שליחת מסרונים מאובטחת

Telegraph

מס' פרויקט: _____

פרויקט גמר בהנדסה

פרויקט לשם מילוי חלקי של הדרישות לקבלת תואר בהנדסה

מאת

מאיר אמיר יוספי **גרמן אוז'יגוב**

הוגש במחלקה להנדסת תוכנה

המכללה האקדמית להנדסה סמי שמעון.

אייר התשע"ד **מאי 2014**

אשדוד

שליחת מסרונים מאובטחת

Telegraph

מס' פרויקט: _____

פרויקט גמר בהנדסה

פרויקט לשם מילוי חלקי של הדרישות לקבלת תואר בהנדסה

מאת

מאיר אמיר יוספי **גרמן אוז'יגוב**

אישור המנחה: _____

אישור ראש המחלקה: _____

אייר התשע"ד **מאי 2014**

אשדוד

העבודה נעשתה בהדרכת

ד"ר אורלוב מיכאל

המחלקה להנדסת תוכנה

המכללה האקדמית להנדסה סמי שמעון

ברצוננו להודות:

למנחה הפרויקט, ד״ר מיכאל אורלוב,
על ההכוונה המקצועית בליווי הפרויקט לאורך כל הדרך.

למזכירת המחלקה, שרון,
על הסבלנות, העזרה הרבה והנכונות לסייע.

1. הקדמה

1.1. תקציר

בשנת 1973 ביצע מרטין קופר, הנחשב לממציא טכנולוגיית הסלולר, את השיחה הסלולרית הראשונה בעולם, בשדרה השישית בלב מנהטן, לעיניהם המשתאות של העוברים והשבים. אז היה לטלפון הסלולרי יישום אחד בלבד - ביצוע שיחות (יוצאות או נכנסות). המכשירים הסלולריים הראשונים היו כבדים ומגושמים ונועדו לשימוש של העשירון העליון של האוכלוסייה.
באמצע שנות ה-90 הטלפונים הסלולריים החלו להתפשט לשכבות אוכלוסייה רבות.
כיום, אי אפשר לדמיין את העולם ללא סמארטפונים.

הסמארטפון הינו חלק מחיינו ונותן לנו מגוון שירותים מידיים שאנחנו לוקחים אתנו לכל מקום: שירותי טלפוניה, אינטרנט, מגוון רחב מאוד של אפליקציות, שליחת הודעות וקבצים בין מכשירים ומחשבים ועוד..

אך ישנו שירות אחד – המשמעותי מכולם:
שליחת הודעות כתובות (SMS) בין טלפונים סלולריים (כשכיום, רובם המוחלט בעולם הוא סמארטפונים).

לשליחת הודעות כתובות יתרונות רבים. כמה דוגמאות:

- אדם אינו יכול לשוחח בקול רם מכיוון שהוא נמצא בפגישת עסקים.
פעם, אינו יכול היה לעשות דבר. הוא לא היה עונה לשיחה ואז היו מתקשרים שוב ושוב כי לא ידעו שהוא אינו יכול היה לדבר.
כיום, במקום לא לענות לשיחת, הוא יכול להשיב בהודעת SMS, קצרה או ארוכה, למתקשר, ובה להסביר שאינו יכול לדבר / לומר שישוב אליו במועד אחר או כל דבר בו יחפוץ.

- נער המעוניין מאוד ליצור קשר עם נערה בת גילו, אך סובל מחוסר בטחון עצמי.
פעם, הנער אינו יכול היה לעשות דבר. שיחת טלפון לא היה מסוגל לעשות ובמידה ורצה להעביר מסר מסוים לאותה נערה, היה צריך לתת לה מכתב ולפגוש אותה פיזית. מה שלא היה אפשרי במצבו.

כיום, נערים רבים מפתחים את הביטחון העצמי שלהם דרך אלפי הודעות SMS שנשלחות
ממכשיר הסמארטפון שלהם ובכך הם יכולים לפנות למי שיחפוץ ליבם מבלי לפגוש אותו פיזית או לאגור ביטחון עצמי רב.

- סטודנט ממכללת סמי שמעון מעוניין לקבל את ציון פרויקט הגמר, שעליו עמל במשך חודשים.
פעם, היה צריך להגיע למכללה ולחפש את תעודת הזהות שלו על דף שהיה תלוי על לוח המודעות.
כיום, הסטודנט יקבל הודעת SMS שתגיד לו בדיוק מה הציון שלו.

- אדם נמצא באזור רועש נורא.
פעם, היה צריך להתאמץ ולנסות לצעוק בשיחת הטלפון על מנת שישמעו אותו.
כיום, יכול לשלוח הודעת SMS שבה יודיע כל דבר שירצה, ללא כל התאמצות.

דוגמאות אלו הן רק קצת המזלג בכל מה שקשור בשימוש בהודעות SMS כיום.
אנו חיים כיום בעולם שבו לכל אדם כמעט יש מכשיר סמארטפון
כשהולכים ברחוב, כשמחכים לתור בבנק, כשמתאמנים בחדר כושר, כשנמצאים בבילוי עם חברים, כשנמצאים באירוע משפחתי, ואפילו, לצערנו, גם בזמן נהיגה - אין כמעט אנשים המביטים שלא לסמארטפון שלהם.

למכשירי הסמארטפון בכלל ולהודעות SMS בפרט, יש תורמות גדולות לחיינו, אך כמו כל דבר בחיים, גם להם יש חסרונות ואף חסרונות קשים מאוד.
בין החסרונות ישנו דבר אחד ואולי הוא אף המסוכן מכולם: אבטחת המידע.
מכאן נבע הפרויקט שלנו, ומכאן הבעיה לה מצאנו פתרון.

1.2. הגדרת הבעיה

בפרויקט שלנו, לקחנו את הנושא הרגיש והנפוץ מאוד היום, עליו הסברנו בפירוט בהקדמה, והחלטנו להתמקד בנושא המסרונים (הודעות SMS).

בהקדמה הסברנו בפירוט על כמה המסרונים חשובים לחיינו וכמה הם עוזרים בכל מיני מצבים בחיי היומיום שלנו.

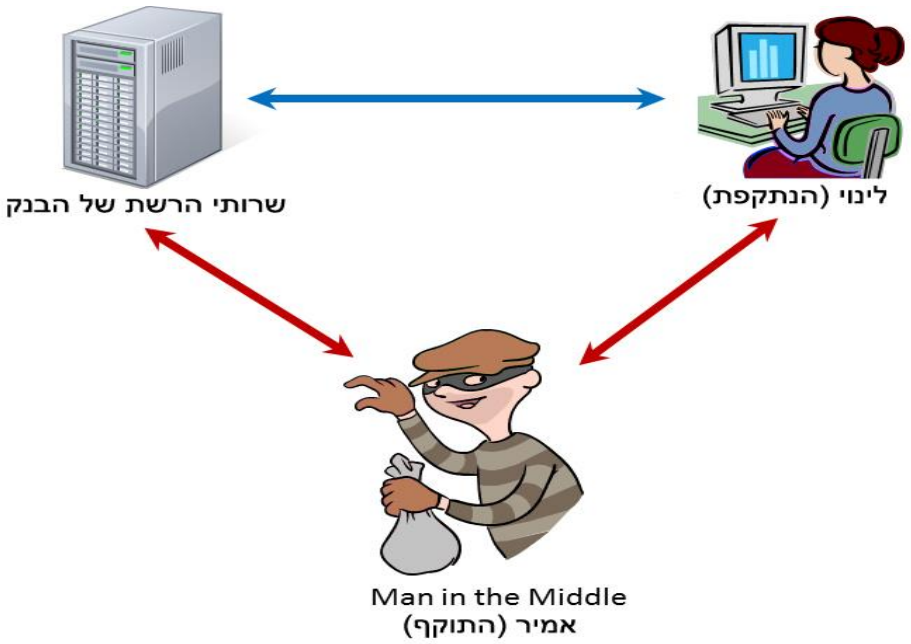
אך יחד עם זאת, יש בעיית בטיחות קשה בכל הנושא הזה.

לדוגמא:

גברת בשם לינוי המקבלת מסרון עם פרטי חשבון הבנק שלו, כולל סיסמא ושם משתמש לאתר בו הוא יכול לעשות העברות בנקאיות לחשבונות אחרים או לבצע כל פעולה אחרת בחשבון הבנק שלו.

נניח ולאותה גברת לינוי יש אויב בשם אמיר המעוניין לפגוע בה או/ו ברכושה, יוכל אמיר "להאזין" ולרגל אחרי ההודעות המתקבלות במכשיר הסמארטפון של לינוי, ובזמן ששם המשתמש והסיסמא יגיעו לידיו, יוכל להיכנס לחשבון הבנק שלה ולעשות בו כרצונו.

התקפה זאת נקראת מתקפת "אדם באמצע" או בשפה מקצועית MITM (Man In The Middle) (מצורף איור).



<u>MITM</u>:

סוג של ציתות אקטיבי שבו התוקף המאזין לשיחה המתקיימת בין שני מחשבים או ישויות ברשת (במקרה שלנו – סמארטפונים), מצליח להתחזות לכל אחד מהם בנפרד, להעביר את התמסורת ביניהם ולגרום להם להאמין שהם מתקשרים ביניהם ישירות בערוץ פרטי, בעוד שלמעשה

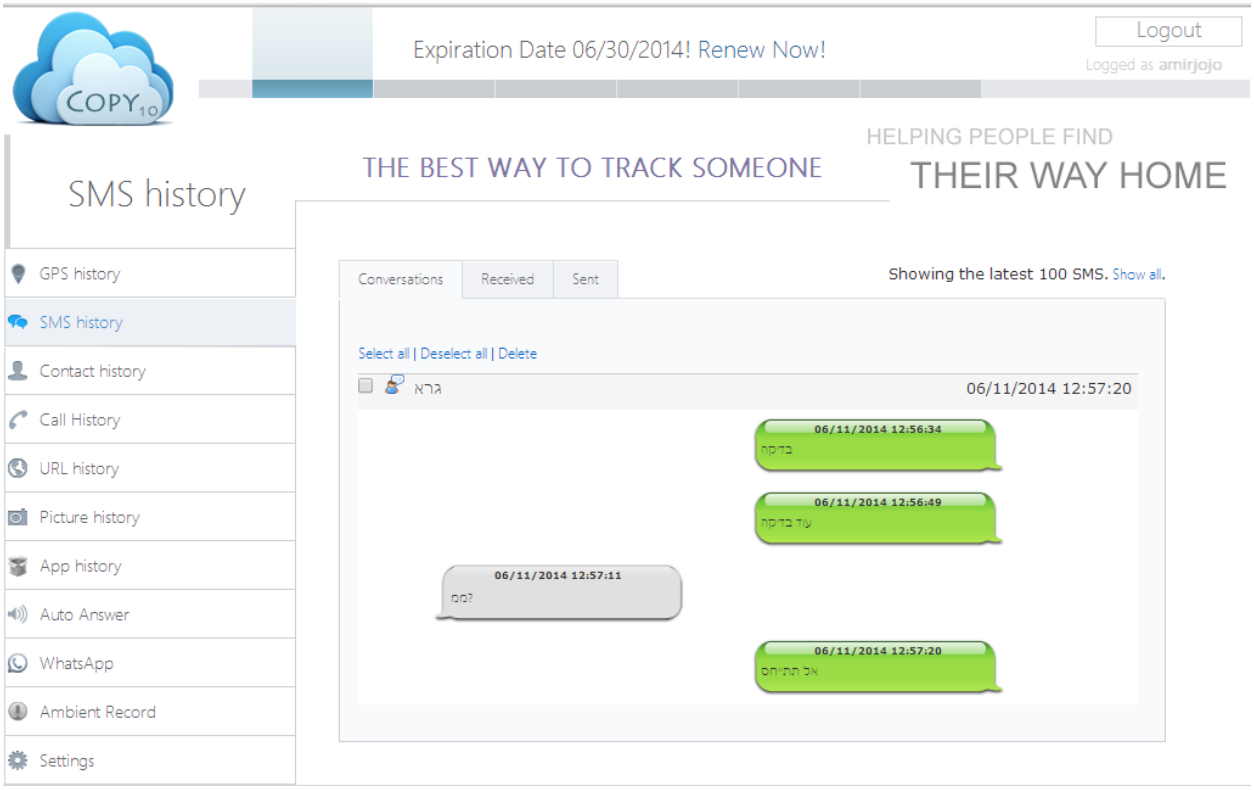
ההתקשרות נשלטת לגמרי בידיו. כדי שהתקפה כזו תצליח ראשית חייב התוקף להתחזות בהצלחה לכל צד לשביעות רצונו של השני. כמו כן חייב להיות מסוגל ליירט את כל המסרים או התשדורות העוברים ביניהם ולהוסיף, למחוק או לשנות כל מסר כרצונו (אם ירצה).

בדוגמא שלנו, אמיר יושב על התשדורות ומקבל אותן ללא ידיעת הבנק או לינוי.
הוא אינו "מתערב" או משנה נתונים, כדי שלא יעלו עליו, הוא פשוט "יושב בשקט ומאזין", כך הוא יקבל את הנתונים הנדרשים לו על מנת שיוכל לחדור לחשבון הבנק ולבצע את זממו.
ברגע קבלת הנתונים הנדרשים, אינו זקוק יותר ללינוי והוא ניגש ישירות לשרתי הבנק, מתחבר ומאמת את נתוני החשבון של לינוי.
מרגע זה – הרי שאמיר סלל לעצמו את "דרך המלך".

