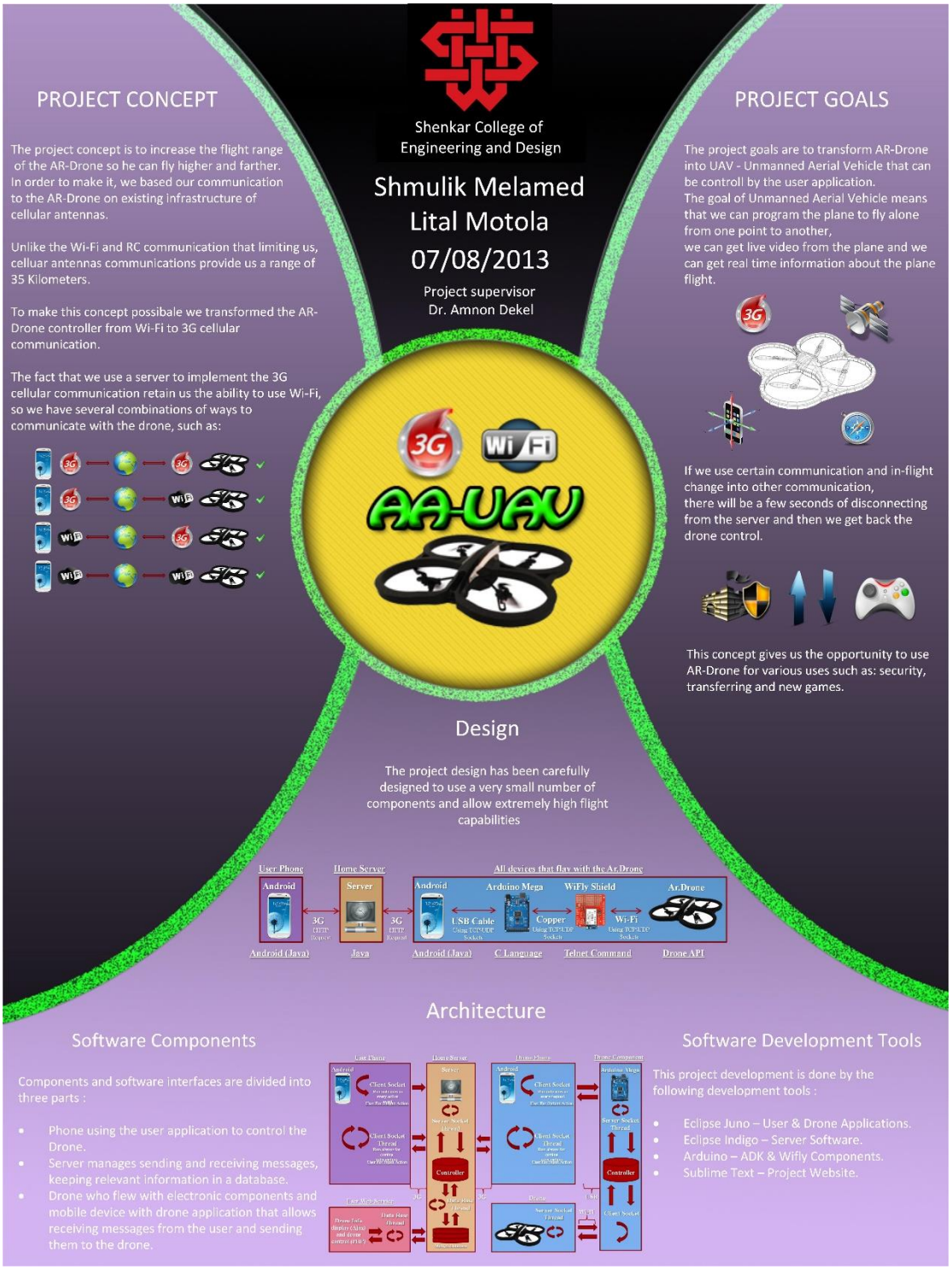
כללי

אתר אייקונים – Icon Finder: iconfinder.com

אתר הטיסן – Parrot Ar.Drone: ardrone2.parrot.com

פוסטר

הפוסטר בא לפרסם את הפרויקט המציג את המידע הרלוונטי לציבור הרחב.
את פוסטר הפרויקט ניתן למצוא כקובץ PDF ו- JPG בגודל טבעי בתיקיית Poster הנמצאת בדיסק המצורף לספר הפרויקט.



Shenkar College of Engineering and Design

Faculty of Engineering

Department of Software Engineering

System management and navigation of UAV (AR.Drone) through smartphones

Final project

By

Shmulik Melamed

Lital Motola

Submitted as Part of the Requirements for receiving

the

Bachelor of Science degree (B.Sc.)

30/08/2013