כיום, ישנן אפליקציות כאלה שניתנות להורדה ע"מ לבצע האזנות על רשתות תקשורת שלמות.
אפליקציות אלה נפוצות מאוד וכל אחד שיבצע root (תהליך המאפשר לנו לקבל הרשאות מנהל למערכת הקבצים הפנימית ולשנות, למחוק ולהתקין קבצים בצורה חופשית ולא מוגבלת) במכשירו יוכל להשתמש בהן ולהאזין לכל האנשים המשתמשים ברשת המדוברת.

דוגמא לאפליקציה כזו היא:

Copy10 (מצורף תצלום מסך)



אפליקציה שניתנת להתקנה בקלות ובתוך שניות מתבצע מעקב אחרי המכשיר הרצוי.

אפשרויות המעקב הן בין היתר על:

שיחות, מיקום נוכחי בדיוק של עד 50 מטרים, היסטוריה גלישה, תמונות, אפשרות האזנה לסביבת המכשיר וכמובן מעקב אחרי הודעות SMS (מסרונים).

זו הדוגמא האידיאלית לבעיה שלנו, שאותה נפתור דרך פרויקט זה.

1.3. הפתרון לבעיה

בפתרון שלנו התמקדנו בחלק של הודעות SMS (מסרונים).

יצרנו אפליקציה לשליחת מסרונים, כזו שיודעת לשלוח אותם ולקבלם – בצורה מאובטחת ומוצפנת, במספר דרכים, כשהכל קורה בצורה שקופה למשתמשים.

יצרנו מעין אפליקציה חדשה, שתפעולה די דומה לאפליקציות שונות של שליחת מסרונים.

המשתמש יקיש את ההודעה ואת היעד, יבחר את סוג ההצפנה המבוקש וישלח את המסרון בבטחה וללא חשש.

האפליקציה תצפין את המידע תוך שימוש במספר סוגי הצפנות (יפורטו בהמשך) שונות כאשר כל אחת מהן מצפינה את המסרונים בדרך שונה ודואגת שלא יוכלו לפענח אותם.
האפליקציה יודעת להסתנכרן מול הסמארטפון בכל מה שקשור לאנשי קשר שאותם נוכל לחפש דרכה ולבחור לשלוח את המסרון לאדם הספציפי אותו רצינו.
שלבים בשליחת מסרון דרך האפליקציה:

1. כניסה לאפליקציה – זיהוי אוטומטי של המספר דרך כרטיס הסים או הרשמה מיידית במידה וזו הכניסה הראשונה לאפליקציה.
2. הוספת שיחה – מוסיף מספר טלפון או איש קשר (מקושר למכשיר) ובחירת סוג תשדורת מתוך 3 האפשרויות הקיימות (הצפנות).
3. מסך ניהול השיחה – שבה נראה את כלל התשדורות ואל מי נשלחו.

1.4. מטרת הפרויקט

מטרת הפרויקט היא לאפשר למשתמש לשלוח מסרון לכל יעד ללא חשש מחשיפת המידע שבו ליעדים לא רצויים וללא חשש מהתקפות סייבר כגון MITM (שהזכרנו למעלה) או דומיה.
פתרון זה הינו הצעד הראשון, מתוך כלל הצעדים שידרשו למניעת זליגת המידע מהסמארטפונים לידיים שאינן בטוחות.

1.5. שלבי ביצוע הפרויקט

- חקירת נושא אבטחת המידע בסמארטפונים על בוריו, תוך חיפוש מאמרים מתאימים ושיחות עם אנשי מפתח העוסקים בנושא.
- התמקדות בבעיה הבנתה לעומק וחשיבה על פתרון שייתן את המענה המרבי והנכון ביותר.
- חשיבה וגיבוש פתרון, מבחינה אפליקטיבית ומבחינה תשתיתית, ע"י אפליקציה מתאימה.
- כתיבת מסמכי אפיון ודרישות לפיתוח האפליקציה.
- חשיבה על מסד הנתונים, ועיצוב האפליקציה.
- תכנון אבני הדרך ע"פ מתודולוגיית הפיתוח שבחרנו.
- כתיבת תסריטי בדיקות.
- פיתוח התשתית (מסד הנתונים, פונקציות תשתיתיות) והכנתה לפיתוח האפילקטיבי.

- פיתוח הצד האפליקטיבי של האפליקציה.
- אינטגרציה וחיבור בין התשתית והאפליקציה.
- הרצת בדיקות הקצה.
- הגשת מוצר סופי.

2. מסמך דרישות SRS

2.1. הקדמה

2.1.1. מטרת המסמך

מטרת המסמך היא לתאר את הקווים המנחים בפיתוח ועיצוב האפליקציה. המסמך יתאר את קהל היעד, את ממשק המשתמש, סביבת העבודה הדרושה לפיתוח והרצת האפליקציה, ואת אילוצי המערכת השונים. כמו כן המסמך יכיל מידע רלוונטי לשם פיתוח האפליקציה ומפרט חומרה דרוש.

2.1.2. קהל היעד

מסמך זה נכתב עבור:

1. צוות פיתוח האפליקציה.
2. תחזוקת האפליקציה ושדרוגה העתידי במידה ויהיה.

2.2. משתמשי המערכת

האפליקציה פותחה בראש ובראשונה, עבור העברת מסרונים בצורה מאובטחת, לחברות ולנתונים רגישים (בנקים, סיסמאות וכו').

בנוסף – האפליקציה תהיה מקור הביטחון למשתמשים אשר עדיין אינם סומכים על אבטחת המידע ומשתמשים בדרכים מיושנות (ללכת לסניף הבנק כדי להוציא טפסים ולבצע פעולות וכדומה) ובכך תביא אותם לקדמת הטכנולוגיה.

האפליקציה מתוכננת עבור שימושים פרטיים ועסקיים כאחד.

2.3. תיאור כללי

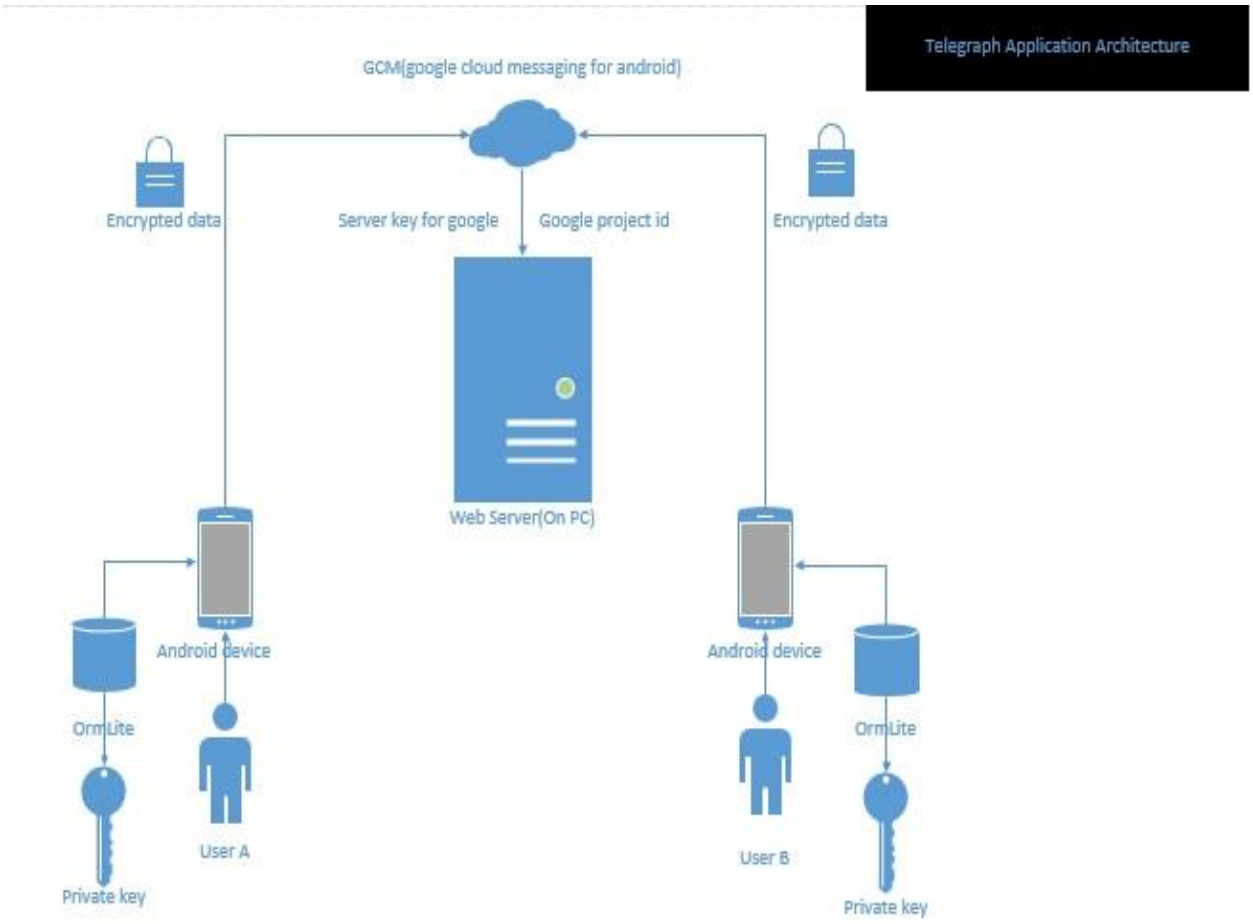
2.3.1. תכלית הפרויקט

תכלית הפרויקט היא שליחת מסרונים מאובטחת.

התהליך שקוף למשתמש ומבחינתו הוא שולח מסרונים ומקבלם כמו שהיה רגיל עד היום.

בפועל המסרונים מוצפנים בדרכים שונות ע"מ שלא יוכלו לפענח אותם בקלות ולהשתמש בהם בשימושים שאינם רצויים למשתמש.

2.3.2. ארכיטקטורה ותפיסת פתרון



השימוש באפליקציה מורכב מהחלקים הבאים:

1. מסד הנתונים עליו נשמר כל המידע של המערכת בצורה מאובטחת.
2. שרת המשמש לשליחת ההודעה המוצפנת בין מכשירי האנדרואיד.
3. המשתמש אשר יכול לגשת באמצעות כל מכשיר המתאים לאפליקציה ולשלוח מסרונים מאובטחים.

2.3.3. מאפייני המוצר

המערכת מאופיינת ב:

1. משק משתמש המפותחת בטכנולוגיות המתקדמות ביותר בתחום, המעניקה חווית משתמש נוחה ואיכותית, בכך הדבר מקל מאוד על המשתמש בשליחת מידע מאובטח מהסמארטפון.
2. חסכון כספי – המסרונים משתמשים ברשת האינטרנט בכדי לשלוח את המסרונים ובכך אין חיוב מיוחד לקו הטלפון על השימוש.

בנוסף, האפליקציה חינמית ופתוחה לשימוש הקהל הרחב ואינה דורשת דמי הרשמה כלל.

2.4. אילוצי המערכת

2.4.1. סביבת עבודה

סביבת עבודה לשימוש:

- מכשיר אנדרואיד בעל גרסה 2 ומעלה.
- אין מגבלות נוספות היות והאפליקציה מתאימה לכלל המכשירים.

סביבת עבודה לפיתוח:

- מערכת הפעלה Windows XP ומעלה.
- Eclipse (מומלץ בגרסת Kepler).
- Java 6 ומעלה.
- Android SDK Manager
- Android Virtual Device manager.

2.5. דרישות

2.5.1. דרישות פונקציונאליות

תחום העבודה :

1. מנחה פרויקט – אחראי ללוות ולעקוב אחר התפתחות הפרויקט מתחילתו ועד סופו.
2. מפתחים – אחראים לפיתוח מוצר התוכנה:

- פיתוח ממשק נוח ואינטואיטיבי ככל הניתן בכדי לאפשר למשתמש לעבוד עם המערכת בקלות.
- פיתוח מנגנון הצפנת מסרונים יעיל, מהיר ובטוח ככל הניתן.
- שמירת הנתונים בכל רגע עבור המשתמש במערכת.

תחום המוצר:

- על המשתמש להתקין את האפליקציה למכשיר השולח ולמכשיר אליו תשלח ההודעה.
- בכניסה הראשונה למערכת – המשתמש מחויב להירשם עם שם משתמש תחת מספר הטלפון שלו שיישלף אוטומטית מכרטיס הSIM.
- מיד לאחר הכניסה יוכל המשתמש להשתמש במערכת ולשלוח מסרים מוצפנים לסמארטפונים הרצויים.

2.5.2. דרישות לא פונקציונאליות

עיצוב (מערכת ניהול החוקות):

- עיצוב איכותי וקליל שיעזור להתמצאות המשתמש באפליקציה.
- עיצוב התומך בחוויית משתמש מהטובות שקיימות בשוק העולמי.

אבטחה:

- רישום ראשוני למערכת מתבצע ע"י כתיבת שם המשתמש בלבד (אינו המפתח) ואילו מספר הטלפון נלקח אוטומטית מכרטיס ה SIMללא התערבות של המשתמש ובכך מונע גניבת זהויות וכדומה.
- הזדהות ע"י שם משתמש המתקבל מהמערכת.

איכות התכנה:

האפליקציה תהיה בעלת התכונות הבאות:

- ניתנת להרחבה בקלות – קוד גני כך שניתן להוסיף תנאים חדשים, פונקציות חדשות ובכלל פעולות עתידיות (הגנה על תמונות, הגנה על שיחות וכו') בצורה פשוטה.
- ניתנת לתחזוקה – כתובה בצורה ברורה לפי שיטות העיצוב המקובלות כך שבעתיד יהיה ניתן לשדרג אותה במידת הצורך.
- נוחה לשימוש – האפליקציה תמומש בצורה שהמשתמש ידע לתפעל בקלות.
- יציבה.
- עצמאית – האפליקציה תפעל ללא תלות באפליקציות אחרות.

3. סקר ספרות

3.1. תיאור הבעיה

העברת מסרים ברשת wireless בצורה בטוחה בלי פגיעות של האזנת פקטות או התערבות בתוכן המסר המועבר.

3.2. תקציר הפרויקט

מטרת העל של הפרויקט היא לדמות תקשורת בין מספר מכשירי אנדרואיד על בסיס פרוטוקול מאובטח שיפותח במקביל עם יישום אנדרואיד כחלק מהפרויקט ויהיה חסין מפני תקיפות מסוג man in the middle(MITM) ופעולות sniffing למיניהן. בנוסף המערכת תיתן אפשרות לשליחת מסרים בצורה אנונימית בין המכשירים. כחלק מהפרויקט יפותח סימולטור המדמה תקיפות 'מוצלחות' על המכשירים המתקשרים ולאחר מכן יתקוף ללא הצלחה את התקשורת של המכשירים על בסיס הפרוטוקול המאובטח. כמו – כן תהיינה תמיכה במגוון רחב של מכשירי אנדרואיד ובוצעו בדיקות למקרי קצה.

3.3. השיטה

מאחר ולא נוכל למנוע מהתוקף להתקרב לאות של התקשורת האלחוטית, נצא מנקודת הנחה שהתוקף מאזין לתעבורת הרשת שכן חשופה לו. בדרך של הצפנת המידע נוכל להבטיח שביכולת התוקף לאסוף מידע מוצפן בלבד בתווך שמאזין לו. עלינו להבטיח מחד, כי ההצפנה תהיה כמה שיותר חזקה ומאידך שלא תכביד על התווך.

3.4. הסקר יתמקד בשלושה מרכיבים העיקריים של הפרויקט:

3.4.1. שיטת הגנה מפני תקיפות והאזנות ברשת wireless.

למעשה מדובר בדרכי פעולה שניתן לנקוט למניעת תקיפות מסוג Hijacking,Eavesdropping , MITM:

מס"ד	שיטה	יתרון	חיסרון
1	ווידוא כי כל ההתקנים המחוברים לרשת מוכרים	זיהוי של התקן זר ברשת	לא שמיש לרוב הצרכנים מאחר ויכולות ניהול מוענקות רק למנהלנים

2	התחברות ל VPN (Virtual Private Network)	קושי התחברות של גורם זר ואימות חזק בצד השרת	במידה והתוקף מצליח להתחבר לרשת המידע חשוף בפני האזנה
3	הצפנת SSL	הצפנת תעבורה	מדובר בפרוטוקול HTTP בלבד שהופך לHTTPS
5	סוגיית ה Handshake הראשוני	יש להקדיש חשיבות עליונה להצלחת ה handshake הראשוני ולהבטחתו, לכן נשתמש בחירות חזק כגון WPA2-PSK. כמו כן נבחן הבדלים בין 2-way ל 4-way פרוטוקול.	ללא handshake מאובטח תיתכן פריצה / התחזות החל מהשלב הראשוני של תקשורת בין המשתמשים
6	NaCl	מספק ממשק קל להצפנות פענוחים וחתימות	לא תומך ב android
7	Sodium	כנ"ל ונתמך באנדרואיד	
8	OpenSSL	מספק שירותי הצפנה ותומך באנדרואיד	עיקר הדגש על NDK של אנדרואיד, שכן מדובר בnative development key
9	DnsCrypt	הצפנת תעבורת DNS בין משתמשי הקצה ברשת	נדרשות הרשאות ROOT
10	RNCryptor	מאפשר מבחר של סוגי הצפנות המופעלות על ה DATA כמו AES	לא תומך במבנה JASON, נדרשת המרה לString
11	Java Cryptography Extension (JCE)	פלטפורמה ומימוש של הצפנות, הגרלת מפתחות וחתימות דיגיטליות; מובנה בJAVA ותומך ב androidsdk	רלוונטי החל מ java 6 – חסרון לא משמעותי

כלים עמם ניתן לבצע תקיפות אלו:

,Cain and Abel, CommView for WiFi, Packet analyzer and etc WireShark

מקורות על שיטת מניעה מועדפת:

- ספרים
- מאמרים

- Two-way Handshake protocol for improved security in IEEE 802.11 wireless LANs by Chang-Seop Park Department of Computer Science, Dankook University, Anseo-dong, Chonan, Choongnam 330-714, South Korea
- Analysis of energy consumption of RC4 and AES algorithms in wireless LANs– By Prasithsangaree, P and Krishnamurthy, P.
- ARMORED: CPU-Bound Encryption for Android-Driven ARM Devices– By Gotzfried, J. and Muller, T.

3.4.2. שיטת העברת מסרים בצורה אנונימית ברשת wireless.

ניכר כי עיקר השימושיות בהעברת מסרים בצורה אנונימית היא למעקף של ציטוטים וניטורים של גופים ממשלתיים רבים בעולם.

דרכי פעולה להעברת מסרים בצורה אנונימית ברשת wireless:

מס"ד	שיטה	יתרון	חיסרון
1	שימוש בקהילת הקוד הפתוח של OrBot	זיהוי של התקן זר ברשת	לא שמיש לרוב הצרכנים מאחר ויכולות ניהול מוענקות רק למנהלנים
2	שימוש באפליקציה לשליחת מסרים אנונימיים באנדרואיד	שליחת מסרים אנונימיים דרך אפליקציה ללא דרישה לביצוע ROOT למכשיר.	תשלום גבוה עבור תמיכה לכל מסרון אנונימי ספציפית.

כלים עמם ניתן לבצע העברת מסרים בצורה אנונית: OrBot, Anonymous SMS, Mr Hide ועוד רבים השולחים הודעות אנונימיות באנדרואיד דרך הרשת האלחוטית.

מקורות על שיטת מניעה מועדפת:

- ספרים.
- מאמרים.
  - An Efficient Anonymous Communication Protocol for Wireless Sensor Networks –
    By Juan - Chen School of Computer Science and Technology,
    Xiaojiang Du - Department of Computer and Information Sciences and
    Binxing Fang - School of Computer Science and Technology
  - Anonymous communications in mobile ad hoc networks –
    By Yanchao Zhang, Wei Liu and Wenjing Lou
  - SDAR: a secure distributed anonymous routing protocol for wireless and mobile ad hoc networks –
    By El-Khatib, K. ; Li Xu ; Korba, L. and Boukerche, A.

3.4.3. מבנה וצורת פיתוח של אפליקציית העברת מסרים ברשת wireless.

נקודה מנחה של הפרויקט הייתה לספק יכולות שתוארו לעיל בצורה פשוטה עם נגישות של כלל האוכלוסיה ולהרחיב את תפוצת האפליקציה באמצעות תיאמותה לכלל מכשירי אנדרואיד ובכך להגביר את המודעות של האוכלוסיה כלפי נקודות התורפה שעורבות בחיי היום יום.

דרכי פעולה לפיתוח האפליקציה:

מס"ד	שיטה	יתרון	חיסרון
1	שימוש ביכול המוענקות בזכות הרשאות ROOT על המכישר	ניצול מיטבי של יכולות המכישר	לא מתאים לאוכלוסיה הרחבה, שכן לא בעל יכולות ROOT על מכשירם.
2	כתיבת native code ושימוש ב NDK	כנ"ל	כנ"ל
3	שימוש ב SDKים להצפנות	קל למימוש; אמין ונבדק	ייתכן כי מוכר ולכן לא ייחודי
4	שימוש ביכולות מובנות של JAVA לביצוע הצפנה	פחות ממשקים – פחות נקודות תורפה. קלות השימוש ותאימות לשפת פיתוח.	מדובר בקוד פתוח ונגיש. רק החל מ JAVA 1.7 מדובר בחלק אינטגרלי מהשפה (עד אז JARים משלימים) אלגוריתמים מתקדמים עם מפתחות פרטיים אצל הנמענים ישמרו ב DB.
5	שימוש ב SDK פשוט של אנדרואיד לeclipse	פשוט, חוצה פלטפורמות ומותאם לכלל המכשירים	לא נגיש בקלות לכלל הרכיבים לעומת native mode
6	העברת מסרים ברשת על ידי socketים	מובנה ב JAVA ללא ממשקים ושרתים חיצוניים	מסתמך על threadים וככל שעולה כמות המשתמשים סבירות גבוהה יותר של threadים ליפול. מסתמך על פורט ושרת מארח – חשוף מפני תקיפות של dos. קשה לסמוך על הסדר של חבילות והתאוששות מפני איבוד נתונים במידה ויש עומס ברשת
7	העברת מסרים ברשת באמצעות GCM(Google Cloud Messaging)	הדרך המקובלת לעדכון והתעדכנות מול מכשירי האנדרואיד. בנוי לעבודה עם ריבוי משתמשים. האחראיות על הצלחה ושלמות ההודעה מנוהלת היטב ללא התערבות המפתח	מסתמך על משתמש מנהלן ב console.developers.google. ממשק מסורבל המצריך החזקת שרת בקשות ייחודי.

סקירות טכנולוגיות וכלים:

/http://www.xda-developers.com

/http://developer.android.com

מקורות לדוגמאות עזר ושיתוף בקהילה:

github

stackoverflow

כלים וסביבות עבודה נבחרות לפיתוח:

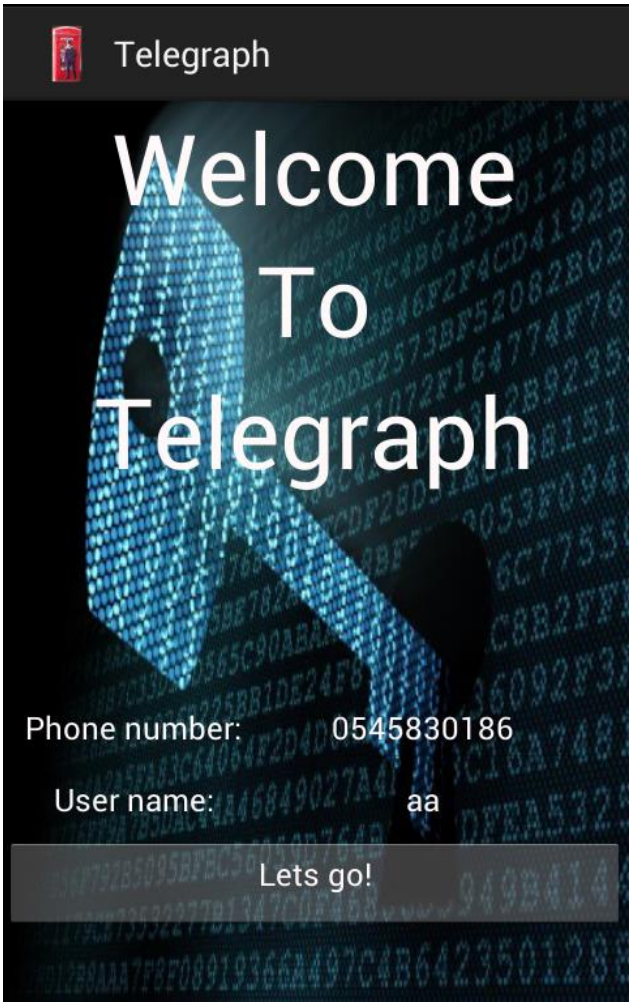
Eclipse עם MANAGER Android SDK, GiT.

שיטת מימוש הקישוריות טרם נבחרה מבין החלופות שהוצגו בסעיף 3 שלעיל.

כמו כן נושא אחסון המידע פתוח לאור ריבוי של הכלים המוצעים כיום בשוק.

4. מסכים באפליקציה

4.1. מסך הכניסה

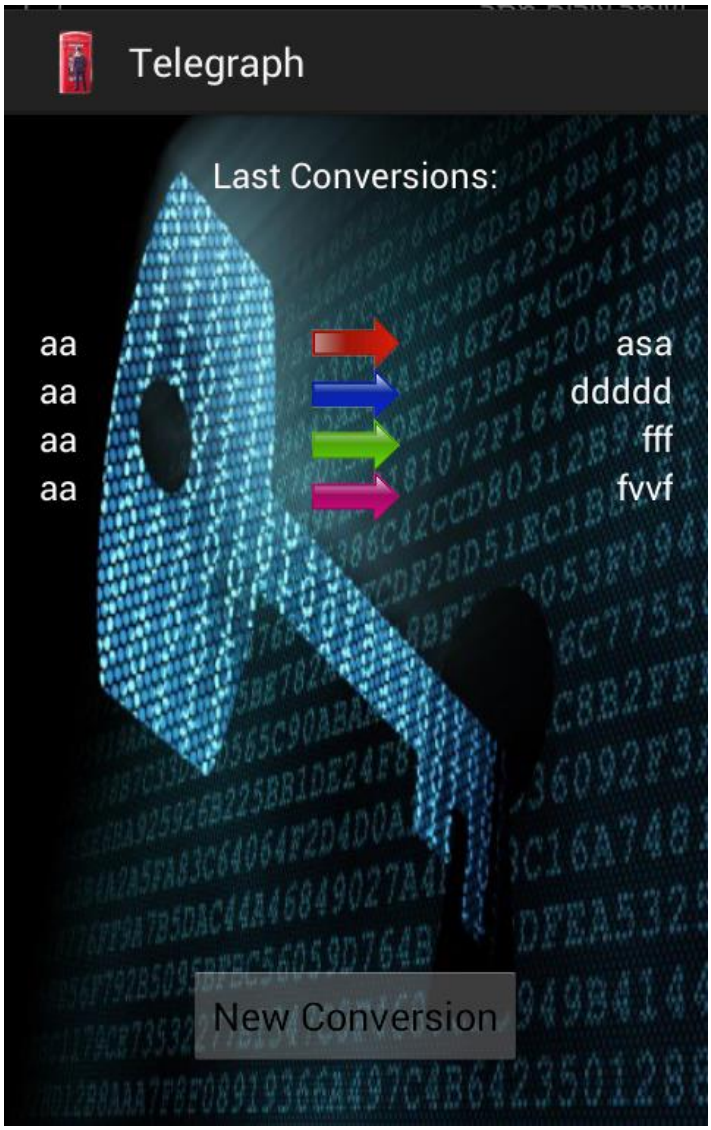


מסך הכניסה יודע לזהות את מספר הטלפון היישר מכרטיס הSIM.

במסך הכניסה, בחיבור לאפליקציה בפעם הראשונה, יש להקליד שם משתמש.

במסך ישנו לחצן שלאחר הכנסת הנתונים ולחיצה עליו – נכנסים לאפליקציה.

4.2. מסך שיחות



במסך השיחות נוכל לצפות בכל השיחות.

כל סוג הצפנה מופרד על ידי צבע חץ שונה וכך נוכל להבחין בין ההצפנות השונות.

בצד שמאל של החץ מופיע שם השולח ובצד ימין שם הנשלח.

ישנו לחצן להוספת שיחה חדשה במסך.

4.3. מסך הוספת שיחה

במסך הוספת שיחה נוכל לבחור מספר טלפון ישירות או לחילופין לחפש באנשי הקשר של המכשיר את האדם אליו נרצה לשלוח את ההודעות.

בנוסף, יש לבחור את סוג ההצפנה.

בלחיצה על חזרה נחזור למסך שיחות.

בלחיצה על מסך אשר – השיחה תתווסף למסך השיחות.

4.4. מסך חלון שיחה

במסך של חלון השיחה ננהל את השיחה עצמה.

נשלח הודעות (שישלחו מוצפנות) ונקבל הודעות (שיפוענחו על ידי האפליקציה).

5. הצפנות

מאז ומתמיד היו שליטי מדינות ומצביעים צבאיים, מודעים לתוצאות החמורות, שעלולות להיות לנפילת מסרים סודיים לידי גורמים עוינים. לכן ניסו לפתח שיטות שונות להסתרת מסרים חשובים, שהעבירו זה לזה, על ידי שימוש בצפנים ובכתבי סתר. כך הלך והתפתח תחום ההצפנה, עד שהיה למדע של ממש. תחום ההצפנה מקיף מגוון של שיטות לשיבוש מסרים והסתרתם, מפני כל מי שהמסרים לא נועדו לו. לצורך זה פותחו מחלקות שלמות, שעסקו בהמצאה וביצירה של צפנים. במקביל לכך העסיק היריב או האויב, מפצחי צפנים, אנשים שכל תפקידם היה לגלות את סודותיו המוצפנים של הצד השני.

אפשר להמשיל את מפצחי הצפנים לאלכימאים, חוקרי החומר שחיו בימי הביניים וקדמו לכימאים של ימינו. הם האמינו בקיומו של חומר פלאי, שקראו לו אבן החכמים, ההופך כל חומר לזהב. הם בילו שנים רבות בניסיונות מתסכלים לגלותו. פיצוח הצופן משול לגילוי אבן החכמים. הוא נותן משמעות לאוסף סמלים, חסר פשר לכאורה. ממש כמו אבן החכמים שאמורה הייתה להפוך חומרים "נחותים" לזהב. קיים כמובן הבדל עקרוני בין האלכימאים למפצחי הצופנים. אבן החכמים היא מושג שווא, חזיון תעתועים. אבל פיצוח צופנים הוא משימה אמיתית, אם כי היא עשויה להיות מאד מפרכת.

ההיסטוריה של ההצפנה היא סיפור ההתמודדות בת אלפי השנים בין יוצרי הצפנים לבין מפצחיהם. לתחרות המוחות הזאת הייתה השפעה גדולה על מהלך ההיסטוריה האנושית. החל מיוליוס קיסר, שכבר לפני 2500 שנה, שלח לשלטונות רומא מסרים מוצפנים משדה הקרב, ועד לפרשה המרתקת של מכונת ההצפנה הגרמנית - אניגמה, שהייתה בשימוש בשנות ה - 30 וה - 40, של המאה ה- 20, והשפיעה באופן דרמטי על מהלך המערכה באירופה במלחמת העולם השניה. אלה רק שתי דוגמאות מבין רבות אחרות.

כיום אנו חיים בחברה גדושת מידע. הנגישות למקורות מידע והשימוש בסוגים שונים של מידע, רחבים יותר מאי פעם ומחלחלים לכל תחומי החיים. פעולות של יום יום כמו שיחות טלפון, שימוש בכרטיסי אשראי, ביצוע עסקות באמצעות האינטרנט ופעולות רבות נוספות כולן אינן חסינות מפני ציתות או התערבות זדונית אחרת. ההצפנה היא דרך יעילה מאוד לאבטחת חסיון המידע של היחיד ושל גופים עסקיים ואחרים.

מדע ההצפנה נקרא בלועזית קריפטוגרפיה. אפשר לומר כי הקריפטוגרפיה, מדע התקשורת הסודית, מאפשרת את שגשוגם של הכלכלה ושל שוק הכספים בעידן המודרני.

אולם ההתפתחות הטכנולוגית המרשימה בתחום זה מעוררת, בעת האחרונה גם בעיות רציניות. מתברר שהיכולת להעביר מסרים מוצפנים באינטרנט עלולה להיות מנוצלת על ידי גורמים עוינים לסדר החברתי, למשל על ידי קבוצות טרור, המתקשרות בדרך זו. גורמי בטחון פנים, במאבקם נגד הטרוריסטים, מגבירים את הציתות והמעקב אחרי כל האזרחים. מה עם חירויות הפרט? מה עם חסיון המידע? האם הם עלולים לסכן את בטחון המדינה? - חומר למחשבה.

5.1. סוגי הצפנות

5.1.1. צופן שחלוף

בהצפנה על פי צופן שחלוף, מוחלפת כל אות במסר המקורי באות אחרת, (או בסמל אחר), כך מתקבל המסר המוצפן. מכאן השם צופן שחלוף. סדר האותיות אינו משתנה בתהליך ההצפנה. אחת משיטות השחלוף הפשוטות והעתיקות, מופיעה כבר בספר ירמיהו בתנ"ך ונקראת אתב"ש. כדי להצפין בשיטה זו כותבים את הא"ב בשתי שורות, העליונה מן ההתחלה לסוף, כמקובל והתחתונה במהופך, מן הסוף להתחלה.
בטבלה שלהלן. המפתח הוא אפוא:

א ב ג ד ה ו ז ח ט י כ ל מ נ ס ע פ צ ק ר ש ת
ת ש ר ק צ פ ע ס נ מ ל כ י ט ח ז ו ה ד ג ב א

ההצפנה מבוצעת על ידי החלפת כל אות בטקסט המקורי, באות שתחתיה וכך מתקבל המסר המוצפן.
לדוגמא, המסר: אם אין אני לי מי לי
ייכתב באתב"ש: תי תמט תטמ כמ ימ כמ
לטבלה אנו קוראים מפתח ההצפנה. המפתח מורה לנו באיזה אות לשחלף כל אחת מאותיות המסר המקורי, על מנת להצפינו. כמו שכבר הזכרנו, המפתח שהכרנו זה עתה, נקרא מפתח אתב"ש.
המפתח נועד כדי להקשות על הפענוח, נהוג לשבור את החלוקה למילים ולקבץ את אותיות המסר המוצפן בקבוצות של 4 או 5 אותיות. כשנבצע זאת על המסר המוצפן נקבל: תיתמ טתטמ כמימ כמ.

5.1.2. DES

5.1.2.1. הרקע לפיתוח DES

בסוף שנות ה- 60 של המאה ה- 20 החל עידן חדש בהצפנה, כאשר חברות נזקקו לדרכים מאובטחות להעברת מסמכים וקבצים. מוסדות ציבוריים, בעיקר המוסדות הפיננסיים, נזקקו לתקן הצפנה בדוק ואמין, בו יוכלו להשתמש בביטחון מלא, להחלפת נתונים. לחברות רבות היו מחשבים שיכלו לבצע הצפנות, אבל כדי שכל אחת מהן תוכל לקיים תקשורת ברמת אבטחה ידועה ומוסכמת על כולן, חסר היה תקן. חסרה שיטת הצפנה מקובלת, מוסכמת אוניברסלית. ב- 1974 החליטה NSA (National Security Agency) בארה"ב, על ה- DES, שיטת הצפנה שהוצעה על ידי IBM, כתקן הצפנה מחייב.
השיטה ידועה גם בשם: DEA (Data Encryption Algorithm). ב- 1977 הוחלט כי השיטה תעמוד לביקורת של NSA, מידי 5 שנים.

5.1.2.2 תיאור ה DES

- האלגוריתם מתוכנן להצפין בלוק נתונים, באורך קבוע של 64 ביטים, ועל כל בלוק מופעל אותו אלגוריתם שאינו תלוי בבלוקים עצמם. לכן ה - DES הוא דוגמא לצופן בלוקים. צופן שבו תווי המידע, נחלקים לקבוצות (בלוקים) באורך קבוע.
- אורך המפתח ב- DES הוא 64 ביטים.
- שני אנשים שרוצים לתקשר באמצעות DES, צריכים להסכים על מפתח סודי. כל יתר הנתונים הקשורים ב- DES הם מידע ציבורי. האלגוריתם ידוע לפרטיו וניתן לרכוש ללא הגבלה חומרה ותוכנה שמבצעות הצפנה DES.
- תיאור האלגוריתם:
  - למפתח הסודי K, בוחרים המשתמשים 7 תווים של 8 ביטים, בסך הכל 56 ביטים. ה- DES

מוסיף 8 ביטים נוספים, המשמשים לביקורת וכך מתקבל מפתח של 64 ביטים.
  - בעזרת K, יוצרים תמורה התחלתית על בלוק הנתונים בן 64 הביטים.
  - מבצעים 16 מחזורים של ערבול ושחלוף של בלוק הנתונים.
  - על הבלוק הסופי (אחרי המחזור ה - 16), מבצעים תמורה הופכית לתמורה ההתחלתית.
  - 64 הביטים שהתקבלו הם המסר המוצפן.
- הפענוח מבוצע באמצעות מפתח ההצפנה, על ידי חזרה על הליכי ההצפנה בסדר הפוך, והם:
  - מבצעים תמורה התחלתית על בלוקים של הצופן.
  - מבצעים 16 מחזורים של ערבול ושחלוף הפוכים לאלה שבוצעו בהצפנה.
  - מבצעים תמורה הפוכה על כל בלוק של הנתונים.

5.1.2.3 עד כמה ה- DES הוא חסין פיצוח?

המפתח של DES הוא בן 56 ביטים. כמה מפתחות שונים אפשר ליצור מ - 56 ביטים? השאלה היא כמה מספרים בני 56 ספרות אפשר ליצור משתי ספרות, והתשובה היא 2 בחזקת 56, 70 קווודריליון מפתחות פוטנציאליים!

זהו מספר עצום! שבעים אלף מיליון מיליונים, מפתחות אפשריים!

הוא חסין פיצוח במונחי הקריפטוגרפיה (תורת ההצפנה) המסורתית. אבל למרות "גילו הצעיר" (ה- DES הוא בן 30 בלבד), כבר אין הוא בטוח במידה מספקת.

גם 70 קוודריליון מפתחות פוטנציאליים אינם מספיקים בימינו למניעת הפיצוח. ניתוח סטטיסטי של שכיחות האותיות, נשקם העיקרי של מפצחי מסרים מוצפנים במשך דורות, אינו מסייע

בהתקפה על שיטה כ - DES. זו שיטה כל כך מאובטחת עד שלמפצחים לא נותרת ברירה, פרט לתקיפת המפתח. כלומר ניסוי כל המפתחות האפשריים.

מדוע שיטת הפיצוח הזו (של ניסוי כל המפתחות), מקטינה את חסינותו של ה- DES לפיצוח כיוון שהיא תלויה במהירות הפיצוח של כל מפתח והתפתחות הטכנולוגיה מקטינה בהתמדה את משך הזמן של הפיצוח הממצה (Brute Force Attacks).כלומר,את הזמן הדרוש לפיצוח של כל המפתחות האפשריים.

נסביר זאת ביתר פירוט: המפצח אינו מכיר את המפתח, הוא מנסה לגלותו, לפצח אותו. כל ניסוי יכול להצליח אך יכול גם להיכשל. הסיכוי להצליח והסיכוי להיכשל שווים זה לזה. לכן מספר ניסיונות הפיצוח, שיהא עליו לבצע הוא בממוצע, מחצית ממספר המפתחות האפשריים. במקרה שלנו, 35 קוודריליון, 35 אלף מיליון מיליונים ניסיונות. לביצוע המשימה יש צורך במחשבי-על חזקים ויקרים. אולם מצב זה משתנה בהתמדה, כיוון שעוצמת המחשבים עולה ומחירם יחסית לה יורד.

ב- 1997 הצליחו לפצח את DES ב- 96 ימים. שנה לאחר מכן הצליחו לעשות זאת ב - 41 יום. ארבעה חודשים לאחר מכן הצליחו לפצח את DES ב- 56 שעות תוך שימוש במחשב, שעלותו הייתה רבע מיליון דולר. חצי שנה לאחר מכן בינואר 1999 חזר אותו צוות על הפיצוח, הפעם תוך פחות מ - 24 שעות.

כיום המפצחים טוענים כי מחשב שעלותו עשרה מליון דולר יפצח את DES במספר דקות.

5.1.3. AES

5.1.3.1. רקע של AES

תקן הצפנה מתקדם - Advanced Encryption Standard בקיצור AES הוא אלגוריתם הצפנה שאומץ על ידי המכון הלאומי לתקנים וטכנולוגיה (NIST) של ארצות הברית כתקן הצפנה רשמי להצפנה מאסיבית.

AES הוא צופן בלוקים סימטרי, בשמו המקורי ריינדל (Rijndael), שפותח על ידי הקריפטוגרפים הבלגיים יוהאן דאמן ווינסנט ריימן והוצע במהלך פרויקט בחירת התקן שאורגן על ידי NIST. הצופן אומץ על ידי ממשלת ארצות הברית באופן רשמי להצפנת נתונים מסווגים עבור הממשל והחליף בכך את קודמו DES שיצא לאור ב-1977.

אלגוריתם AES נמצא בשימוש מעשי נרחב בכל העולם הן בתוכנה והן בחומרה וידוע כאלגוריתם בטוח.

AES הוכרז כתקן הצפנה FIPS PUB 197 בנובמבר 2001 ואושר רשמית במאי 2002 על ידי מזכיר המסחר של ארצות הברית וכן נכלל בתקן איזו (ISO/IEC 18033-3). זהו הצופן הסימטרי הפומבי הראשון שקיבל את אישור הסוכנות לביטחון לאומי האמריקאי (NSA) כראוי להצפנת נתונים המוגדרים ברמת סיווג SECRET וכן TOP SECRET עבור ממשלת ארצות הברית, אם נעשה בו שימוש כחלק ממודול הצפנה מאושר.

5.1.3.2. מבנה הצופן

ריינדל הינו צופן בלוקים אשר מקבל בלוק P להצפנה בגודל 128 סיביות ומפתח K בגודל של 128, 192 או 256 סיביות ומייצר בלוק מוצפן בגודל 128 סיביות.

בהנתן בלוק (של P או של מפתח K) בן 128 סיביות (16 בתים) נוכל להסתכל על הבתים בו בתור טבלה, בצורה הבאה:

ריינדל הינו אלגוריתם איטרטיבי – ההצפנה בו נעשית במספר שלבים (10, 12 או 14 תלוי בגודל המפתח) כאשר בכל שלב מבוצעות מספר פעולות. כל פעולה מבוצעת על תוצאת הפעולה הקודמת.

5.1.3.3. סיכום

חשוב לשים לב לכך שהצופן עצמו אינו סודי ובסה"כ מורכב מפעולות די פשוטות .

כל אחד ללא ניסיון רב בתכנות יכול להוריד את המסמך המגדיר את הצופן מהאינטרנט ולממשו.

השילוב של כל הפעולות ביחד ומפתח הסודי (שחשוב לשים לב שזהו המרכיב היחידי

שסודי בהצפנה - כל יתר הפעולות מוגדרות היטב) הוא הדבר שיוצר את החוזק של הצופן .

כמו כן למספר השלבים בצופן יש השפעה רבה על חוזקו - ככל שמספר השלבים רב יותר כך קשה יותר לשבור את הצופן.

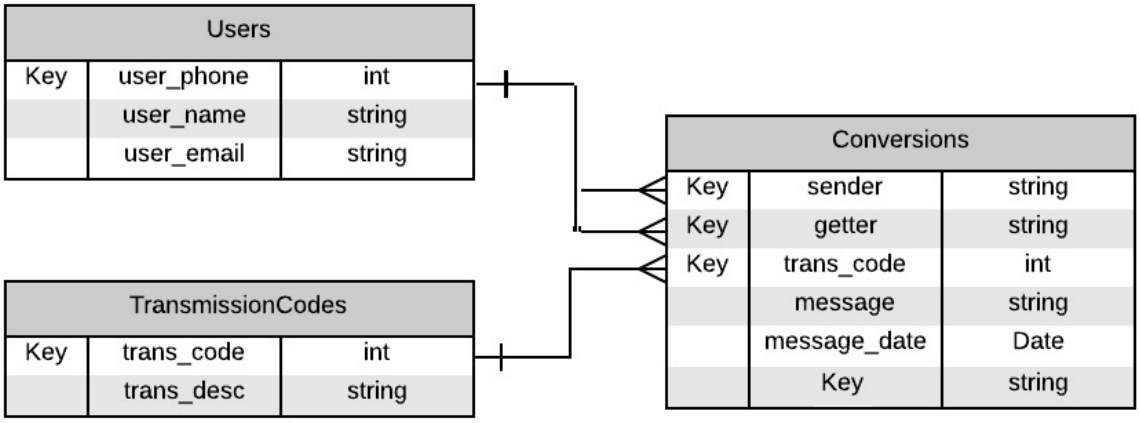
לאחרונה נמצאו התקפות על גרסאות ה-192 ביט וה-256 ביט של הצופן שסיבוכיותן טובה יותר מזו של חיפוש ממצה.

ההתקפות הללו אמנם עדיין אינן מעשיות – הן דורשות סיבוכיות גדולה (2 הפעלות של הצופן עבור גרסת ה-256 ביט) והן אפשריות רק בתנאים מסוימים מאוד, אולם התקפות כאלו מעידות על חולשות של הצופן ואולי אף מבשרות על מציאתן הקרובה של התקפות מעשיות.

בשנת 2012, טען דוקטורנט מאונ' ת"א הצליח לפרוץ הצפנת AES של 128 ביט בעזרת ניתוח צריכת המתח של המעבד.

אך אם נדייק, מדובר ב 342 שניות ל-10% הצלחה (שגם זו בפני עצמה התחלה מרשימה, סביר מאוד להניח שבקרוב גם AES של 128 ביט ייפרץ).

ERD .6



7. בדיקות

7.1 הגדרה

בדיקות תוכנה הוא תהליך הנועד לאפשר לבעלי העניין במוצר לקבל מדד לאיכותו ועמידתו בדרישות שהוצבו לו. כיום הנדסת בדיקות תוכנה הוא מקצוע נלמד באוניברסיטאות. בדיקות התוכנה כוללות כתיבת תסריטי בדיקות תוכנה ושימושים מתקדמים בכלי בדיקה אוטומטים. בדרך כלל המשימות להבטחת איכות משולבות לאורך מחזור הפיתוח השלם של היישום.

בפרק זה נרצה לקבל מדד לאיכות התוכנה (האפליקציה) ע"י תסריטי בדיקות וביצוען, הגדרת תכנית בדיקות למערכת ויישומה.

לשם כך נבצע את השלבים הבאים:

1. הגדרות ומונחים בנושא איכות התוכנה.
2. תכנון הבדיקות – כתיבת מסמך STP (Software Test Plan).

3. ביצוע הבדיקות והצגת התוצאות – כתיבת מסמך STD + STR
(Software Test Details + Software Test Results).

7.2. הגדרות ומושגים בנושא איכות תוכנה

7.2.1. מסמך Software Test Plan (STP)

מסמך תכנון בדיקות. זהו המסמך העיקרי עליו מבוסס תהליך ביצוע הבדיקות. בד"כ נכתב ע"י ראש הצוות. תכולת המסמך:

- תיאור המערכת - מה המערכת עושה ולמי מיועדת המערכת.
- מטרת הבדיקות - מה מטרת הבדיקות, מה התוצאה הנדרשת מהבדיקות, לשם מה מבצעים את הבדיקות, מהם ההישגים הנדרשים מהבדיקות. ישנה חשיבות גבוהה בהגדרת מטרת הבדיקות מכיוון ומכאן באה חשיבות הבדיקות במערכת.
- משאבי כוח אדם בבודקים - כמה בודקים יש, על פי איזו חלוקה, כיצד מנוהל הצוות וכו'.
- מקרי הבדיקה - מהם מקרי הבדיקה, מה ייבדק ובאיזו שיטה, מה ייכלל בבדיקות, אלו חלקים של המערכת יבדקו. לגבי כל מקרה יש לציין על איזו דרישה הבדיקה מתארת.
- מבנה לתכנון מקרי הבדיקות יכול להיות כמתואר בטבלה זו.

7.2.2. מסמך Software Test Description (STD)

מסמך תיאור הבדיקות. זהו מסמך אשר מתאר את השלבים בבדיקות, כיצד המערכת תיבדק בפירוט, המסמך הינו כלל השלבים בבדיקות, כיצד המערכת מבצעת את הפעולות בפירוט של הצעדים לביצוע כל פעולה או פונקציה. תכולת המסמך:

- הקדמה- מטרת המערכת, מונחים וראשי תיבות, הערות על המערכת, מסמכים קשורים למערכת. ההקדמה הינה סיכום כללי של מסמך ה STP.
- תיאור המערכת- חלק זה כולל את תיאור הרכיבים והמודולים של המערכת, תפקיד כל מודול, מה תפקיד המודול, דרכי הגעה למודול, לאיזו דרישה קשור המודול וכו'.
- דרישות לבדיקה- מהם הדרישות לבדיקות, אילו מהדרישות צריכות להיבדק.
- פירוט הבדיקות- מהי הבדיקה וכיצד הולכים לבצע אותה. חלק זה הוא החשוב ביותר והינו העיקרי במסמך. חלק זה מתאר את הבדיקות, קובע מהי התוצאה הנדרשת מכל פעולה וכן קובע את תהליך הבדיקות המלא.

7.2.3. הבדלים בין שני המסמכים STP ו STD

- ההבדל העיקרי הינו בנתונים המוצגים במסמך.
- במסמך ה-STP מופיע תיאור כללי של הבדיקות, בחירת מתודולוגיות לבדיקה וכו'.
- במסמך ה-STD מופיעים תסריטי הבדיקות עצמם- כיצד כל בדיקה אכן תתבצע.
- מסמך ה-STD הינו מסמך אשר נבנה על סמך מסמך ה-STP ומסמך האפיון.

7.2.4. חשיבות המסמכים

- למסמכי הבדיקות חשיבות גבוהה שעל כן הם קובעים את הסטנדרטים לביצוע הבדיקות.
- תיעוד דורש זמן עבודה, אך עם זאת חוסך זמן לאורך הפרויקט ובזכותו תחזוקת המערכת אפשרית וקלה יותר.
- בקרה על ביצוע הבדיקות – כך יהיה ניתן לדעת מה נבדק וכיצד.

7.2.5. שלבים מרכזיים בבדיקות תכנה

קיימים 4 שלבים מרכזיים בבדיקות התוכנה.

1. **<u>בדיקות יחידה (Unit)</u> -** בדיקות ברמת יחידת תוכנה (מודול). לרוב מבוצעות על ידי מפתח התוכנה. זוהי היחידה הקטנה ביותר שניתנת לבדיקה, או אפילו חיבור לוגי של כמה יחידות. הן הבדיקות היחידות שבאחריות המלאה של התכניתן. באחריותו להכין את רשימת הבדיקות עבור המודול/תהליך אותו הוא מממש, לבצע את הבדיקות ולמסור את המודול לאינטגרציה רק לאחר שעבר את בדיקות היחידה במלואן. בדיקות יחידה מכילות בתוכן שני סוגי בדיקות ,כאשר כל בדיקה משלימה את השנייה:

   - בדיקות קופסה שחורה (Black Box) - בדיקות אלו אינן מכירות את המבנה הפנימי של המערכת ומתבססות על בדיקת הפלט הצפוי לקלט מסוים בהתאם לתכנון מוקדם כלשהו. בדרך כלל, בדיקות קבלה מבוצעות בשיטה זו. בסוג בדיקות זה, הבודק חייב לדעת את פירוט דרישות המערכת, וכן, עליו לדעת לאיזה פלט מהתוכנה עליו לצפות עבור קלט מסוים. עם זאת, הבודק אינו חייב להכיר את הלוגיקה של הבעיה או אפילו לדעת את שפת התכנות בה היא כתובה.

   - בדיקות קופסה לבנה (White Box) - בדיקות אלו משלימות את בדיקות הקופסה השחורה. מתבססות על הכרת קוד המקור של התוכנה ובניית תוכניות בדיקה

המותאמות לנתיבי הזרימה האפשריים של הקוד. בדיקות יחידה עשויות להיות בדיקות מסוג קופסה לבנה. בסוג בדיקות זה, על הבודק להכיר את הלוגיקה של הקוד, וכן, עליו להיות בעל ידע בשפת התכנות בה כתובה התוכנה.

ברמת בדיקת המודול, בדיקות הקופסה הלבנה מיועדות:

1. לבחון את המבנה הפנימי של המודול .
2. לבחון את מסלולי החישוב.
3. לבחון את נכונות החישובים (חישובי הביניים).
4. לבחון את נכונות ההחלטות הלוגיות.

2. **בדיקות אינטגרציה** (Integration) - בדיקת שילוב יחידות תוכנה בהיקפים שונים, החל משתי יחידות ועד לכלל היחידות במערכת.

3. **בדיקות מערכת** (System) - בדיקות המערכת בכללותה, בדרך כלל בראיית המשתמש של יכולות המערכת.

4. **בדיקות קבלה** (Acceptance) - בדיקות הנעשות על ידי המשתמש או הלקוח במטרה לוודא כי המערכת פועלת בהתאם לדרישות שהוגדרו במסמך הדרישות המקורי ובשינויי דרישה (change request) שהועברו במהלך מחזור חיי הפיתוח.

7.2.6. אסטרטגיות הבדיקות

- **בדיקות פונקציונליות** (Functional) – בדיקות אלו נועדו לאימות פעילות המערכת. בדיקות אלו מבוססות על מסמך הדרישות ומסמך האפיון ומטרתן לבדוק כי המערכת עושה את מה שהיא צריכה ולא עושה את מה שאינה צריכה לעשות (valid and invalid testing). בדיקות פונקציונליות יכתבו ברמת פירוט גבוהה ביותר .
- **בדיקות לא פונקציונליות** (Nonfunctional) - בדיקות אלו בודקות איך פועלת המערכת וכוללות בדיקות עומסים, ביצועים, שימושיות וסוגי בדיקות רבים נוספים .
- **בדיקות ממשק לקוח** (GUI) - בדיקות הפקדים והשדות במסך. התנהגות תקינה, פורמט של שדות, בהתאם לחוקיות המוגדרת ברמת המסך ולא הלוגיקה העיסקית. לדוגמה: בדיקת מינימום ומקסימום תווים בשדה.

- **<u>בדיקות תהליכיות</u>** - בדיקות הכוללות רצף של פעולות פונקציונאליות – רצף של פעולות לאורך ציר הזמן. בבדיקות תהליכיות מתבצע תהליך שבו מצב המערכת משתנה כתלות בזמן. בבדיקות תהליכיות אנו למעשה בודקים את האינטגרציה של כל הפעולות אחת לשנייה.
- **<u>בדיקות נסיגה</u>** (Regression) - לאימות פעילות המערכת לאחר שבוצעו בה שינויים. לוודא שמה שעבד לא התקלקל בעקבות העברת גרסה.
- **<u>בדיקות הרשאות</u>** – בדיקת הרשאות מתבצעות בשני הכיוונים:

2. לוודא שהפעולות המותרות אכן מתאפשרות לקבוצת המשתמשים הספציפית.
3. לוודא שהפעולות האסורות אכן לא מתאפשרות לקבוצה הספציפית.

7.2.7. אופי הבדיקות

- **בדיקות תוכנה ידניות (Manual Testing)** - בדיקות תוכנה הנעשות על ידי עובד שהוכשר לכך בדרך כלל על פי תכנית בדיקות מסודרת ומוסכמת.
- **בדיקות תוכנה ממוכנות (Automation Testing)** - בדיקות תוכנה הנעשות בצורה מכנית, רצוי עם מינימום התערבות אנושית

7.3. מסמך STP עבור הפרויקט

7.3.1. תיאור המערכת

המערכת הינה אפליקציה המפותחת בשפת Java בסביבת Eclipse ורצה על android.
לאפליקציה קיים רישום ראשוני ואחריו ישנה אפשרות לשימוש בה.
לשליחת המסרונים בצורה מאובטחת אנו מצפינים את המידע ושולחים אותו מוצפן דרך שירות של google לשליחת הודעות בandriod ופענוח המסרון מתבצע במכשיר המקבל אותו.

7.3.2. מטרת הבדיקות

חשיבותה של בדיקת מוצרים עתירי תוכנה הולכת וגוברת, ולכן בדיקות התוכנה הן התחום המתפתח ביותר בעולם ההיי-טק בשנים האחרונות. תחום זה הוא מרכיב חשוב ביותר והכרחי בפיתוח תוכנה או מוצרים טכנולוגיים. חברות רבות בארץ ובעולם הכירו בכך שנושא איכות בדיקות התוכנה חיוני להבטחת האמינות, הזמינות, הביצועים ואפשרויות הגידול והשיפור של מוצריהן, ולכן הוא משפיע ישירות על מכירתם ועל רווחי החברות.

בפרויקט שלנו , מטרות הבדיקות הן:

- הגדרת הפעילות הנדרשת באפליקציה.
- ביצוע כלל הדרישות תוך התחשבות במקרי קצה הרבים ככל הניתן.
- ניסוי כלל הבדיקות האפשריות בכדי לאפשר את דרישות המערכת כפי שפורטו.

7.3.3. מרחב הבדיקות

בדיקות ספציפיות יפותחו כדי לבדוק את הפונקציונאליות הבסיסית של המערכת ולבצע בדיקות של הוספה/ עריכה והסרה של נתונים אל מול בסיס הנתונים. כיוון שרוב הבדיקות חוזרות על עצמן הוחלט לפרט כאן באופן מדגמי.

7.3.4. דרישות סביבת עבודה לבדיקות

- מערכת הפעלה android 2 ומעלה.

7.3.5. הנחות

קיימים משתמשים שונים רשומים למערכת, למשתמשים יש כמה שיחות צ'ט מהסמארטפון שלהם אל סמארטפון אחר.

7.3.6. בדיקות

STP	
מספר בדיקה	1
מקרי הבדיקה	פתיחת האפליקציה והעלאת מסך פתיחה.
ידנית/אוטומטית	ידנית.
חשיבות	גבוהה.
הערות	מסך פתיחה ייפתח
STD & STR	
מספר בדיקה	1
תיאור הבדיקה	פתיחת האפליקציה והעלאת מסך פתיחה.

מטרת הבדיקה	עלינו לוודא כי המערכת עולה באופן תקין.			
צעדים לביצוע:	תיאור הצעד	תוצאה צפויה	עבר / נכשל	מה התקבל בפועל
1	לחץ על אייקון האפליקציה.	מסך טעינה ייפתח	עבר	

STP	
מספר בדיקה	2
מקרי הבדיקה	התחברות ראשונה תקינה למערכת
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	עלינו לוודא שהמשתמש מתבקש לכתוב רק שם ומספר הטלפון נשלף אוטומטית מהסים

STD & STR				
מספר בדיקה	2			
תיאור הבדיקה	התחברות תקינה למערכת			
מטרת הבדיקה	עלינו לוודא כי רק משתמשים מורשים יכולים להיכנס למערכת.			
צעדים לביצוע:	תיאור הצעד	תוצאה צפויה	עבר / נכשל	מה התקבל בפועל
1	הזן שם משתמש	נכנס למערכת והמשתמש ירשם	עבר	

STP	
מספר בדיקה	3
מקרי הבדיקה	כניסה לא בפעם הראשונה למערכת
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	עלינו לוודא כי המשתמש כבר קיים במערכת

STD & STR

מספר בדיקה	3			
תיאור הבדיקה	התחברות תקינה למערכת			
מטרת הבדיקה	עלינו לוודא כי המשתמש כבר קיים במערכת.			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**
1	הזן שם משתמש קיים	כניסה לאפליקציה ללא שמירה מחודשת של שם המשתמש	עבר	

STP	
מספר בדיקה	4
מקרי הבדיקה	צפייה במסך שיחות
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא שבסמך השיחות מופיעות כל השיחות הקודמות

STD & STR				
מספר בדיקה	4			
תיאור הבדיקה	צפייה במסך שיחות			
מטרת הבדיקה	יש לוודא שבסמך השיחות מופיעות כל השיחות הקודמות			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**
1	לאחר כניסה דרך המסך הראשי נגיע למסך השיחות	נוכל לראות את כל השיחות	עבר	

STP	
מספר בדיקה	5
מקרי הבדיקה	צפיה בשיחות לפי סוג הצפנה
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא כי כל אחת מההצפנות מופרדת ע״י חץ בצבע שונה.

STD & STR				
מספר בדיקה	5			
תיאור הבדיקה	צפיה בשיחות לפי סוג הצפנה			
מטרת הבדיקה	יש לוודא כי כל אחת מההצפנות מופרדת ע"י חץ בצבע שונה.			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**
1	כניסה למערכת וצפייה במסך שיחות	יופיעו לנו כל השיחות כך שכל הצפנה תהיה בעלת חץ בצבע שונה.	עבר	

STP	
מספר בדיקה	6
מקרי הבדיקה	פתיחת מסך הוספת שיחה
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא כי מסך הוספת שיחה נפתח.

STD & STR				
מספר בדיקה	6			
תיאור הבדיקה	פתיחת מסך הוספת שיחה			
מטרת הבדיקה	יש לוודא כי מסך הוספת שיחה נפתח.			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**

1	כניסה למערכת, ובמסך השיחות לחץ על הוספת שיחה	יפתח מסך הוספת שיחה	עבר	

STP	
מספר בדיקה	7
מקרי הבדיקה	הוספת שיחה
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא שהשיחה אכן מתווספת.

STD & STR				
מספר בדיקה	7			
תיאור הבדיקה	הוספת שיחה			
מטרת הבדיקה	יש לוודא שהשיחה אכן מתווספת.			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**
1	כניסה למערכת ולחיצה על הוספת שיחה במסך הוספת שיחה יש להוסיף שם, טלפון, וסוג הצפנה.	נחזור למסך השיחות ותתווסף לנו שיחה לפי הנתונים שבחרנו	עבר	

STP	
מספר בדיקה	8
מקרי הבדיקה	אכיפת הכנסת אותיות במספר הטלפון במסך הוספת שיחה
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא כי אפשרות הוספת האותיות למספר הטלפון חסומה.

STD & STR	
מספר בדיקה	8

תיאור הבדיקה	אכיפת הכנסת אותיות במספר הטלפון במסך הוספת שיחה			
מטרת הבדיקה	יש לוודא כי אפשרות הוספת האותיות למספר הטלפון חסומה.			
צעדים לביצוע:	**תיאור הצעד**	**תוצאה צפויה**	**עבר / נכשל**	**מה התקבל בפועל**
1	הכנס לאפליקציה	מסך השיחות יופיע	עבר	
2	בחר הוספת שיחה	מסך הוספת שיחה יופיע	עבר	
3	בחר במספר טלפון "אבג"	לא תינתן אפשרות כתיבה שאינה נומרית בשדה זה.	עבר	

STP	
מספר בדיקה	9
מקרי הבדיקה	חזרה למסך שיחות
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא שחזרנו למסך שיחות.
STD & STR	
מספר בדיקה	9
תיאור הבדיקה	חזרה למסך שיחות

מטרת הבדיקה	יש לוודא שחזרנו למסך שיחות			
צעדים לביצוע:	תיאור הצעד	תוצאה צפויה	עבר / נכשל	מה התקבל בפועל
1	כניסה למערכת	מסך שיחות יופיע	עבר	
2	לחץ הוספת שיחה	מסך הוספת שיחה יופיע	עבר	
3	לחיצה על חזרה	מסך שיחות יופיע	עבר	

STP	
מספר בדיקה	10
מקרי הבדיקה	כניסה למסך שיחה ספציפי
ידנית/אוטומטית	ידנית
חשיבות	גבוהה
הערות	יש לוודא שנכנסנו למסך של השיחה אותה רצינו

STD & STR				
מספר בדיקה	10			
תיאור הבדיקה	כניסה למסך שיחה ספציפי			
מטרת הבדיקה	יש לוודא שנכנסנו למסך של השיחה אותה רצינו			
צעדים לביצוע:	תיאור הצעד	תוצאה צפויה	עבר / נכשל	מה התקבל בפועל
1	כניסה לאפליקציה	מסך שיחות יופיע	עבר	
2	נבחר שיחה ספציפית	מסך השיחה הספציפי יופיע	עבר	

8. ניהול עבודה בצוות

את האפליקציה פיתחנו בצוות של שני מפתחים כך שהעבודה צריכה הייתה להיות מחולקת בצורה ברורה, בכדי שלא תבוצע עבודה כפולה.

ולכן העבודה חולקה בעזרת הנושאים הבאים:

1. חלוקת העבודה לתתי משימות לפי מודולים.
2. חלוקת תתי המשימות בין חברי הצוות והערכת זמנים לביצוע המשימות.
3. אינטגרציה בין תתי המשימות למוצר שלם.

בעזרת חלוקה זו – הצלחנו לעבוד במקביל ולפתח את המערכת בזמן קצר יחסית.

8.1 מתודולוגיית פיתוח תוכנה

המונח "מתודולוגיית פיתוח תוכנה" בהנדסת תוכנה הוא אוסף מוסכם של עקרונות, תהליכים, פעילויות וכלים על פיהם מפותחות ומתוחזקות מערכות תוכנה. כיום, קיימות כתריסר מתודולוגיות עיקריות, וכן מאות אחרות - משניות. המתודולוגיות נבדלות ביחסן למקצוע הנדסת התוכנה ועקרונות היסוד, המיקוד (ניהול פרויקט, ארכיטקטורה, עיצוב ותכנות, הבטחת איכות), הטכניקות ובהיבטים נוספים. בשל גילו הצעיר של הענף ובשל ריבוי המתודולוגיות אין עדיין הסכמה באשר למידת התאמתן של מתודולוגיות מסוימות לבעיות מסוימת, אם כי מקובל לחלק את המתודולוגיות למשפחות נפרדות.

המתודולוגיה המוכרת ביותר בשנות ה-90 ועד לתחילת שנות ה-2000 הינה מתודולוגיית Water fall("מפל המים") אשר בה המוצר תוכנן לפני תחילת הפיתוח, ומרגע תחילת הפיתוח התבצעו תהליכים פנים ארגוניים לצורך הרמת המערכת לאוויר ללא ממשק מול הלקוח ועדכונו – מרגע קבלת הפרויקט עד לתאריך סיומו שנקבע מראש, הפרויקט התקדם בתוך החברה המפתחת.

מתודולוגיה חדשה שבשנים האחרונות נכנסה כמתודולוגיה מועדפת ולה יש גרסאות רבות ומגוונות, אך עם דגש אחד עיקרי – גמישות לשינויים, היא מתודולוגיית הAgile.

8.1.1 פיתוח תוכנה בעזרת מתודולוגיית Agile

המשמעות המילולית למילה "Agile" היא מהיר, קל תנועה, גמיש לשינויים.

מתודולוגיה אשר הותאמה לפיתוח תוכנה בצוותים קטנים תוך שימת דגש על יעילות, זריזות ואיכות.

- בשנים האחרונות, תפס תאוצה מודל חדש לפיתוח תוכנה, ששמו Agile Programming. המודל הפופולארי מהווה תחליף מרענן למתודולוגיות פיתוח קלאסיות, שיכולות להיות לעיתים די מייגעות. לאור ריבוי מקרים של פרויקטים כושלים בתחום התוכנה, הן בלוחות הזמנים והן בתכנון הפרויקטים נהגתה מתודולוגיית פיתוח חדשה ששמה Agile. עקרונותיה של מתודולוגיה זו מובילה ליתרונות הבאים:
- **קיצור בזמן הפיתוח** – הקמת צוות פיתוח ממוקד מטרה. הקמת צוות פיתוח אשר כולל מפתחי ובודקי תוכנה יחדיו, וזאת על מנת להעצים את המטרה המרכזית שלשמה נבנה הצוות. מיקוד - קיום ישיבות סטאטוס יומיות קצרות ביותר (10 דקות בערך) על מנת לבדוק את סטאטוס התקדמות הפרויקט, חיבור בין אנשי הצוות, הבנה האם הצוות נתקל בקושי אשר מעקב את פיתוח הפרויקט או כל נושא אחר. התייחסות אל מסמכים כאל אמצעי לקידום המטרה ולא המטרה עצמה. דגשים אלו ואחרים נועדו על מנת לקצר ולייעל את פיתוח הפרויקט.
- **שיתוף פעולה מלא עם הלקוח לכל אורכו של הפרויקט** – הלקוח הוא המטרה. שיתוף הלקוח נעשה מתחילת הפרויקט ועד סופו. הלקוח מעורב בכל שלב של הפרויקט. הלקוח יוכל בכל שלב של הפרויקט לקבל מוצר חלקי.
- **אפשרות גמישה יותר לעריכת שינויים** – הבנה שניתן לערוך שינויים בפרויקט תוך כדי עבודה. אם בעבר נהגו לבצע פרויקטים ארוכי טווח אזי כיום הפרויקטים הינם קצרי טווח ומתפרשים על פני אבני דרך. על פי מתודולוגיית ה Agile במהלך פיתוח פרויקט יש לבצע כמה סבבי פיתוח. כל סבב פיתוח נקרא ספרינט והוא ימשך לא יותר משלושה שבועות. כך שכל תקופה של שלושה שבועות יבחן הפרויקט מחדש הן מצד הלקוח והן מצד העובדים ומנהל הפרויקט.

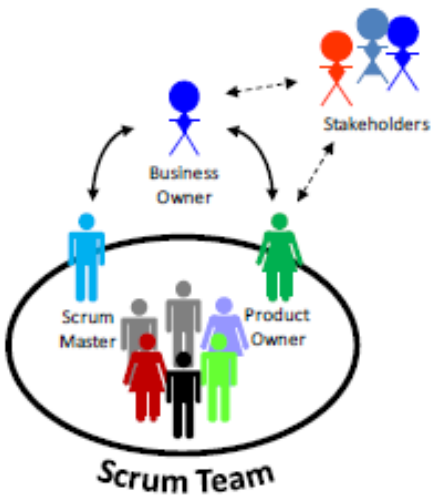
8.1.2. Scrum

Scrum היא מתודולוגיה שפותחה באמצע שנות ה-90 על ידי קן שוואבר וג'ף סאתרלנד. השיטה מתבססת על ההנחה שפיתוח תוכנה הוא בעיה אמפירית ולא ניתן לפתור אותה בשיטות מסורתיות המתבססות על חיזוי או תכנון. Scrum הוא תהליך "אג'ילי" לניהול פרויקטים לפיתוח תוכנה המאפשר לנו להתמקד בהפקה של ערך עסקי גבוה בזמן הקצר ביותר. כן גם, תהליך זה מאפשר לנו בזריזות ובמחזוריות לבחון את התוכנה במצב עובד (כל שבועיים עד חודש). הגוף

העסקי מספק סדרי עדיפויות. הצוותים מנהלים את עצמם על מנת להבין כיצד לספק בצורה הטובה ביותר את הדרישות בעלות העדיפות הגבוהה ביותר. קבוצת העבודה ב Scrum מכילה מפתחים, בודקים ובעלי מקצוע אחרים ומנוהלת עצמאית באחריות משותפת. בכל שבועיים עד חודש כל אחד יכול לראות תוכנה עובדת ולהחליט לשחרר אותה או להמשיך לשפר אותה בספרינט נוסף.

מאפיינים עיקריים:

- Product Backlog (עתודת המוצר) - תחזוקה של רשימת פריטי העבודה לביצוע, מסודרים לפי קדימויות.
- Sprint - השלמת מנה קבועה של פריטי עתודה בסדרה של איטרציות קצרות הנמשך 4 שבועות, ובסיומו מסופקת תוכנה עובדת למשתמשים.
- Daily Scrum – פגישת צוות יומיומית קצרה, בפגישה מציג כל אחד מחברי הצוות את ההתקדמות, העבודה המתוכננת וקשיים אפשריים. הפגישה מתקיימת לרוב בעמידה.



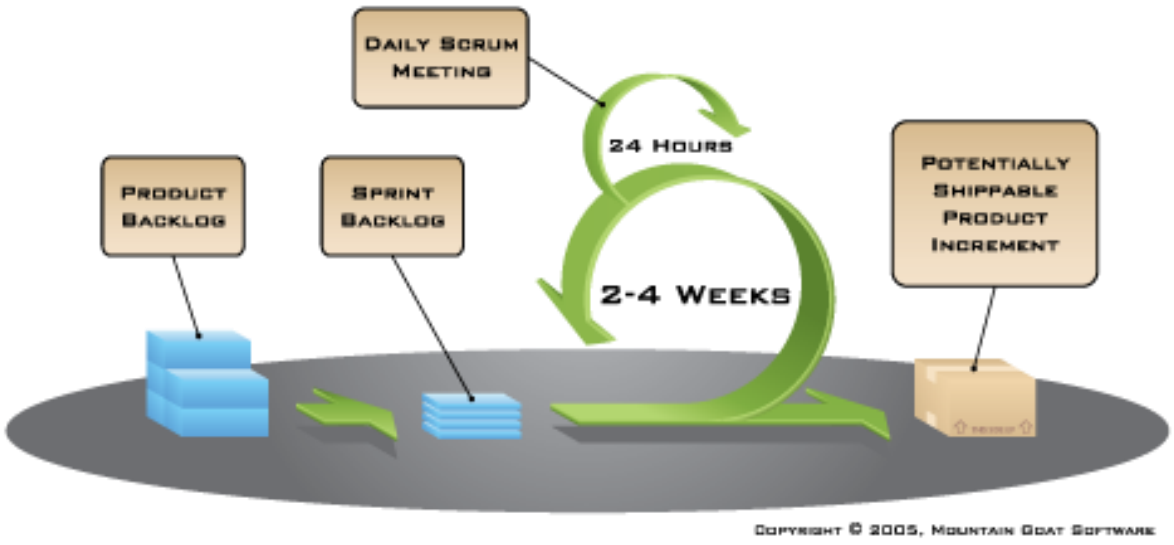
- Sprint Planning - פגישת תכנון ספרינט שבה מוגדרים פריטי העתודה לאותו ספרינט.
- Sprint Retrospective - פגישת ניתוח ספרינט קצרה להפקת לקחים מהספרינט הקודם.
- השיטה מיושמת בהנחיית Scrum Master שתפקידו העיקרי לסלק מכשולים המפריעים לצוות לעמוד ביעדי הספרינט. ה-Scrum Master אינו ראש הצוות (מאחר שמדובר בצוות בהכוונה עצמית), אלא חוצץ בין הצוות לבין השפעות העלולות להפריע לו.

- כדי לאפשר את יצירתם של צוותים בהכוונה עצמית, השיטה מעודדת ריכוז של כל חברי הצוות במיקום אחד, וכן **תקשורת מילולית** בין חברי הצוות ועם צוותים תומכים.

אחד מעקרונות המפתח של Scrum הוא הכרה בכך שאתגרים אמפיריים ביסודם אינם ניתנים לפתרון בשיטות מסורתיות המתבססות על חיזוי או תכנון. מכיוון שכך, שיטת Scrum מאמצת גישה אמפירית, המניחה שלא ניתן להבין או להגדיר את הבעיה במלואה מראש. במקום זאת, השיטה מתמקדת בשיפור יכולתו של הצוות לספק תוצרים במהירות ולהגיב לדרישות העולות תוך כדי התהליך.

8.1.3. חלוקת תפקידים וניתוח העבודה של חברי הצוות

באיור הבא מתואר השלב ההתחלתי בתהליך ביצוע הפרויקט. בכל פרק זמן של בין 2-4 שבועות, כל אחת מחברות הצוות קיבלה משימה לבצע. כל שבוע נפגשנו ודנו בבעיות שעלו, העלינו נושאים חדשים שיש להתחשב בהם כחלק מהפרויקט וביצענו אינטגרציה. איטרציה זו נמשכה עד שהמוצר הושלם. לא נשארו משימות לבצע.



9. סביבת פיתוח

9.1. שפת פיתוח

את האפליקציה פיתחנו באמצעות Eclipse בגרסת Kepler (העדכנית ביותר) בשפת Java.

ביצענו התקנת Android 4.4.2 SDK בגרסת kit-kat בחרנו בגרסה הזו מכיוון שהיא כמעט החדשה ביותר והיא תומכת אחורנית עד גרסאות 2.*.

בנוסף, התקנו AVD (Android Virtual Device) שעליו אמורים היינו לעבוד ולדבג, בעת הצורך, את האפליקציה, אך נתקלנו בתקלות לאין ערוך, איטיות נוראית ובקשיי הגדרה שלעיתים אף לא קיבלו תשובה כלל.

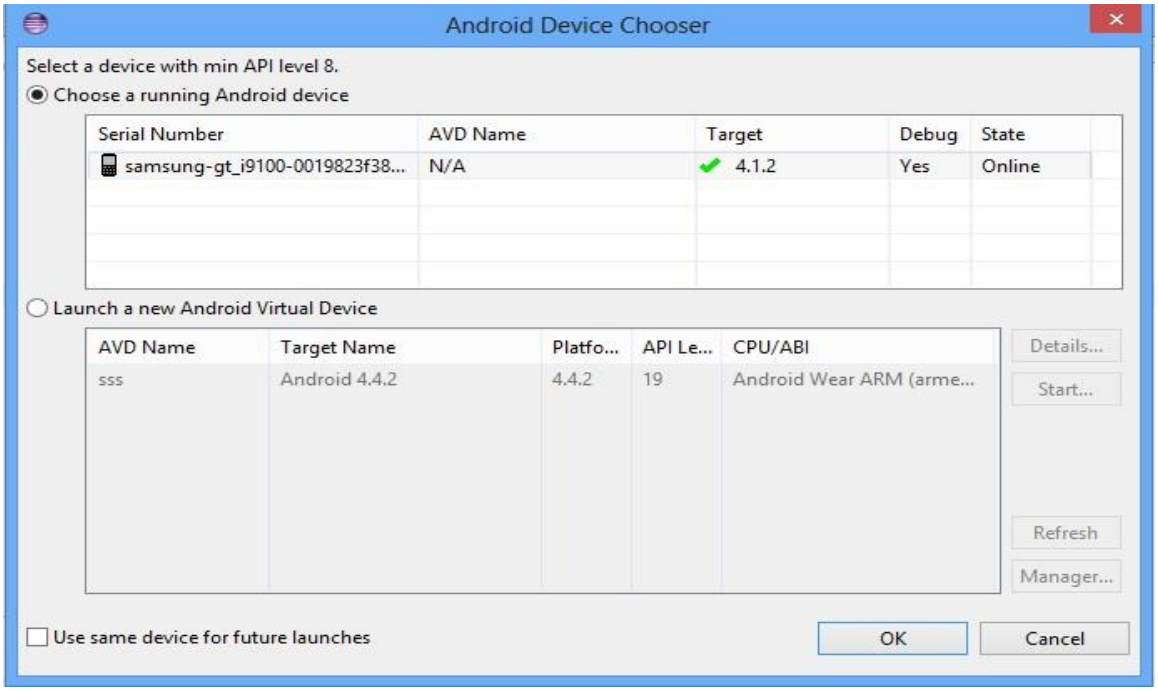
לכן, את מהלך פיתוח האפליקציה ומצב הדיבאג כשהזדקקנו לו, ביצענו דרך המכשיר עצמו.

פתחנו את המכשירים שעליהם עשינו את מצב הדיבאג ל"אפשרויות למפתחים" – תפריט נסתר שכלול במכשירי האנדרואיד שעל מנת לפתוח אותו יש להקיש מספר פעמים (7) על גרסת המכשיר ובכך נפתחת האפשרות. בתוך תפריט זה יש אפשרות לפתוח את המכשיר למצב debug שאפשר והתאים לצרכים שלנו.

את האפליקציה הרצנו על מספר מכשירים שונים, בניהם הנפוצים ביותר (Samsung Galaxy s2, Samsung Galaxy s4, Asus Google Nexus 7

דוגמא ל AVD - הרצת האפליקציה על המכשיר או על מכשיר דמה (על המחשב עצמו):

בטבלה העליונה יהיו המכשירים המחוברים למחשב ובטבלה האחרונה יהיו מכשירי הדמה שהוגדרו ועליהם אפשר להריץ את האפליקציה.



9.2. טכנולוגיית פיתוח צד שרת

ראשית, נציין מהו "צד שרת" ולמה אנחנו מתכוונים כשאנו כותבים אפליקציה מורכבת.
אפליקציה, מאפשרת למשתמש האנדרואיד להציג מידע אינטראקטיבי בצורות שונות ובתחומים שונים. בחלק מהמקרים, כל המידע או הזיכרון הדרוש להפעלת האפליקציה ידוע לנו מראש ויכול להישמר מקומית בתוך האפליקציה בזיכרון של האנדרואיד.
אולם, במקרים אחרים, אנו זקוקים למידע מעבר למידע שניתן לקבוע מראש ולשומרו באפליקציה ולכן נצטרך בדרך כלל מקום חיצוני שישמור וינהל את המידע הזה.

המקום שמנהל את הנתונים החיצוניים של האפליקציה הוא השרת, והחלק בפיתוח שמכין את השרת ומנהל את הנתונים הוא "פיתוח צד השרת". כאשר אנו רוצים לפתח אפליקציה מורכבת המכילה שרת, אנו צריכים להחליט ולתכנן אילו נתונים אנו רוצים לשמור, איך לשמור אותם, ואיך לנהל את הנתונים הללו.

התרומה העיקרית של פיתוח בצד שרת הינה האפשרות לעדכן מסד נתונים אחד שעליו יעדכנו וממנו ישלפו נתונים כלל המכשירים המשתמשים באפליקציה ברחבי העולם.

כדי להשתמש בצד שרת יש לרכוש או לחילופין להירשם לאחד מהאתרים התמוכים ומספקים צד שרת ראוי, יש ללמוד את דרך השימוש ואת דרך החיבור לאפליקציה.

בנוסף, כמו שנסביר בהמשך, סקרנו גם את עולם התוכן ואת ההיצע של עולם האנדרואיד ספציפית בכל מה שקשור למסד הנתונים בהתמקדות על SQLite ועל המעטפת שלו ORMLite.
SQLite הינה טכנולוגית מסד נתונים מקומית המבוססת על קבצי מערכת באנדרואיד.
התרומה העיקרית של כלים אלו היא פשטות השימוש ומהירות הכלי, פחות חיבורים מסובכים ויותר עבודה מהירה ופשוטה מול מסד הנתונים אך אלו אינם מסדי נתונים חזקים במיוחד ואינם תומכים בכל מבני הנתונים ואינם בעלי יכולות מיוחדות.

בסופו של תהליך, ובעקבות הצרכים הספציפיים של האפליקציה שלנו, בחרנו להשתמש ב SQLite ע"י המעטפת שלו ORMLite שאפילו מפשט את השימוש עוד יותר.

9.3. טכנולוגיית מסד נתונים

על מנת שנוכל לבחור מסד נתונים נכון הן מבחינה ביצועית והן מבחינת מענה לבעיה שלנו, חקרנו את נושא מסד הנתונים באנדרואיד בשלל כיוונים.

תחילה, עשינו סקר כללי לגבי מסדי הנתונים הנפוצים והארכיטקטורה שלהם.
בחנו כלים רבים החל משמירה ע"י קבצים ועד MySQL עד שהחלטנו על הדרך הנכונה עבורנו שעונה על הצרכים לאפליקציה שלנו.
בחנו את האפשרות לעבוד דרך DB חיצוני ולבצע חיבור בינו לבין האפליקציה.
כלל הכלים שנבחנו הינם כלים חינמיים ובעיקרם תומכים בopen source ובפשטות ההגדרה.
החקר שלנו התמקד ב-3 חלקים עיקריים: כלי ניהול מסדי נתונים נפוצים, פיתוח צד שרת ומבני נתונים באנדרואיד.

9.3.1. ניהול מסד נתונים על ידי כלים נפוצים וחיבורם לאפליקציה:

בחנו מספר כלים והחלטנו לבחון יותר לעומק את הכלי MySQL של חברת ORACLE.
MySQL הוא מסד נתונים יחסי, רב נימי ורב משתמשים מבוסס שפת SQL (Structured Query Language).
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.
MySQL יכולה לפעול על מספר רב של פלטפורמות: Solaris, Windows , Unix ועוד..
בנוסף הכלי דרש דרייבר(JAR) לחיבור בין אפליקציה לשרת ה DB כמו שנהוג כיום.
כלי פשוט יחסית לשימוש, התומך במספר משתמשים רב ויודע לתת ביצועים גבוהים יחסית.

9.3.2. SQLite

SQLlite הינו מסד הנתונים באנדרואיד.
הוא לא מאפשר יכולות מלאות כמו לדוגמא ב SQL Server אך יש לו יכולות כמו לשלוח Transactions ו Prepared Statements.

היתרון הבולט שלו הוא שהוא פשוט לשימוש והוא דורש רק מעט זיכרון בזמן ריצה ולכן, בגלל המגבלות בזיכרון שישנם במכשירי אנדרואיד הוא מתאים לפיתוח עליהם.

ה SQLite תומך במבני הנתונים: Text, Integer, Real בלבד ואין בו type checking.

בתחילת התוכנה יש ליצור את מסד הנתונים ע"י כתיבת פקודות בשפת SQL בסיסית ליצירת הטבלאות, המפתחות, האינדקסים וכו'.
לאחר מכן מסד הנתונים מנוהל ע"י מערכת ההפעלה של אנדרואיד.

כאשר אנו יוצרים מסד נתונים הוא נשמר במערכת הקבצים של אנדרואיד בתיקייה:
data/data/<your application package name>/database/<your database name>
זוהי תיקייה שרק למערכת ההפעלה יש גישה אליה, ולכן הגישה אליה חסומה אלא אם כן המכשיר פרוץ.
המידע בקובץ הDB ישמר כל עוד לא עשינו uninstall או clear data לאפליקציה שלנו.

הדרישות שלנו ממסדר הנתונים היו:
ליצור אותו בפעם הראשונה שמפעילים את האפליקציה וזקוקים למסד הנתונים.
להשתמש בו בפעמים הבאות שנפעיל את המערכת ונצטרך להשתמש בו.
להיות מסוגלים לשדרג ולשנות אותו כשנרצה.

9.3.3. ORMLite:

Object Relational Mapping Lite - זוהי תשתית אפליקטיבית מובנית המחברת מחלקות בJAVA למסד נתונים בסביבת האנדרואיד. הבסיס נתונים ממש את צורת הפיתוח המקובלת בסגנון JPA או מימושה – Hibernate , שכן מייצגת טבלאות כאובייקטים ובכך מונעת גישות JDBC בעלות ערך O/I גבוהה וכן עושות מספר פעולות על גבי טרנזקציה בודדת.

ORMLite תומכת בחיבור JDBC ל MySQL, PostgreSQL, Microsoft SQL Server, H2, Derby, HSQLDB וכמובן לSQLite ויכולה לתמוך במסדי נתונים רלאציונים נוספים בקלות.
ORMLite יודעת לעבוד עם DB2 וגם עם ORACLE.

השימוש בORMLite הינו פשוט ביותר ומכיל:
1. הכנת מחלקות היוצרות את הטבלאות (טבלה, עמודות, טיפוסים, גדלים וכו')
2. הכנת מחלקות המקשרות והשולפות את הנתונים ממסד הנתונים ע"י שימוש בשפת Java

(-getים ו set-ים).

מכאן והלאה – השימוש פשוט ביותר ואינו מצריך דבר פרט לקריאה ל-get ים ו set-ים הרצויים.

10. קישוריות

נושא הקישוריות מהווה נידבך משמעותי מהפרויקט שכן מטרתו לספק יכולת תקשורת בסיסית בין משתמשי המכשיר וכן לעמוד בריבוי תמסורת ללא קריסה וללא איבוד מידע. בנוסף על תווך התקשורת מתבצעת התקיפה. כמו שתואר מטרת הפרויקט אינה הענקת חסינות מלאה לרשת ולכן הפרויקט לא לוקח בחשבון פריצה למנהלן/הרשאות חזקות של התוקפים.

נבחרו ואף מומשו(בrevisionים שונים של הפרויקט) שני חלופות עיקריות:

1. תקשורת על בסיס socketים :

   מימוש socketים מובנה ב JAVA ובעל יתרון של מימוש פשוט ללא ממשקים חיצונים מהם נרצה להימנע בשל היותם נקודות תורפה. כמו כן שימוש ב socket מוזיל משמעותית עלויות של קיבולת רוחב פס מכיוון שלא צריך לתרגם עמודים שלמים של מידע web-י לבקשות (request), אלא שולח את המידע הבסיסי (רשומה דקה).

   חסרונות של ה socketים הם ברורים מכיוון שאינם מממשים פרוטוקולי רשת נפוצים אינם מתאימים לתקשורת ברשתות גדולות ומגוונות(בעיית מימוש עם threadים) וכן אינה מטפלת ברציפות תפקודית של המידע שכן יכול להישלח ולהתפרס על פני כמה חבילות (packets) תלוי נתבים ופרוטוקולים (TCP/UDP/IPV6/IPV4) וכן הלאה.

   מקרים בהם נעדיף שימוש ב socketים הם בעיקר ברשתות סגורות וקטנים מספיק עמיד בכדי להחזיק רשת קטנה עם מעט משאבים נוספים נדרשים וכן תקשורת דוגמת WiFi.

2. GCM (Google Cloud Messaging):

   הדפ"א הנבחרת.

   התקשורת הנפוצה היום בקרב מכשירי האנדרואיד והינה חלק מחבילת google play ב Android SDK Manager.

   התקשורת מספקת שירות העברת מסרים ממכשיר למכשיר וכן ממכשיר ולשרת ונהפוכו.

   השירות המסופק מבוקר היטב במקום מרכזי אחד (Google Developers Console)

ולמעט שירותי התקשורת מעניק בטחון בהזדהות של כל אחד מגורמי ה"שיחה" מול השני בזכות מפתחות מחושבים (RegId) ושרת שמנהל בקשות XMPP/ HTTP בשיטת ה doPost המאובטחת כשלעצמה.

המטרה המקורית הייתה לאפשר לאפליקציות (שכן להם יתרון משמעותי מבחינת חווית משתמש על פני אתרים מותאמים) להפסיק לעבוד בתצורה client-server שמאוד מקשה על עדכוני גרסאות ותמיכה במערכות על ידי שליחת מסרים(DATA) למשתמשים רשומים.

השיטה תפסה והפכה גם לדרך מרכזית להעברת מידע בין מכשירי קצה מכיוון שהארכיטקטורה יודעת להתמודד עם ריבוי משתמשים ובקשות וכן לוקחת אחראיות על העברת הודעה (גם במצב טיסה).

תרשים להמחשה: