ספר פרויקט

מערכת LOCKEDAPP

2019

קודש פופיק

דודו ויזל

מנחה: אפי פרופוס

1 תוכן

2 תיאור מוצר

עם התפתחות עולם ה – IOT (Internet Of Things) - קיימים פתרונות ומוצרים הנותנים מענה ואפשרות לשליטה מרחוק על מגוון מכשירים בבית, באמצעות גישה לאינטרנט

לדוגמה: סגירה ופתיחה של תריסים, הפעלת מזגן, דוד חשמלי ועוד

כיום קיימים מגוון מוצרים לפתיחת וסגירת מנעולים מרחוק – מבדיקה שנעשתה פתרונות אילו נותנים מענה לשליטה במספר מנעולים אך אינם עונים על הצורך לשליטה על מספר בתים שונים המכילים כמה מנעולים בתוכם.

LOCKEAPP – הינה מערכת לשליטה על מנעולים מרחוק, המאפשרת נעילה ופתיחה של דלתות מרחוק באמצעות מכשיר הסמרון ו/או באמצעות אתר WEB.
המערכת כוללת אפליקציה (למכשירי אנדרואיד), אתר WEB, ומנעול "חכם" מבוסס בקר Arduino המאפשרים יצירה של משתמשים חדשים והוספת מנעולים ובתים חדשים למערכת.
בעת הרשמה נוצר משתמש אישי עבור כל משתמש ואמצעות החיבור וההזדהות מול המערכת נעשה באמצעות כתובת מייל וסיסמה אישית המוגדרים בעת ההרשמה למערכת.

לאחר התחברות למערכת מוצגים בפני המשתמש כל הבתים אליהם הוא רשום אשר מכילים מנעולים שביכולתו לנעול ולפתוח לפי בקשה, בנוסף המערכת מאפשרת נעילת אדמין – כלומר נעילה מיוחדת שרק משתמש עם הרשאת אדמין יוכל לנעול ו/או לשחרר את המנעול.
על מנת לתמוך באפשרות זו נוצרים משתמשי אדמין עבור על מנעולים אשר ביכולתם לתת גישה ואפשרות למשתמשים חדשים לנעול ולפתוח את אותו מנעול – ובנוסף באפשרותם להעניק הרשאת אדמין לאותו משתמש חדש.
המערכת מאפשרת להוסיף מנעולים חדשים לבתים קיימים או ליצור בית חדש באותה העת

3 מסמך דרישות – SRS

3.1 הרשמה למערכת:

1. הרשמה באמצעות מייל וסיסמה פרטיים

3.2 התחברות למערכת:

1. התחברות דרך האפליקציה ו/או האתר באמצעות המייל והסיסמה המתאימים

3.3 הצגת בתים

1. עבור כל משתמש יוצגו רק הבתים שמכילים מנעול שאותו משתמש רשום אליהם
2. בבחירה וכניסה לתוך בית יוצגו רק המנעולים עבור אותו משתמש
3. לא יוצגו בתים ריקים ללא מנעולים
4. ליד כל בית יופיע שם וכתובת הבית

3.4 הצגת מנעולים

1. יוצגו רק המנעולים בתוך הבית בעלי הרשאות עבור אותו משתמש
2. יצוין המצב הרלוונטי של אותו מנעול (פתוח, סגור או נעילת אדמין)
3. ליד כל מנעול יצוין אם המשתמש בעל הרשאת אדמין
4. ליד כל מנעול יופיע כפתור שיאפשר לשנות את הסטטוס

3.5 שינוי מצב מנעול

1. מנעול במצב OPEN – השתנה למצב CLOSE
2. מנעול במצב CLOSE - השתנה למצב OPEN
3. תינתן אפשרות לנעול (נעילת אדמין) מנעול אחד או כמה
4. רק משתמש בעל הרשאת אדמין יכול לנעול או לשחרר מנעול מנעילת אדמין

3.6 הפעלת מנעול חדש

1. התחברות למנעול תבוצע באמצעות BLUETOOTH באמצעות האפליקציה
2. בעת ההתחברות יש לקבל את ה ID הייחודי של אותו מנעול ורשתות WIFI זמינות
3. תינתן אפשרת להפעיל (Activate) את המנעול ורק לאחר מכן יתאפשר לשלוח בקשות הצטרפות עבור אותו מנעול
4. תינתן אפשרות לשלוח בקשת הצטרפות – במקרה והמנעול מוגדר אך המשתמש לא רשום אליו
5. תינתן אפשרות להגדיר רשת WIFI וסיסמה רק עבור מנעולים שעדיין לא הוגדרו או מנעול מוגדר כאשר המשתמש בעל הרשאת אדמין

3.7 הוספת מנעול

1. הוספת מנעול תתבצע ע"י חיפוש מספר מזהה(ID) ייחודי של כל מנעול
2. במידה והמנעול קיים תתאפשר שליחת בקשת הצטרפות אשר תשלח לכל המשתמשים בעלי הרשאת אדמין על אותו מנעול
3. לא תינתן אפשרות ליצור בית חדש ללא מנעול

3.8 בקשת הצטרפות

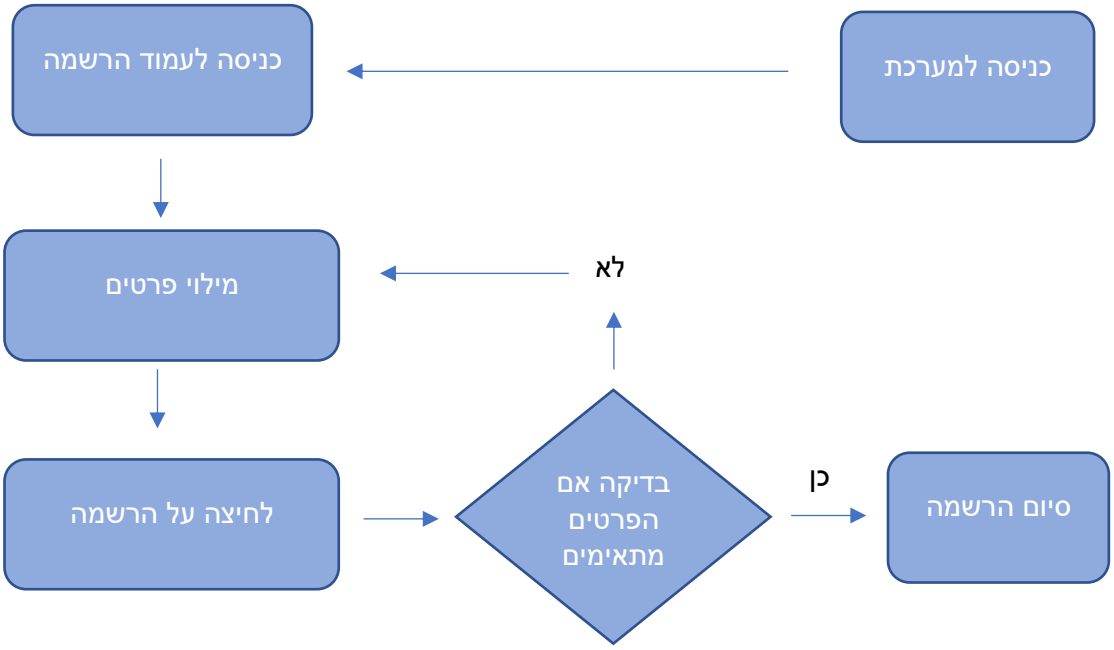
1. כל משתמש בעל הרשאת אדמין יקבל בקשת הצטרפות עבור אותו מנעול
2. כל משתמש יוכל להחליט אם לדחות או לאשר את הבקשה וכמו כן האם לתת הרשאת אדמין עבור אותו משתמש
3. דחיית הבקשה – הבקשה לא תוצג שוב אלא אם כן המשתמש המבקש שלח שוב את אותה בקשה
4. אישור בקשה – בעת שהקשה אושרה יש להסיר את הבקשה עבור שאר המשתמשים בעלי הרשאת אדמין עבור אותו מנעול

3.9 מחיקת מנעול

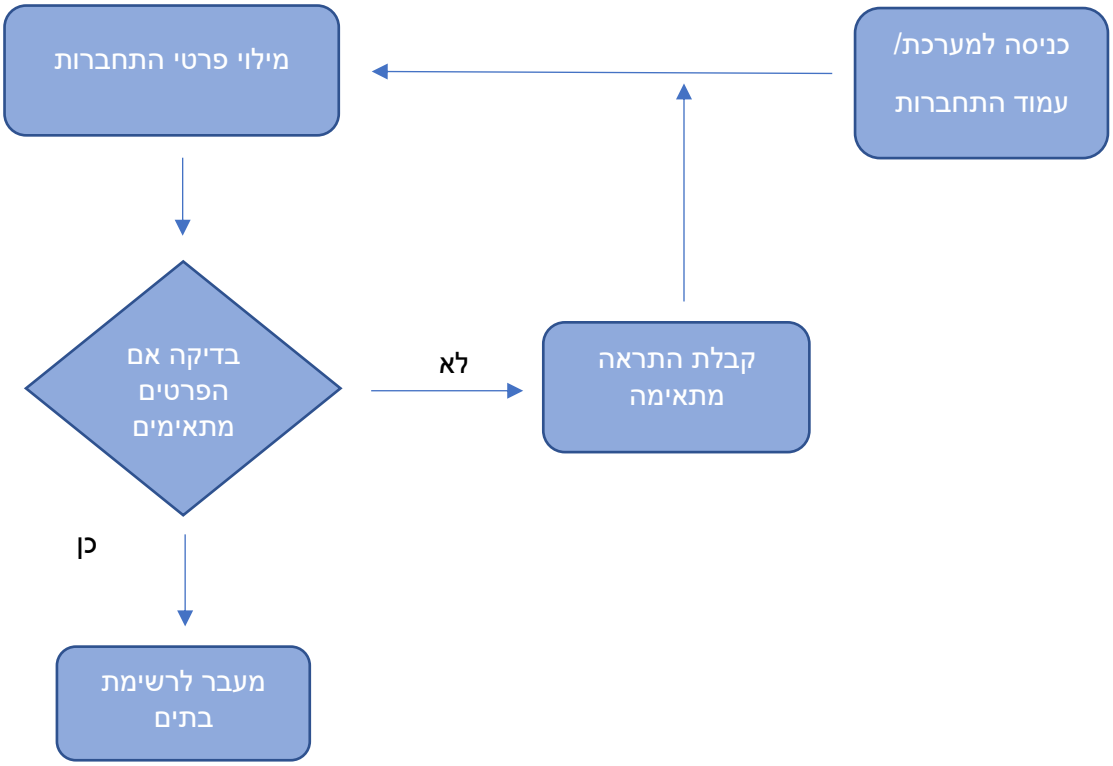
1. בעת המחיקה יש לוודא שקיים לפחות משתמש אחד עם הרשאת אדמין עבור אותו מנעול
2. במידה ואין עוד משתמש אדמין יש לתת הרשאה מתאימה עבור משתמש רגיל שרשום לאותו מנעול
3. במידה ואין עוד משתמשים רשומים תחת אותו מנעול – יש למחוק את המנעול לחלוטין
4. במידה ומוחקים מנעול יש לוודא שהבית שהכיל את המנעול – מכיל עוד מנעולים, במידה והבית ריק יש למחוק אותו
5. אין להשאיר בתים ריקים ללא מנעולים
6. אין אפשרות למחוק כמה מנעולים ביחד

4 תרחישי שימוש – Use Case ותרשימי פעילות Activity Diagrams

ID	UC - 1
שם	**4.1 הרשמה למערכת**
Pre - condition	משתמש חדש שלא רשום במערכת
trigger	כניסה למסך הרשמה מילוי פרטים והשלמת הרשמה
MSS	1. כניסה למערכת 2. לחיצה על כפתור הרשמה 3. הזנת פרטים מלאים ותקינים 4. אורך סיסמה – לפחות 6 תווים 5. NICK – פנוי 6. MAIL – חוקי ופנוי
Exception	1. פרטי משתמש קיימים - המערכת תקפיץ הודעה מתאימה 2. אורך סיסמה פחות מ6 תווים - המערכת תקפיץ הודעה מתאימה

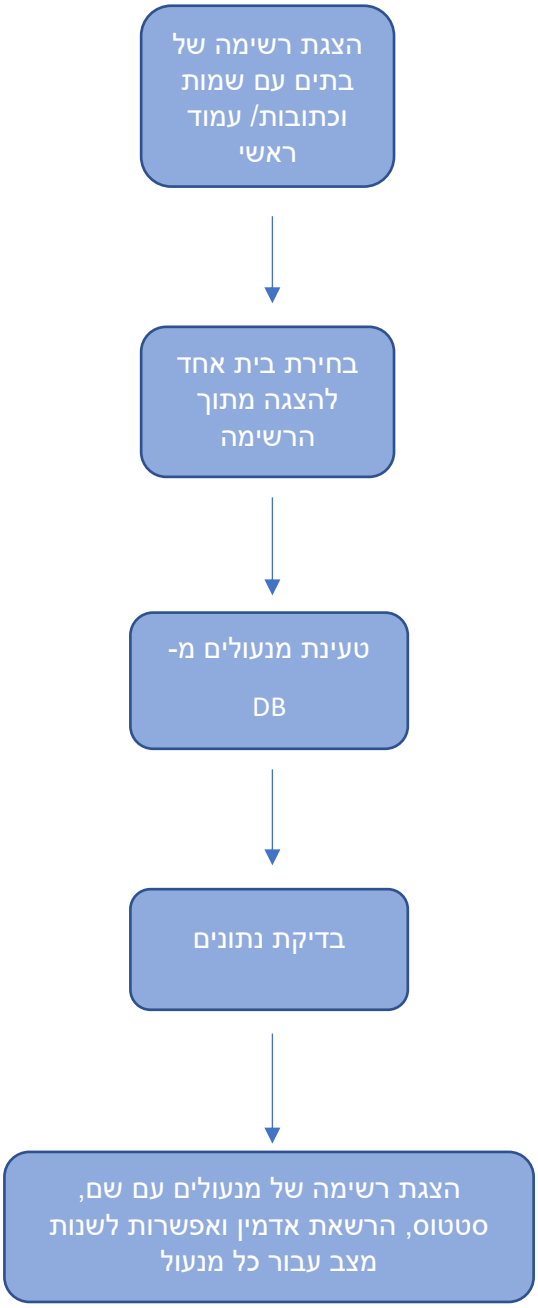


ID	UC - 2
שם	**4.2 התחברות למערכת**
Pre - condition	משתמש קיים
trigger	כניסה למערכת
MSS	1. כניסה למערכת 2. הזנת פרטי משתמש 3. מעבר לעמוד ראשי
Exception	1. פרטי הזדהות לא נכונים – המערכת תקפיץ התראה בהתאם

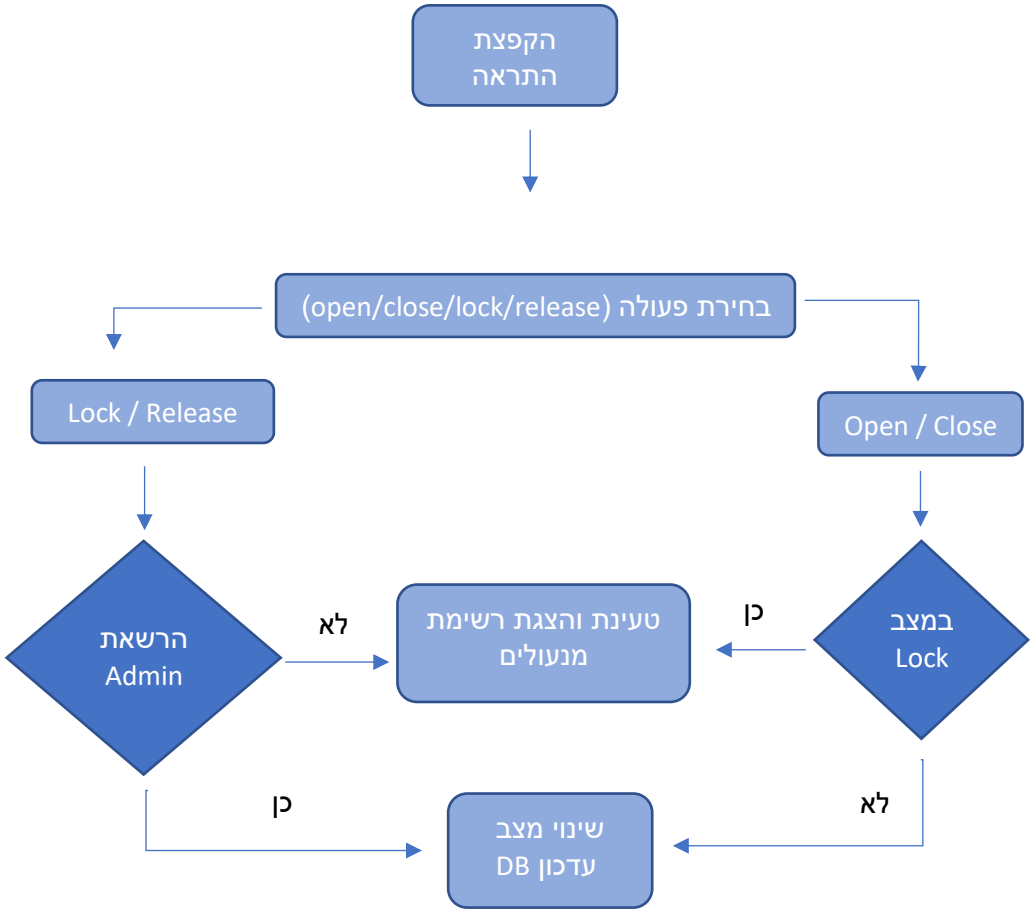


ID	UC - 3
שם	**4.3 הצגת בתים**
Pre - condition	משתמש קיים
trigger	ביצוע התחברות למערכת בהצלחה
MSS	1. התחברות למערכת בהצלחה 2. טעינת רשימת בתים עבור אותו משתמש מ- DB 3. מעבר לתצוגת בתים ראשית 4. הצגת רשימה של בתים עם שם וכתובת
Exception	

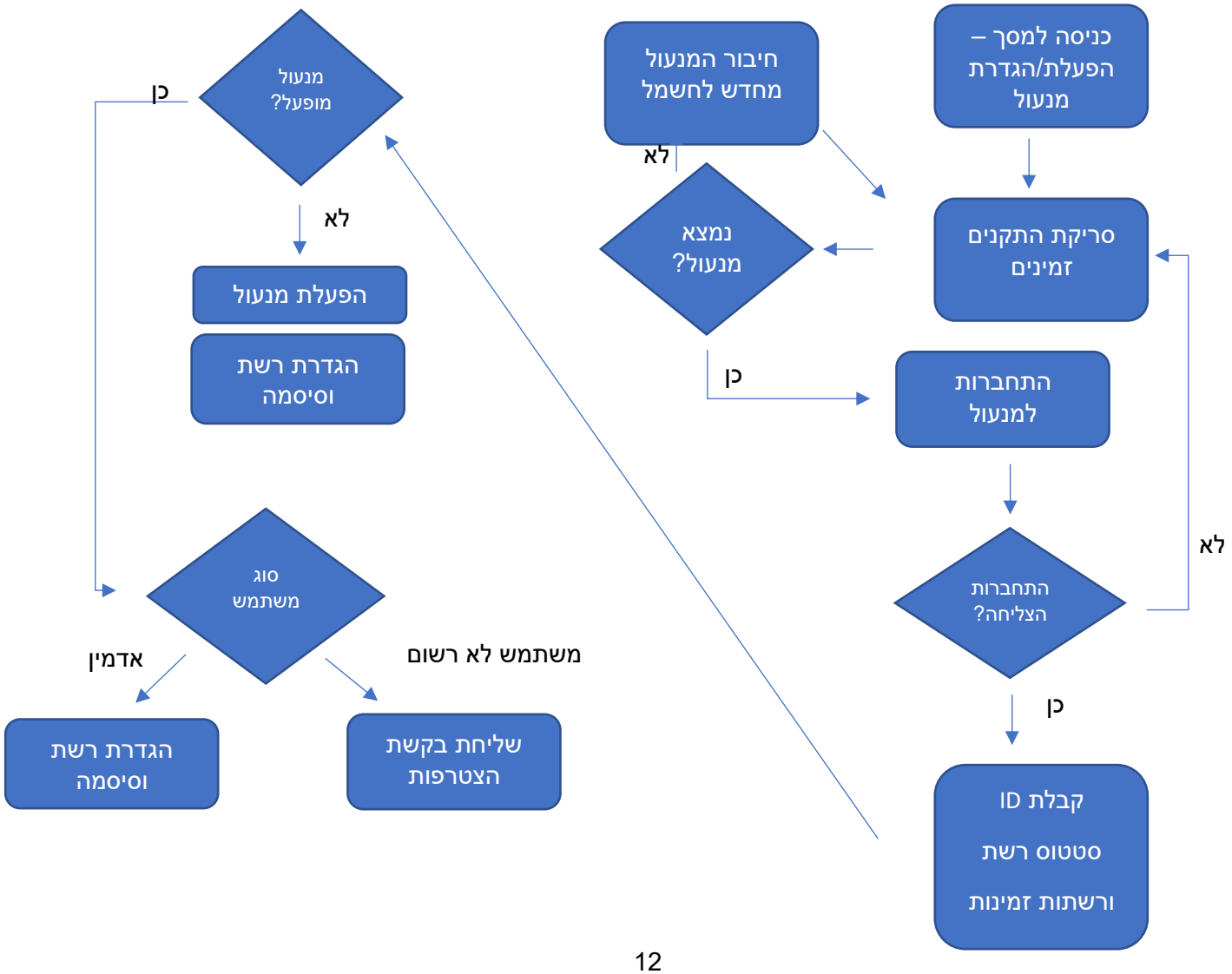
ID	UC - 4
שם	**4.4 הצגת מנעולים**
Pre - condition	התחברות בהצלחה והרשמה לפחות למנעול אחד
trigger	כניסה לרשימת מנעולים עבור בית ספציפי
MSS	1. כניסה לבית מתוך רשימת הבתים 2. טעינת רשימת מנעולים עבור אותו בית מ- DB 3. מעבר לתצוגת מנעולים 4. הצגת רשימה של מנעולים עם סטטוס, שם והרשאת אדמין
Exception	



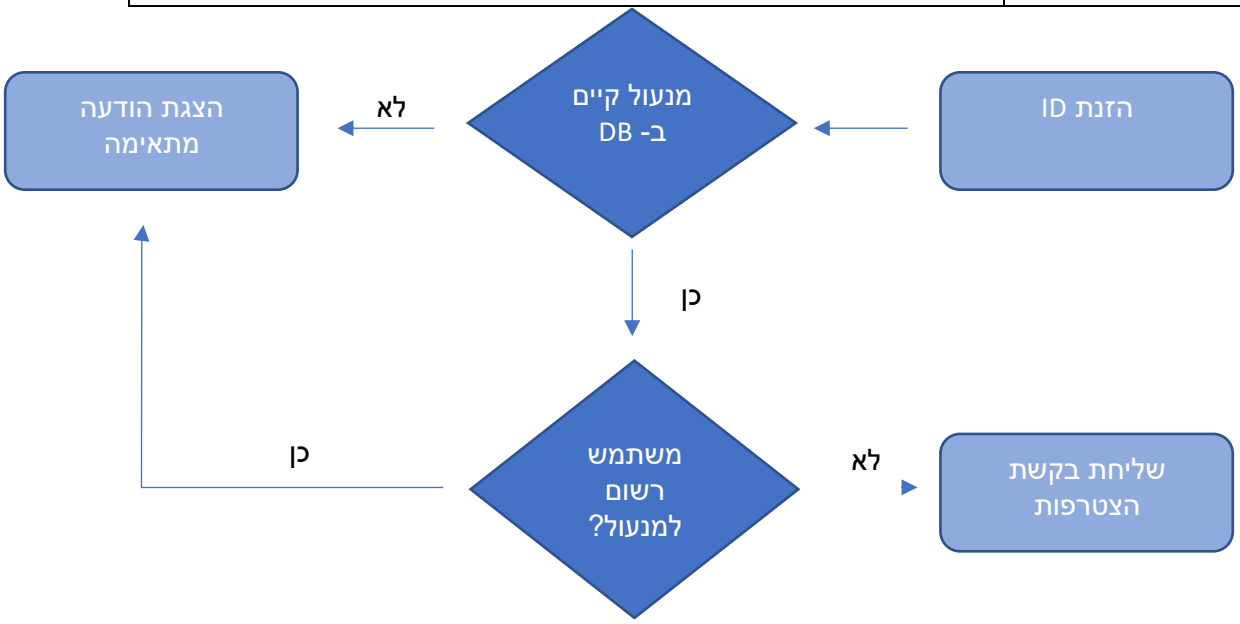
ID	UC - 5
שם	**4.5 שינוי מצב מנעול**
Pre - condition	טעינת רשימת מנעולים בהצלחה
trigger	לחיצה על כפתור שינוי מצב / שינוי מצב של כמה מנעולים
MSS	1. לחיצה על שינו מצב 2. בדיקת מצב נוכחי של המנעול 3. מנעול במצב CLOSE – משתנה ל OPEN 4. מנעול במצב OPEN – משתנה ל CLOSE 5. מנעול במצב LOCK – לא משתנה ללא הרשאת אדמין 6. מעבר למצב LCOK רק על מנעול עם הרשאת אדמין 7. עדכון DB
Exception	1. ניסיון לפתוח מנעול במצב LOCK ללא הרשאת אדמין – המערכת תקפיץ הודעה בהתאם 2. ניסיון להעביר מנעול למצב LOCK ללא הרשאת אדמין – המערכת תקפיץ הודעה בהתאם



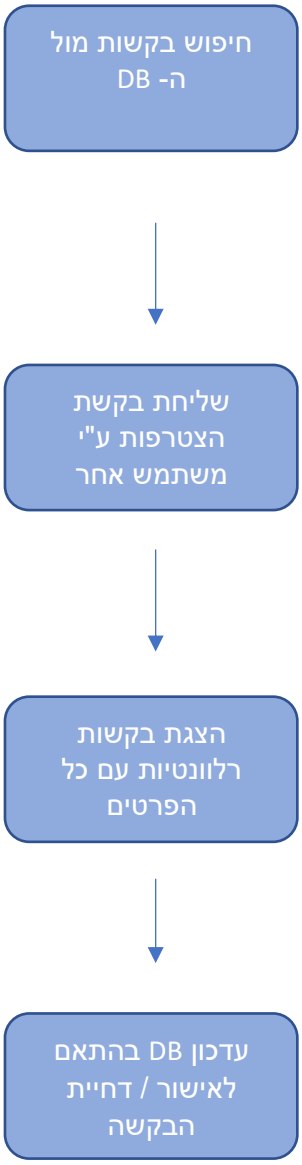
ID	UC - 6
שם	**4.6 הפעלת מנעול חדש**
Pre – condition	חיבור למנעול לחשמל והתחברות בהצלחה
Trigger	כניסה למסך הפעלת/הגדרת מנעול
MSS	1. סריקת התקני BLUETOOTH זמינים 2. התחברות מול המנעול 3. קבלת ID ייחודי של המנעול 4. קבלת רשת מוגדרת וסטטוס חיבור לרשת 5. קבלת רשתות WIFI זמינות 6. במידה והמנעול לא מופעל או שהמשתמש בעל הרשאת אדמין – לאפשר הגדרת רשת וסיסמה חדשים 7. במידה והמנעול לא מופעל – לאפשר הפעלת המנעול 8. במידה והמנעול מופעל אך המשתמש ללא הרשאת אדמין – לאפשר שליחת בקשת הצטרפות
Exception	1. בעת סריקה מנעול לא נמצא – ניתוק וחיבור מחדש של המנעול לחשמל 2. התחברות לוקחת יותר מידי זמן – סריקה והתחברות מחדש 3. הגדרת רשת לוקחת יותר מידי זמן – שליחת רשת וסיסמה מחדש



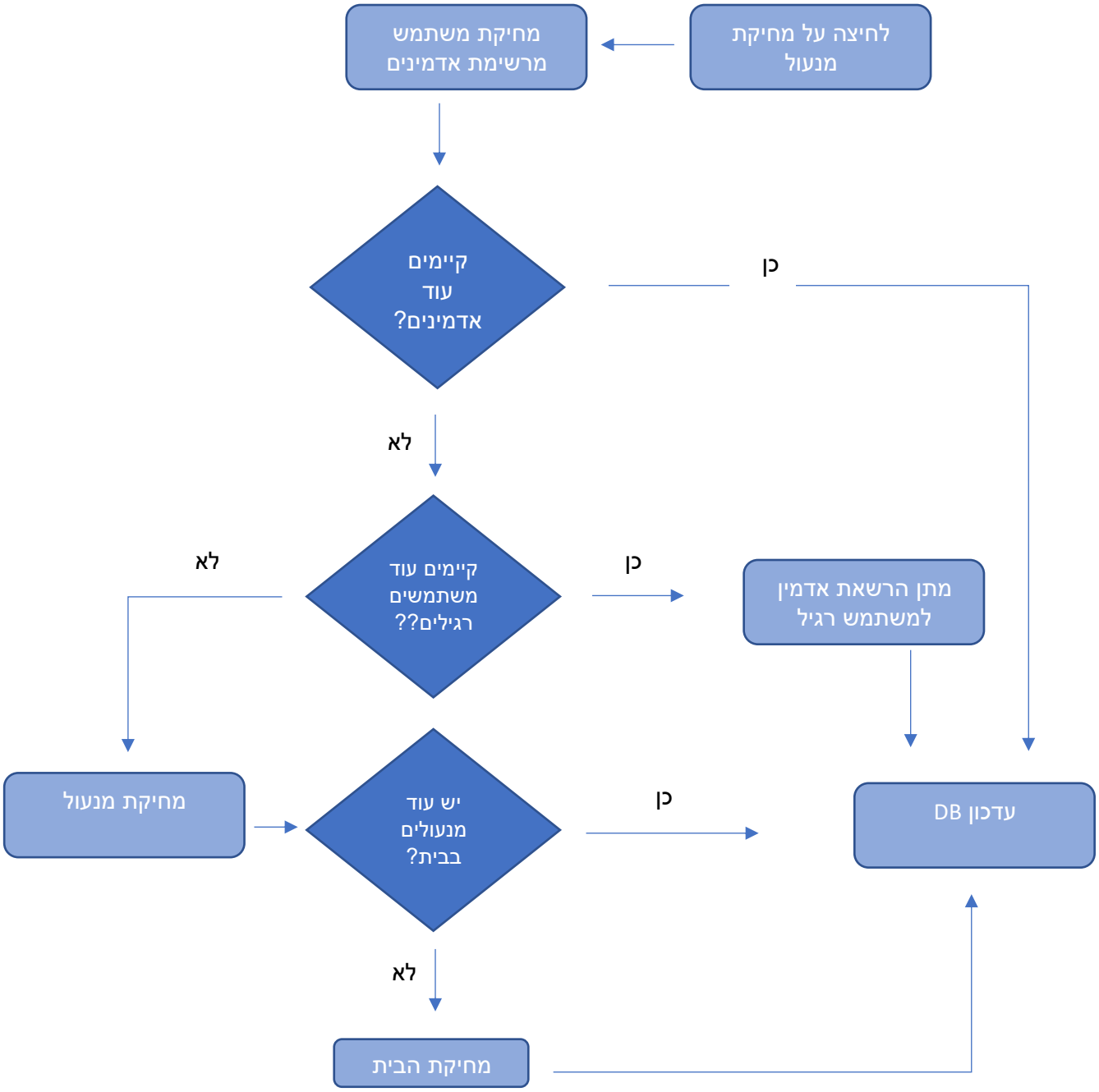
ID	UC - 7
שם	**4.7 הוספת מנעולים**
Pre – condition	מנעול מאותחל ורשום במערכת
Trigger	חיפוש מנעול לפי ID
MSS	9. חיפוש ID ב – DB 10. שליפת המנעול מ – DB 11. שליחת בקשת הצטרפות לכל המשתמשים הרשומים בתור אדמין עבור אותו מנעול
Exception	4. מנעול לא רשום במערכת – המערכת תציג הודעה מתאימה 5. משתמש כבר רשום לאותו מנעול – המערכת תציג הודעה מתאימה



ID	UC - 8
שם	**4.8 בקשת הצטרפות**
Pre - condition	הרשאת אדמין עבור אותו מנעול
trigger	שליחת בקשת הצטרפות ממשתמש אחר
MSS	1. חיפוש בקשות ב DB 2. הצגת בקשה כוללת: שם מנעול, שם משתמש, אישור בתור אדמין, אישור בתור משתמש רגיל, דחיית הבקשה 3. לחיצה על אישור – עדכון DB בהתאם להרשאת אדמין 4. לחיצה על דחיית הבקשה – עדכון הפרטי הבקשה ב DB
Exception	



ID	UC - 9
שם	**4.9 מחיקת מנעול**
Pre - condition	משתמש רשום למנעול שקיים במערכת
trigger	בחירת מנעול ולחיצה על מחיקה
MSS	1. מחיקת משתמש נוכחי מרשימת האדמינים של המנעול 2. אם לא קיימים עוד משתמשים עם הרשאות אדמין - אם יש משתמשים רגילים? יש – מתן הרשאת אדמין לאחד המשתמשים הרגילים אין – מחיקה של המנעול בדיקה אם הבית שבו היה מנעול ריק ממנעולים? ריק – מחיקה של הבית 3. עדכון DB
Exception	



5 מסמך בדיקות - STD

חלק זה נועד לבדיקת התרחישים באמצעות סט של ערכי קלט, תנאים ותוצאה רצויה עבור הבדיקה. תוצאות הבדיקה מתועדות בתרחיש לאורך ביצוע הבדיקה ובחינת התוצאות וההתנהגות.
כל מקרה בדיקה (Test Case) מכיל מכלול של תהליכים שמטרתם להעריך את עמידתה של המערכת בדרישות
שהוצבו. כמו כן להוכיח שהרכיב הנבדק או המערכת מתאימים למטרתם ולצרכי המשתמש ולאתר כשלים. הבדיקות יבוצעו לכל תרחיש שימוש שהוגדר במסמך הדרישות, יתכנו מספר בדיקות לאותו תרחיש שימוש כדי לבחון שהוא עומד בכל המצבים ובכדי להבטיח את מימוש הדרישות.

מערכת LOCKEADPP בנויה משני פלטפורמות משתמש ולכן מקרי הבדיקות ו/או שלבי בדיקות ספציפיים יסומנו בהתאם לפי סוג המערכת הנבדקת: בדיקות WEB יסומנו ירוק, בדיקות אנדרואיד יסומנו בכחול, בדיקות משותפות יסומנו בצהוב

5.1 TC – 1 הרשמה למערכת (UC – 1)

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
הרשמה מוצלחת (MSS)	1. הזנת NICK פנוי 2. הזנת כתובת MAIL חוקית 3. הזנת סיסמה חוקית 4. הזנת סיסמה פעם שנייה תואמת 5. לחיצה על כפתור הרשמה	1. לא מוצגת הערה 2. לא מוצגת הערה 3. לא מוצגת הערה 4. לא מוצגת הערה 5. הרשמה הושלמה מעבר למסך התחברות		
השארת שדה NICK ריק	1. לחיצה על השדה הנבדק 2. לחיצה על שדה אחר 3. לחיצה חזרה על השדה הנבחר	1. השדה הנבחר קיבל פוקוס ומקלדת נפתחה 2. סימן הערה מופיע בצד ימין בשדה הנבחר 3. מופיעה הערה שהשדה לא יכול להישאר ריק		
הזנת כינוי/מייל תפוס	1. לחיצה על השדה הנבדק 2. הזנת כינוי/מייל קיים ולחיצה על שדה אחר 3. לחיצה חזרה על השדה הנבדק	1. השדה הנבדק קיבל פוקוס ונפתחה מקלדת 2. סימן הערה מופיע בצד ימין בשדה הנבדק 3. מופיעה הערה שהכינוי/סיסמה כבר תפוס		
הזנת סיסמה לא חוקית	1. לחיצה על שדה הסיסמה 2. הזנת סיסמה באורך 5 תווים ולחיצה על שדה אחר 3. לחיצה על שדה הסיסמה	1. שדה הסיסמה קיבל פוקוס ונפתחה מקלדת 2. סימן הערה מופיע בשדה הסיסמה 3. מופיעה הערה שהסיסמה קצרה מדי		
הזנת סיסמה כפולה לא תואמת	1. הזנת סיסמה חוקית בשדה הסיסמה ולחיצה על שדה הסיסמה הכפולה 2. הזנת סיסמה לא תואמת או לא באותו אורך	1. לא מופיע הערה בשדה הסיסמה הראשונה 2. במהלך ההקלדה תופיע הערה המציינת שהסיסמה לא תואמת		
לחיצה על כפתור ההרשמה עם נתונים לא טובים	1. לחיצה על כפתור ההרשמה עם פרטים לא טובים/חסרים	1. המערכת מציגה התראה מתאימה		

5.2 TC – 2 התחברות למערכת (UC - 2)

תנאים מקדימים: משתמש רשום בהצלחה

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
התחברות מוצלחת (MSS)	1. הזנת מייל וסיסמה תואמים ולחיצה על התחברות	1. מעבר למסך הראשי		
התחברות מוצלחת באמצעות כינוי	1. הזנת כינוי וסיסמה תואמים ולחיצה על התחברות	1. מעבר למסך הראשי		
התחברות עם מייל לא רשום	1. הזנת מייל לא רשום במערכת, הזנת סיסמה ולחיצה על התחברות	1. קבלת הודעת התחברות נכשלה		
התחברות עם סיסמה לא תואמת	1. הזנת סיסמה רשומה במערכת עם סיסמה לא נכונה ולחיצה על התחברות	1. קבלת הודעת התחברות נכשלה		
התחברות עם כינוי לא רשום	1. הזנת כינוי לא רשום וסיסמה ולחיצה על התחברות	1. קבלת הודעת התחברות נכשלה		
התחברות עם מידע חסר	1. לחיצה על התחברות מבלי למלא סיסמה ו/או מייל או כינוי	1. קבלת הודעה שחסר מידע		

5.3 TC – 3 הצגת בתים – מסך ראשי (UC – 3)

תנאים מקדימים: הרשמה לפחות למנעול אחד וביצוע התחברות מוצלחת

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
הצגת בתים מתאימים בלבד	1. הצגת מסך ראשי – המכיל רשימת בתים בתים מופיעים במצב תצוגה מצומצם	רשימת בתים נגללת למעלה למטה)		
מעבר בין מצבי תצוגה מצומצם / מורחב	1. ברירת מחדל כאשר המסך נטען בפעם הראשונה היא – מצב מצומצם 2. לחיצה על שם הבית ומעבר למצב תצוגה מורחב 3. לחיצה נוספת על שם הבית ומעבר חזרה למצב תצוגה מצומצם 4. וודא שלא מופיעים בתים בלי מנעולים	1. כל רשומה של בית מציגה את שם וכתובת הבית וכמות המנעולים בבית שהמשתמש רשום אליהם 2. רשומת הבית מתרחבת, מספר המנעולים בבית מתחלף לאייקון דלת, רשימת מנעולים נגללת (ימין שמאל) מופיעה 3. רשימת המנעולים נעלמת, אייקון הדלת הופך חזרה למספר המנעולים והרשומה מצטמצמת 4. בכל רשומה של בית חייב להיות לפחות מנעול 1 שהמשתמש רשום אליו אחרת לא יוצג ברשימה – מתעדכן בזמן אמת		
תצוגת כפתורים נוספת	1. וודא הצגת כפתור הפעלת מנעול 2. וודא הצגת כפתור הוספת מנעול 3. וודא הצגת כפתור בקשות הצטרפות	1. כפתור עם אייקון BLUETOOTH מופיעה בצד ימין של חלקו התחתון של המסך 2. כפתור הוספת מנעול חדש מופיע במרכז צידו התחתון של המסך 3. כפתור עם אייקון הודעות מופיע בצד שמאל של חלקו התחתון של המסך עם מספר ההתראות בכל רגע (כאשר אין הודעות לא מוצגת הספרה 0) – מתעדכן בזמן אמת		

5.4 TC – 4 הצגת מנעולים – מסך רשימת מנעולים (UC - 4)

תנאים מקדימים: הרשמה לפחות למנעול אחד וביצוע התחברות מוצלחת

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
הצגת מנעולים – מסך ראשי	1. מעבר למצב תצוגת בית מורחב	1. רשימת מנעולים נגללת (ימין - שמאל)באותו בית, שהמשתמש רשום אליהם מופיעה עם אייקון מתאים עבור הסטטוס העדכני של המנעול – מתעדכן בזמן אמת		
הצגת מנעולים במסך רשימת מנעולים	1. במצב תצוגה מורחב של בית לחיצה על אייקון הדלת 2. וודא שם וכתובת הבית מוצגים 3. וודא רשימת מנעולים נגללת (למעלה – למטה) מופיעה	1. מעבר למסך רשימת מנעולים 2. שם וכתובת הבית מופיעים בחלקו העליון של המסך 3. רשימת מנעולים המכילה רק את המנעולים שהמשתמש רשום אליהם באותו בית מופיעה, כאשר כל רשומה מכילה את שם המנעול, אייקון סטטוס עדכני ותיבת סימון המסמנת אם למשתמש יש הרשאת אדמין לאותו מנעול – מתעדכן בזמן אמת		
הצגת כפתורים נוספים	1. וודא סרגל כפתורים לשינוי מצב מנעול 2. וודא כפתורים להוספת ומחיקת מנעול3 3. לחץ על שם המנעול, וודא הודעת כמות מנעולים שנבחרו 4. וודא שלא נבחר אף מנעול	1. ברגל כפתורים עם אייקונים של מצב פתוח, סגור ונעול מופיע מתחת לרשימת המנעולים 2. כפתורים להוספת ומחיקת מנעול מופיעים מתחת לסרגל הכפתורים 3. הודעה המציגה את כמות המנעולים שנבחרו וכמות המנעולים הכללית ברשימה מופיעה מעל רשימת המנעולים ושם המנעול הנבחר מופיע בפונט גדול ובולט יותר 4. כל שמות המנעולים מופיעים בפונט רגיל והודעה מעל רשימת המנעולים לא מופיעה		

5.5 TC – 5 שינוי מצב מנעול (UC – 5)

תנאים מקדימים: הרשמה לפחות למנעול אחד עם הרשאת אדמין

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
שינוי מצב במסך ראשי – מצב תצוגת בית מורחב	1. במצב תצוגת בית מורחב – לחץ על אייקון מצב המנעול	1. מנעול במצב פתוח – משתנה למצב סגור מנעול במצב סגור – משתנה לפתוח מנעול במצב נעול – מתקבלת הודעה שצריך לשחרר את המנעול קודם כל המצבים מתעדכנים בזמן אמת		
שינוי מצב מנעול בודד במסך רשימת מנעולים	1. במצב תצוגת בית מורחב – לחיצה על אייקון הדלת 2. לחיצה על אייקון המצב של מנעול אחד	1. מסך רשימת הבתים מוצג כנדרש 2. אותה תוצאה כמו בשינוי מצב תצוגת בית מורחב		

שינוי מצב של כמה מנעולים ביחד – באמצעות סרגל המצבים	1. בחירה של מנעול אחד או יותר ע"י לחיצה על שם המנעול 2. לחיצה על אייקון פתיחה 3. לחיצה על אייקון סגירה 4. לחיצה על אייקון נעילת אדמין	1. שם המנעול הנבחר מופיע בפונט בולט וגדול והודעה המייצגת כמה מנעולים נבחרו מתוך הרשימה מופיעה מעל רשימת המנעולים 2. מנעול במצב סגור – משתנה לפתוח מנעול במצב סגור – נשאר סגור מנעול במצב נעול עם הרשאת אדמין - משתנה לפתוח מנעול במצב נעול ללא הרשאת אדמין – המערכת מודיעה שחסר הרשאת אדמין והמצב לא משתנה 3. מנעול במצב סגור – לא משתנה מנעול במצב סגור – משתנה לפתוח מנעול במצב נעול עם הרשאת אדמין - משתנה לסגור 4. מנעול במצב נעול ללא הרשאת אדמין – המערכת מודיעה שחסר הרשאת אדמין והמצב לא משתנה מנעול עם הרשאת אדמין במצב פתוח / סגור - משתנה לנעול מנעול ללא הרשאת אדמין – לא משתנה והמערכת מודיעה על חוסר בהרשאת אדמין		

5.6 TC – 6 הפעלת מנעול חדש (UC – 6)

תנאים מקדימים: הפעלה ראשונית של מנעול לא מוגדר והתחברות בהצלחה

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
משיכת מידע מהמנעול באמצעות BLUETOOTH	1. לחיצה על כפתור ה BLUETOOTH במסך הראשי 2. לחיצה על כפתור הסריקה 3. לחיצה על כפתור ההתחברות ברשימת המנעול 4. וודא משיכת מידע מהמנעול 5. וודא הצגת המידע המתקבל 6. וודא לאחר קבלת המידע הצגת כפתורים	1. במידת הצורך בקשה להפעלת BLUETOOTH והצגת מסך הפעלת מנעול חדש עם כפתור סריקה 2. סטטוס מציג: סריקה ורשימת התקנים זמינים (נגללת למעלה – למטה) מוצגת על המסך, במידה ולא מופיע המנעול – לחבר מחדש את המנעול לחשמל וסרוק שוב 3. סטטוס מציג: מתחבר במידה ולוקח יותר מ20 שניות יש להריץ את הבדיקה מההתחלה 4. סטטוס מציג הודעה מתאימה בהתאם למידע המתבקש, במידה ולא מגיע לסטטוס:		

		"התחבר" יש לבצע את הבדיקה מחדש 5. מוצג על המסך ID המנעול, תיבת סימון שמציגה אם המנעול מופעל או לא, רשת הWIFI המוגדרת למנעול ותיבת סימון המסמנת אם המנעול מחובר לרשת או לא 6. הצגת כפתורים SET WIFI ו- ACTIVATE LOCK		
הגדרת WIFI	1. לחץ על כפתור SET WIFI 2. בחר רשת WIFI מתוך הרשימה 3. הזן סיסמה 4. לחץ שלח/OK 5. וודא הצלחת תהליך 6. התחברות מחדש למכשיר	1. הצגת חלון עם רשימת רשתות נגללת 2. בחר רשת מתוך הרשימה הנגללת 3. ניתן להשאיר ריק במידה ואין צורך בסיסמה 4. סטטוס מציין: מגדיר WIFI 5. המערכת מודיעה על הצלחה ומציאה להתחבר מחדש למנעול ע"מ לוודא את התהליך, במידה והתהליך לא מסתיים יש לבצע את הבדיקה מחדש 6. בצע את בדיקת משיכת מידע מהמנעול שוב וודא שהרשת שבחרת מוגדרת למנעול		
הפעלת מנעול חדש	1. לחץ על כפתור הפעל מנעול 2. הגדר שם מנעול 3. בחר בית חדש/ בית קיים מתוך הרשימה 4. לחץ על שלח/OK 5. חזרה למסך ראשי 6. כניסה למסך רשימת מנעולים - וודא הרשאת אדמין	1. מסך הוספת מנעול חדש מופיע 2. הזן שם מנעול 3. במידה ונבחר בית חדש יוצגו שדות להזנת שם וכתובת הבית בחדש 4. כפתור המנעול נעלם 5. אם נבחר להוסיף בית חדש – הבית מופיע ברשימה עם המנעול אם נבחר להוסיף לבית קיים – המנעול התווסף לרשימת המנעולים באותו בית 6. במסך רשימת מנעולים של אותו בית וודא שתיבת הסימון להרשאת אדמין עבור אותו מנעול מסומנת		
הצגת כפתורים	ק משיכת מידע מהמנעול - וודא הופעת התאם	1. כפתורי SET WIFI ו- ADD LOCK / ACTIVATE בהתאם לתרשים ב UC-6		

5.7 TC – 7 הוספת מנעול (UC-7)

תנאים מקדימים: לחיצה על הוספת מנעול חדש מהמסך הראשי או מתצוגת בית מורחבת וחיפש ID של

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
הוספת מנעול מופעל באמצעות מסך חיפוש ID	1. הצגת מסך חיפוש מנעול 2. הזנת ID של מנעול מופעל ולחיצה על חיפוש 3. לחיצה על שלח/OK	1. וודא נראות לפי הצילום מסך 2. הצגת הודעה לשליחת בקשת הצטרפות 3. חזרה למסך ראשי / תצוגת מנעולים וקבלת הודעה שבקשת הצטרפות נשלחה למשתמשים המתאימים		
חיפש ID של מנעול לא מופעל	1. הזנת ID של מנעול שלא מופעל ולחיצה על חיפוש	1. חזרה למסך ראשי/ הצגת מנעולים וקבלת הודעה על מנעול שלא מופעל		
חיפוש ID של מנעול שכבר רשומים אליו	1. הזנת ID של מנעול שכבר רשומים אליו	1. חזרה למסך ראשי / הצגת מנעולים וקבלת הודעה מתאימה		
הוספת מנעול מופעל באמצעות מסך הפעלת מנעול חדש	1. במסך ראשי לחץ על כפתור ה BLUETOOTH ובצע בדיקת משיכת מידע מהמנעול (TC-6) 2. לחיצה על ADD LOCK 3. לחיצה על שלח/OK	1. הצגת כפתור ADD LOCK 2. הצגת הודעה לשליחת בקשת הצטרפות 3. חזרה למסך הוספת מנעול וקבלת הודעה על שליחת בקשת הצטרפות		

5.8 TC – 8 בקשת הצטרפות (UC – 7)

תנאים מקדימים: שליחת בקשת הצטרפות ע"י משתמש אחר

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
ניסיון להצגת בקשות כשאין בקשות זמינות	1. במסך ראשי – לחץ על כפתור ההודעות כשלא מופיע עליו אף מספר	1. קבלת הודעה מתאימה		
הצגת בקשות זמינות	1. במסך ראשי – לחץ על כפתור ההודעות כשמצוין מספר הודעות זמינות 2. וודא הצגת ומספר הודעות 3. לחץ על תיבת הסימון כמה פעמים 4. לחץ על אשר/דחה 5. וודא מול ה DB הוספת מנעול והרשאה בהתאם לפעולה הקודמת	1. מעבר למסך בקשות הצטרפות 2. רשימת בקשות נגללת (למעלה – למטה) מופיעה עם מספר בקשות זהה למספר האינדיקציה במסך הראשי, כל רשומת הודעה כוללת את שם המנעול, שם המשתמש המבקש, תיבת סימון להרשאת אדמין וכפתור אישור ודחייה 3. סימון התיבה משתנה בהתאם לכמות הלחיצות 4. הבקשה נמחקת מהרשימה – במידה ואין עוד בקשות חזרה למסך הראשי 5. אושרה הבקשה – עם/בלי הרשאת אדמין נדחתה הבקשה – לא בוצעה שום פעולה		

5.9 TC – 9 מחיקת מנעול (UC – 9)

תנאים מקדימים: הרשמה למנעול אחד לפחות וכניסה למסך רשימת מנעולים ושני מנעולים לפחות לבדיקה השנייה

תיאור הבדיקה	שלבי הבדיקה	תוצאה רצויה	תוצאה	הערות
מחיקת מנעול	1. לחיצה על שם המנעול מתוך הרשימה 2. לחיצה על כפתור מחיקת המנעול 3. לחיצה על ביטול 4. לחיצת על שם המנעול, לחיצה על מחיקת המנעול ולחיצה על אישור	1. שם המנעול מופיע בפונט גדול ובולט 2. הצגת הודעה לאישור מחיקת המנעול עם שם המנעול 3. חזרה למסך רשימת המנעולים ללא שינוי 4. מחיקת המנעול וחזרה למסך הראשי		
מחיקת כמה מנעולים	1. בחירת כמה מנעולים מתוך רשימת המנעולים 2. לחיצה על כפתור המחיקה	1. לפחות 2 מנעולים נבחרו, פונט המנעולים בולט וגדול יותר וההודעה בהתאם 2. קבלת הודאה מתאימה שלא ניתן למחוק כמה מנעולים ביחד וחזרה למסך רשימת המנעולים		

6 מנעול

ממשק העבודה מול מתבסס על תקשורת BLUETOOTH מול מכשיר האנדרואיד, ותקשורת WIFI על מנת להתעדכן בזמן אמת על פעולת פתיחה, סגירה או נעילה של המנעול – דרך האפליקציה או ממשק ה WEB.

מכונת מצבים

בעת חיבור המנעול לחשמל, המנעול קורא מזיכרון ROM את ה- ID הייחודי, ואת הרשת WIFI והסיסמה שנתנה לו – בעת הפעלה ראשונית הרשת והסיסמה אינם מוגדרים (ריקים)
המנעול מתוכנת לעבוד בצורה של מכונת מצבים - אשר מציינת את פעולת ומצב המנעול בכל רגע.
להלן הסבר המצבים למנעול:

6.1 חסר רשת וסיסמה

מצב התחלתי כאשר לא מוגדר רשת וסיסמה להתחבר – אינדיקציה באמצעות מנורה אדומה מהבהבת

6.2 רשת וסיסמה מוגדרים

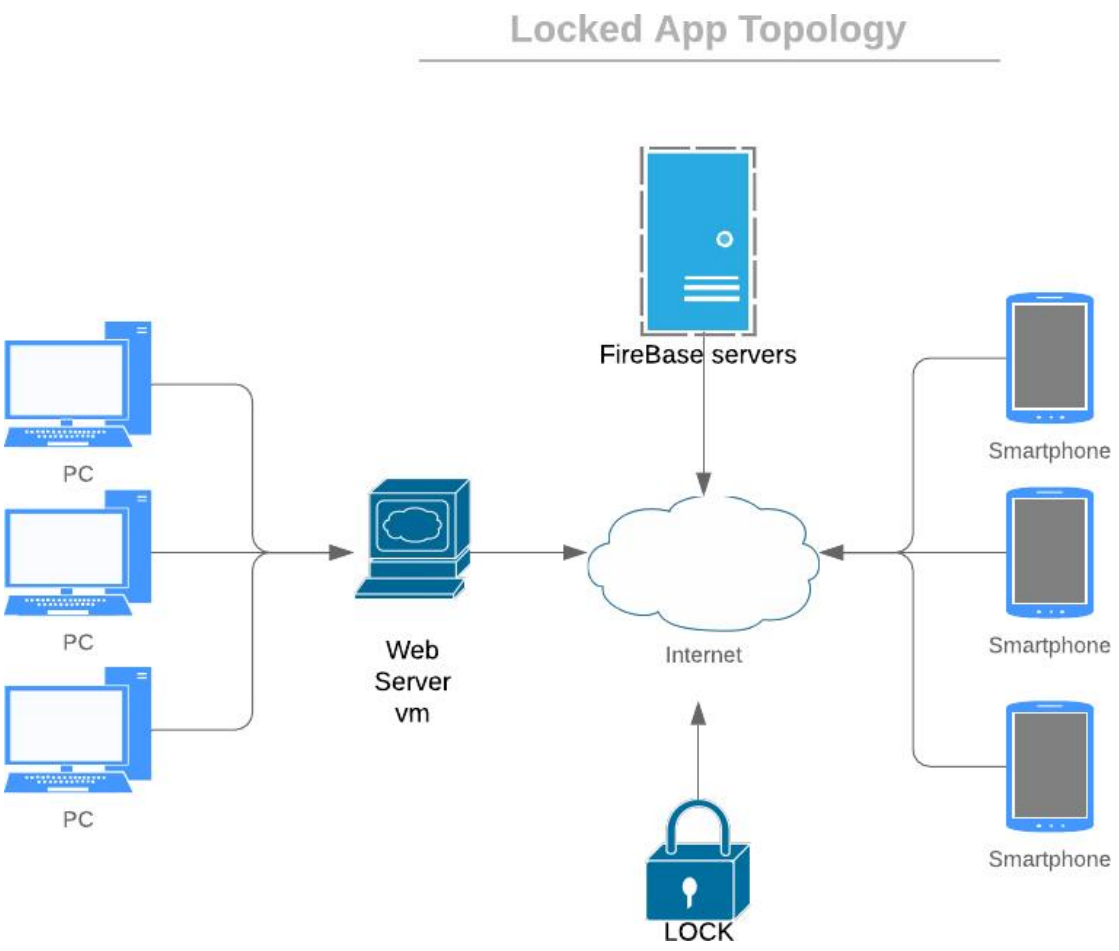
במצב זה המכשיר קרא מזיכרון ה ROM רשת וסיסמה ומנסה להתחבר איתם – אינדיקציה באמצעות ריקוד אורות, 2 צבעים דלוקים בכל פעם

6.3 מחובר לרשת

במצב זה בוצעה התחברות לרשת והמנעול מתעדכן בזמן אמת על המצב המוגדר לו מול ה- DB
אינדיקציה – סגור, מנורה אדומה דולקת קבוע
פתוח, מנורה ירוקה דולקת קבוע
נעול, מנורה כחולה דולקת קבוע
מנעול חדש (לא מוגדר ב- DB), כל שלושת המנורות מהבהבים ביחד
תקשורת BLUETOOTH, מנורה כחולה מהבהבת פעמיים

מעבר בין המצבים מתבצע לבד ללא כל התערבות מצד המשתמש, למעט הגדרת רשת וסיסמה להתחברות

7 תרשים פריסה



8 בסיס נתונים

בסיס נתונים מבוסס טכנולוגיית NoSQL –

NoSQL הוא קטגוריה חדשה יחסית של בסיסי נתונים ,אשר נותנים פתרון אחסון וגישה למידע שאינו ממודל במבנה טבלאי יחסי אשר נפוץ בבסיסי נתונים יחסיים.

המוטיבציה בגישה לפיתוח בסיס נתונים כזה, כוללת פשטות של אפיון ,סילומיות רוחבית ובקרה מוגברת על זמינות.

מבנה המידע שונה ממערכות בסיסי נתונים יחסיים, ולכן ישנן פעולות שמהירות יותר ב NoSQL- וישנן פעולות שמהירות יותר בבסיס נתונים יחסי. בסיסי נתונים מסוג NoSQL הופכים נפוצים יותר במערכות Big Data וכן במערכות זמן אמת.

מערכות אלו נקראות לעיתים ,"Not Only SQL" כדי להדגיש שלחלקן תמיכה בשפת השאילתות.SQL

8.1 מבט על:

8.2 רשומת HOUSE:

8.3 רשומת LOCKS:

```
locks
  22
  30
    admin: "vgx390ImoXbVQyaV2VIAMOHeZ9(
    id: "30"
    name: "30"
    notAdmin: "8lL4Xd8nx0Nmp1kdbJo1FhRqYe
    status: "close
  33
  95
  96
  97
  98
  99
```

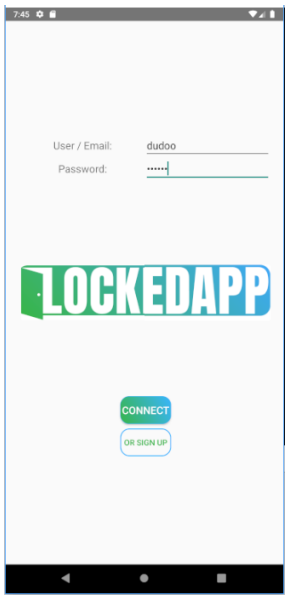
8.4 רשומת REQ:

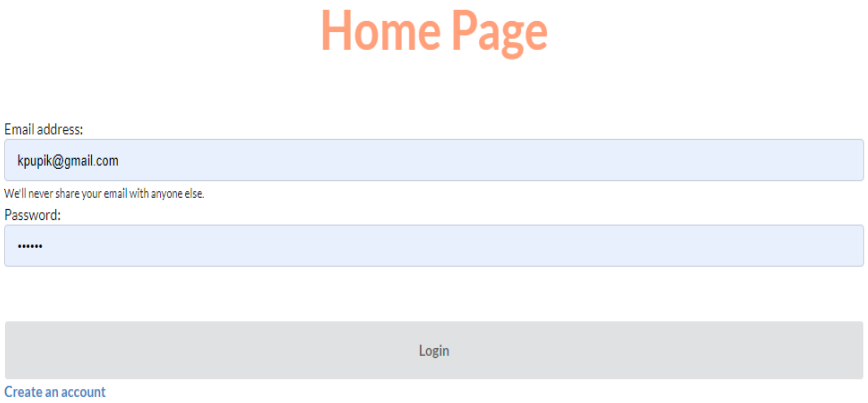
```
req
  -LriHdCoMmWPGlYUxNNU
    fromUser: "stBYnfpRWQZLmUF1ffktpF8MImr
    houseId: "-LrKBV8pAmPB324DhHO
    lockId: "95"
    toUsers: "vgx390ImoXbVQyaV2VIAMOHeZ9(
```

8.5 רשומת USERS:

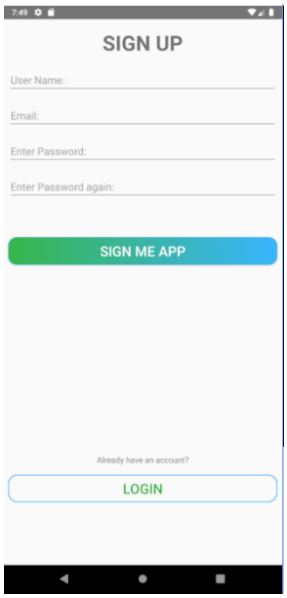
```
users
  8lL4Xd8nx0Nmp1kdbJo1FhRqYey2
  F2S6dpfwidYRM0KeGaR6hlpaH7o2
  MyKqjknhOyS4LqFvS9C2ey0ox8F2
  stBYnfpRWQZLmUF1ffktpF8MImn1
  vgx390ImoXbVQyaV2VIAMOHeZ9G2
    email: "wizelt@gmail.cor
    houseList: "-LpxN2_PipHtKOehIbGK,-LpxQRDe0V41eT6ls8GD,-Lq0
    lockList: "22,30,33,98,99,9
    nick: "dudoo
```

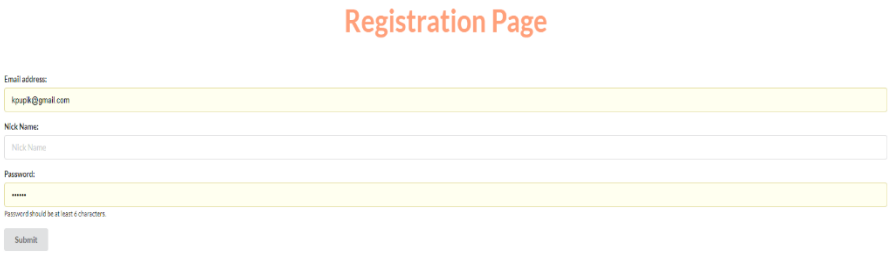
מסך התחברות:



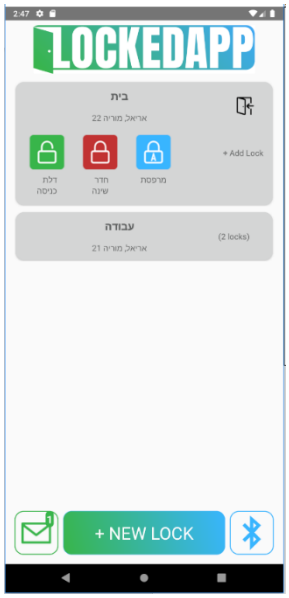


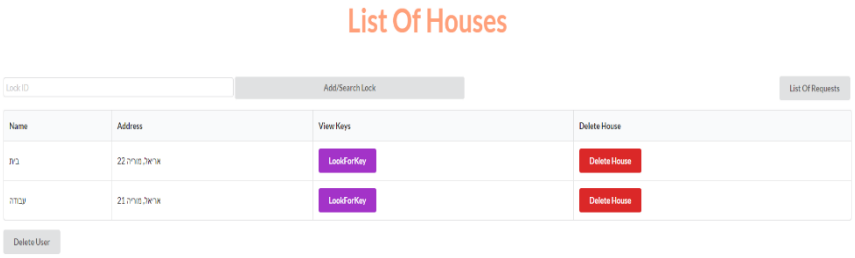
מסך הרשמה:



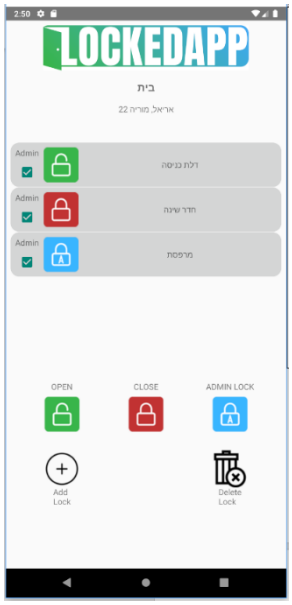


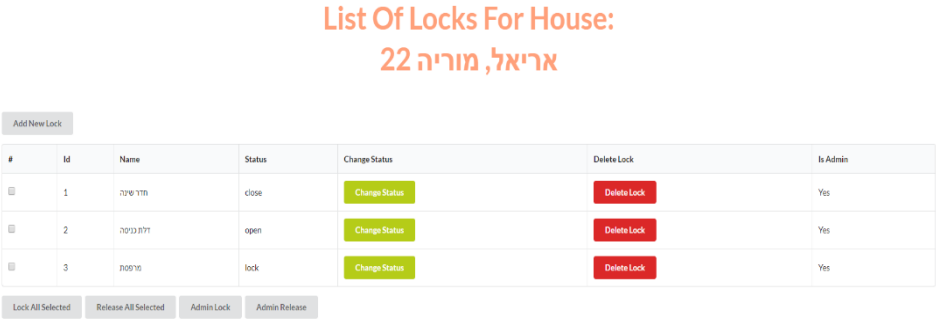
מסך ראשי



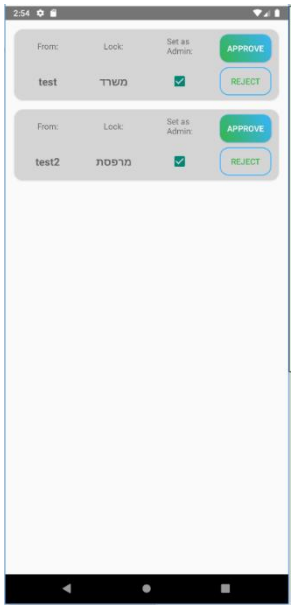


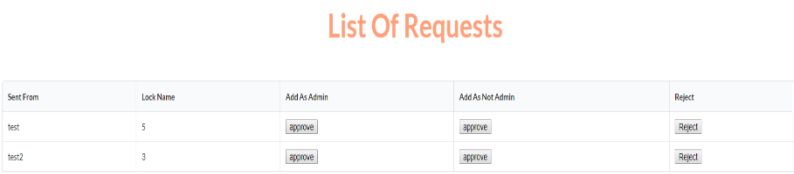
רשימת מנעולים



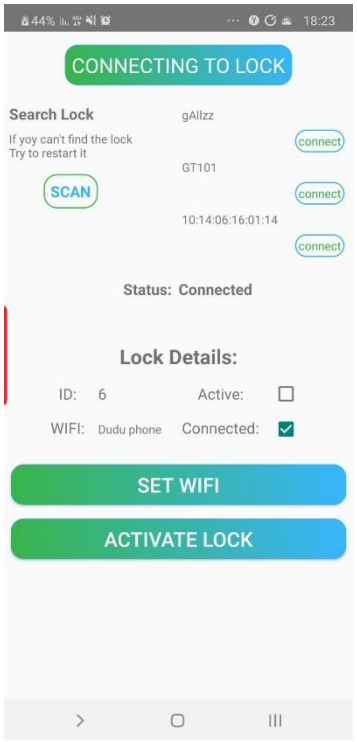


רשימת בקשות הצטרפות

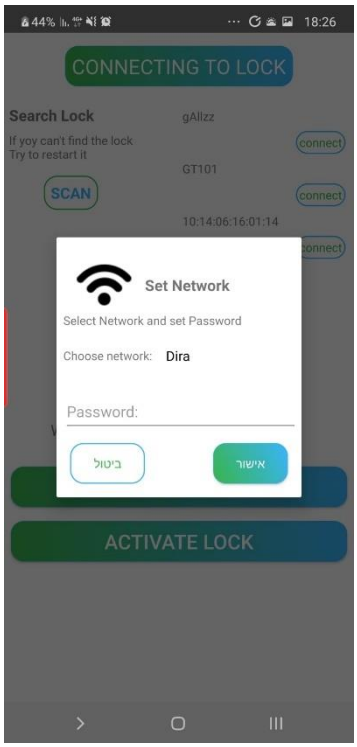




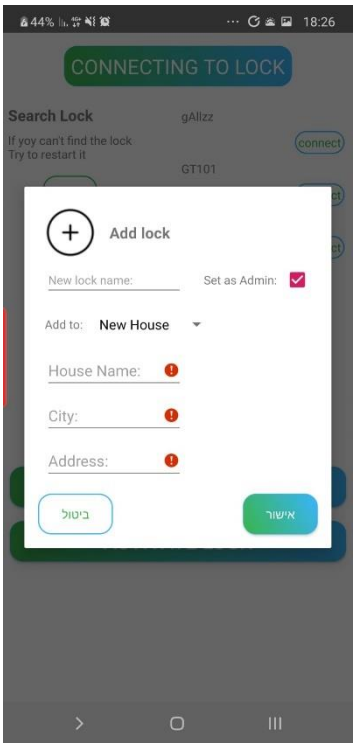
הפעלת/הגדרת מנעול



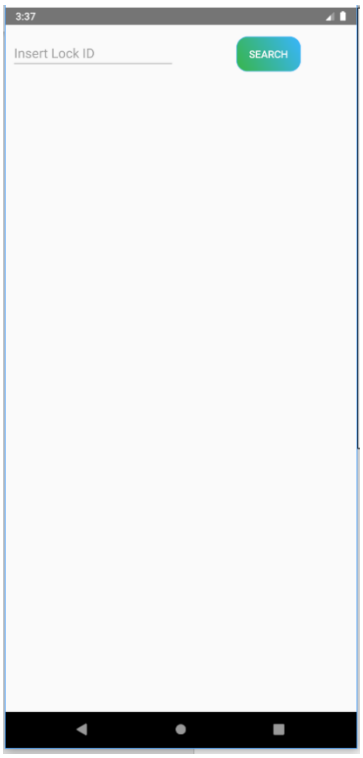
קביעת רשת WIFI



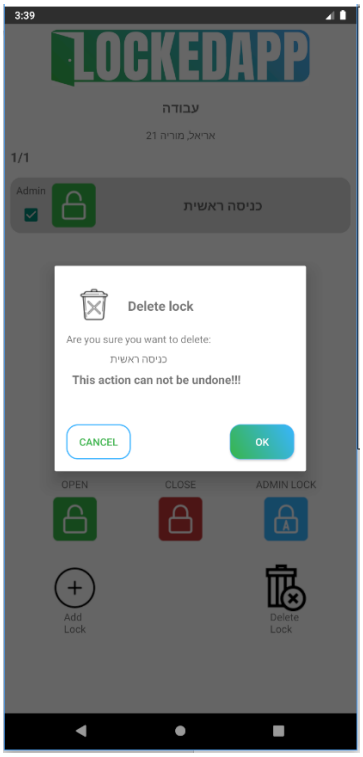
הוספת מנעול חדש



חיפוש מנעול

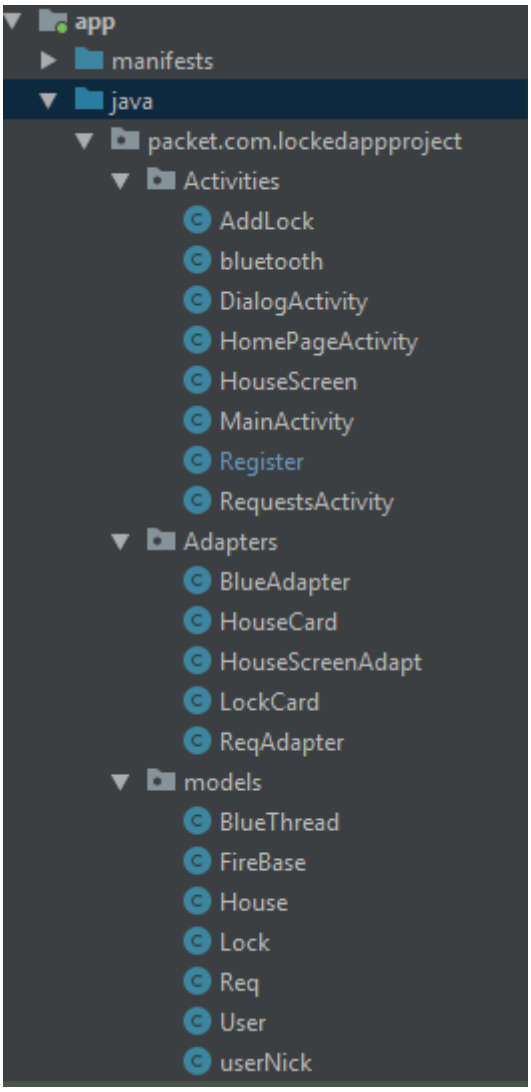


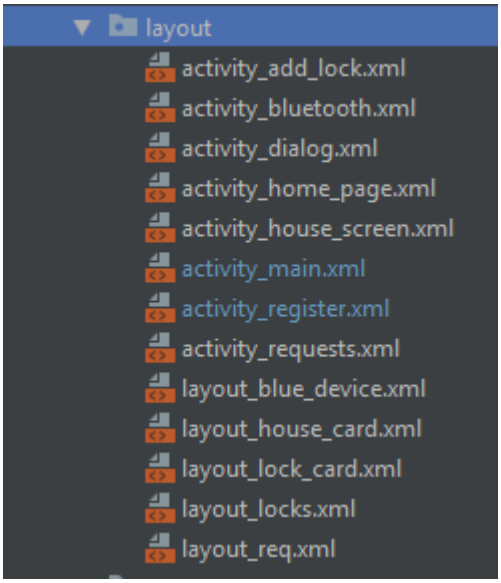
מחיקת מנעול



10 קוד – פלטפורמת ANDROID

10.1 עץ תיקיות





10.2 קטעי קוד נבחרים:

10.2.1 מימוש זמן אמת (real time)

```
private static ChildEventListener lockListener = new ChildEventListener() {
    @Override
    public void onChildAdded(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        Lock lock = dataSnapshot.getValue(Lock.class);
        if (user.lockList.contains(lock.id)) {
            userLocks.add(lock);
            Collections.sort(userLocks);
            for (UpdateLockData u : updateLocks) {
                u.Notify();
            }
        }
    }

    @Override
    public void onChildChanged(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        Lock lock = dataSnapshot.getValue(Lock.class);
        if (lock.admin.contains(mAuth.getUid()) || lock.notAdmin.contains(mAuth.getUid())) {
            int i;
            for (i = 0; i < userLocks.size() && !userLocks.get(i).id.equalsIgnoreCase(lock.id); i++)
                ;
            if (i < userLocks.size()) {
                userLocks.set(i, lock);
            }
        } else {
            int i;
            for (i = 0; i < userLocks.size() && !userLocks.get(i).id.equalsIgnoreCase(lock.id); i++)
                ;
            if (i < userLocks.size())
                userLocks.remove(i);
        }
        Collections.sort(userLocks);
        for (UpdateLockData u : updateLocks) {
            u.Notify();
        }
    }

    @Override
    public void onChildRemoved(@NonNull DataSnapshot dataSnapshot) {
        Lock lock = dataSnapshot.getValue(Lock.class);
        int i;
        for (i = 0; i < userLocks.size() && !userLocks.get(i).id.equalsIgnoreCase(lock.id); i++)
            ;
        if (i < userLocks.size()) {
            userLocks.remove(i);
            for (UpdateLockData u : updateLocks) {
                u.Notify();
            }
        }
    }
```

```java
private static ChildEventListener houseListener = new ChildEventListener() {
    @Override
    public void onChildAdded(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        House house = dataSnapshot.getValue(House.class);
        if (user.houseList.contains(house.id) && !house.id.equalsIgnoreCase( anotherString: "")) {
            userHouses.add(house);
            Collections.sort(userHouses);
            for (UpdateHouseData u : updateHouse) {
                u.Notify();
            }
        }
    }

    @Override
    public void onChildChanged(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        House house = dataSnapshot.getValue(House.class);
        if (house.admin.contains(mAuth.getUid())) {
            int i;
            for (i = 0; i < userHouses.size() && !userHouses.get(i).id.equalsIgnoreCase(house.id); i++)
                ;
            if (i < userHouses.size()) {
                userHouses.set(i, house);
            }
        } else {
            int i;
            for (i = 0; i < userHouses.size() && !userHouses.get(i).id.equalsIgnoreCase(house.id); i++)
                ;
            if (i < userHouses.size())
                userHouses.remove(i);
        }
        Collections.sort(userHouses);
        for (UpdateHouseData u : updateHouse) {
            u.Notify();
        }
    }

    @Override
    public void onChildRemoved(@NonNull DataSnapshot dataSnapshot) {
        House house = dataSnapshot.getValue(House.class);
        int i;
        for (i = 0; i < userHouses.size() && !userHouses.get(i).id.equalsIgnoreCase(house.id); i++)
            ;
        if (i < userHouses.size()) {
            userHouses.remove(i);
            for (UpdateHouseData u : updateHouse) {
                u.Notify();
            }
        }
    }
```

```java
private static ChildEventListener reqListener = new ChildEventListener() {
    @Override
    public void onChildAdded(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        Req req = dataSnapshot.getValue(Req.class);
        if (req.getToUsers().contains(getUid()))
            requests.put(dataSnapshot.getKey(), req);
        for (UpdateRequests u : updateRequests) {
            u.Notify(requests.size());
        }
    }

    @Override
    public void onChildChanged(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
        Req r = dataSnapshot.getValue(Req.class);
        if (r.getToUsers().contains(getUid()))
            requests.put(dataSnapshot.getKey(), r);
        else if (requests.containsKey(dataSnapshot.getKey()))
                requests.remove(dataSnapshot.getKey());
        for (UpdateRequests u : updateRequests)
            u.Notify(requests.size());
    }

    @Override
    public void onChildRemoved(@NonNull DataSnapshot dataSnapshot) {
        if (requests.containsKey(dataSnapshot.getKey()))
            requests.remove(dataSnapshot.getKey());
        for (UpdateRequests u : updateRequests) {
            u.Notify(requests.size());
        }
    }

    @Override
    public void onChildMoved(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {}

    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {}
};
```

```java
//Sign up
public static void signUp(final String user, String pass, final UpdateUi u) {
    mAuth.signInWithEmailAndPassword(user, pass)
            .addOnCompleteListener((Activity) u, (task) -> {
                    if (task.isSuccessful()) {
                        // Sign in success, update UI with the signed-in user's information
                        if (userRef != null)
                            userRef.removeEventListener(userListener);
                        userRef = db.getReference( path: "users").child(getUid());
                        userRef.addValueEventListener(userListener);
                        download();
                        u.Success();
                    } else {
                        // If sign in fails, display a message to the user.
                        u.Failed(task.getException());
                    }
            });
}

//Download data
private static void download() {
    userHouses = new ArrayList<>();
    userLocks = new ArrayList<>();
    requests = new HashMap<>();
    if (houseRef != null) {
        houseRef.removeEventListener(houseListener);
        lockRef.removeEventListener(lockListener);
    }
    houseRef = db.getReference( path: "house");
    houseRef.addChildEventListener(houseListener);
    lockRef = db.getReference( path: "locks");
    lockRef.addChildEventListener(lockListener);
    reqRef = db.getReference( path: "req");
    reqRef.addChildEventListener(reqListener);
}
```

10.2.2 מחיקת מנעול

```
//deleting lock process
public static void deleteLock(String lName, String hId) {
    User user = getUser();
    Lock lock = getLockByStr(lName);
    House house = getOneHouse(hId);
    boolean dltLock = false, dltHouse = false, dltHfromU = true;

    //טיפול במנעול
    if (lock.admin.contains(getUid())) {
        //מחיקת המשתמש מרשימת האדמינים
        lock.admin = remove_id_from_string(lock.admin, getUid());
        //בדיקה אם אין עוד אדמינים
        if (lock.admin.equalsIgnoreCase( anotherString: "")) {
            //בדיקה אם יש עוד משתמשים רגילים והפיכה של אחד לאדמין
            if (!lock.notAdmin.equalsIgnoreCase( anotherString: "")) {
                String temp[] = lock.notAdmin.split( regex: ",");
                lock.admin = temp[0];
                lock.notAdmin = remove_id_from_string(lock.notAdmin, temp[0]);
            } else {
                house.locks = remove_id_from_string(house.locks, lock.id);
                dltLock = true;
                dltHouse = house.locks.equalsIgnoreCase( anotherString: "");
            }
        }
    } else
        //מחיקת המשתמש מרשימת היוזרים הרגילים
        lock.notAdmin = remove_id_from_string(lock.notAdmin, getUid());

    //מחיקת בקשות של אותו מנעול
    for (String key : requests.keySet()) {
        Req req = requests.get(key);
        if (req.getLockId().equalsIgnoreCase(lock.name))
            rjctReq(key);
    }

    //מחיקת המנעול מהרשימה של המשתמש
    user.lockList = remove_id_from_string(user.lockList, lock.id);

    //טיפול בבית
    //בדיקה אם יש עוד מנעולים לאותו משתמש בבית
    String locks[] = house.locks.split( regex: ",");
    for (String l : locks)
        if (l.length() > 0 && user.lockList.contains(l))
            dltHfromU = false;

    //אם אין למשתמש עוד מנעולים באותו בית - צריך למחוק את הבית
    if (dltHfromU) {
        user.houseList = remove_id_from_string(user.houseList, house.id);
        house.admin = remove_id_from_string(house.admin, getUid());
    }

    //עדכון או מחיקת המנעול
    if (dltLock) {
        chkReq(lock);
        db.getReference( path: "locks").child(lock.id).removeValue();
    } else
        db.getReference( path: "locks").child(lock.id).setValue(lock);

    //עדכון או מחיקת הבית
    if (dltHouse)
        db.getReference( path: "house").child(house.id).removeValue();
    else
        db.getReference( path: "house").child(house.id).setValue(house);

    //עדכון המשתמש
    userRef.setValue(user);
}
```

10.2.3 טיפול בבקשות

```
//approve request
public static void apprvReq(String houseId, String lockId, boolean admin, String reqId) {
    House house = getOneHouse(houseId);
    Lock lock = getLockByStr(lockId);
    Req req = requests.get(reqId);
    User user = users.get(req.getFromUser());

    //עדכון המנעול
    if (admin)
        lock.admin = add_id_to_string(lock.admin, req.getFromUser());
    else
        lock.notAdmin = add_id_to_string(lock.notAdmin, req.getFromUser());
    lockRef.child(lock.id).setValue(lock);
    user.lockList = add_id_to_string(user.lockList, lockId);

    //בדיקה ועדכון הבית והמשתמש לפי הצורך
    if (!user.houseList.contains(houseId)) {
        house.admin = add_id_to_string(house.admin, req.getFromUser());
        houseRef.child(houseId).setValue(house);
        user.houseList = add_id_to_string(user.houseList, houseId);
    }

    //עדכון המשתמש
    db.getReference().child("users").child(req.getFromUser()).setValue(user);

    //מחיקת הבקשה
    dltReq(reqId);
}

//reject request
public static void rjctReq(String reqId) {
    Req req = requests.get(reqId);
    req.setToUsers(remove_id_from_string(req.getToUsers(), getUid()));
    if (req.getToUsers().equalsIgnoreCase( anotherString: ""))
        dltReq(reqId);
    else
        reqRef.child(reqId).setValue(req);
}

//delete request
private static void dltReq(String reqId) {
    reqRef.child(reqId).removeValue();
}

//using to delete relevant requests when deleting lock
private static void chkReq(Lock lock) {
    Iterator it = requests.entrySet().iterator();
    while (it.hasNext()) {
        Map.Entry data = (Map.Entry) it.next();
        Req req = (Req) data.getValue();
        if (req.getLockId().equalsIgnoreCase(lock.id))
            rjctReq(data.getKey().toString());
    }
}
```

10.2.4 ממשק BLUETOOTH

```
public void updateUi(int status, List<String> msg) {
    System.out.println("testing ---> case: " + status);
    switch (status) {
        case DISCONNECT:
            sts.setText("Disconnect");
            id.setText("");
            wifi.setText("");
            signed.setChecked(false);
            wifiCheck.setChecked(false);
            info.setVisibility(View.INVISIBLE);
            button1.setVisibility(View.INVISIBLE);
            button2.setVisibility(View.INVISIBLE);
            break;
        case CONNECTING:
            if (msg == null || !msg.get(0).equalsIgnoreCase( anotherString: "SYNC"))
                bThread.write( str: "1", CONNECTING);
            else {
                bThread.write( str: GET_LID + "", GET_LID);
                sts.setText("Getting Lock details...");
                info.setVisibility(View.INVISIBLE);
            }
            break;
        case GET_LID:
            id.setText(msg.get(0));
            lid = msg.get(0);
            FireBase.searchGeneralLock(msg.get(0),  cb: this);
            sts.setText("Getting WIFI status...");
            bThread.write( str: GET_SSID + "", GET_SSID);
            break;
        case GET_SSID:
            wifi.setText(msg.get(0));
            bThread.write( str: GET_WIFI + "", GET_WIFI);
            break;
        case GET_WIFI:
            wifiCheck.setChecked(msg.get(0).equalsIgnoreCase( anotherString: "1"));
            if (button1.getText().toString().equalsIgnoreCase( anotherString: "SET WIFI")) {
                sts.setText("Scanning networks...");
                bThread.write( str: "5", GET_NET);
            } else {
                sts.setText("Connected");
                button1.setVisibility((button1.getText().length() > 0) ? View.VISIBLE : View.INVISIBLE);
            }
            break;
        case GET_NET:
            network = new ArrayList<>(msg);
            sts.setText("Connected");
            button1.setVisibility((button1.getText().length() > 0) ? View.VISIBLE : View.INVISIBLE);
            button2.setVisibility((button2.getText().length() > 0) ? View.VISIBLE : View.INVISIBLE);
            break;
        case SET_NET:
            System.out.println("testing ---> ssid = " + ssid + ", income = " + msg.get(0) + ", test = " + test);
            if (msg.get(0).equalsIgnoreCase(ssid))
                bThread.write( str: SET_PASS + pass, SET_PASS);
            else {
                if ((test++) < 3)
                    bThread.write(ssid, SET_PASS);
                else {
                    Snackbar.make(button1,  text: "Something wrong, try to set WIFI again...", Snackbar.LENGTH_SHORT).show();
                }
            }
            test = 0;
            break;
        case SET_PASS:
            System.out.println("testing ---> pass = " + pass + ", income = " + msg.get(0) + ", test = " + test);
            if (msg.get(0).equalsIgnoreCase(pass)) {
                bThread.write( str: RECONNECTING + "", RECONNECTING);
            } else {
                if ((test++) < 3)
                    bThread.write( str: SET_PASS + pass, SET_PASS);
                else {
                    Snackbar.make(button1,  text: "Something wrong, try to set WIFI again...", Snackbar.LENGTH_LONG).show();
                }
            }
            test = 0;
            break;
        case RECONNECTING:
            Snackbar.make(button1,  text: "Reconnect to check WIFI...", Snackbar.LENGTH_LONG).show();
            bThread.cancel();
    }
}
```

```
28      #define LID "6"
29      //State
30      #define MISSING_SSID_PASS 0
31      #define HAVING_SSID_PASS 1
32      #define WIFI_DISCONNECT 2
33      #define WIFI_CONNECT 3
34      #define NEW_LOCK 4
35      #define BLUETOOTH_CONNECT 5
36
37      FirebaseData FBdata;
38      FirebaseJson snapshot;
39      FirebaseJsonObject jsonParseResult;
40      SoftwareSerial bluetoothSerial(TxD, RxD);
41
42      String path = "/locks/";
43      String data;
44      char ssid[21], pass[21], lockId[] = LID;
45      char t_ssid[21],t_pass[21];
46      char state = 0, t;
47      char wifiTemp, pass_len, ssid_len,ptr, ssidNum;
48
49    void setup() {
50          //  Serial.begin(9600);
51          bluetoothSerial.begin(9600);
52          EEPROM.begin(512);
53
54          //reseting the EEPROM - LID, SSID and PASS
55          reset(LID, "Dudu phone", "pass4646");
56
57          pinMode(blue, OUTPUT);
58          pinMode(green, OUTPUT);
59          pinMode(red, OUTPUT);
60          turn_RGB(0, 0, 0);
61          path += LID;
62          path += "/status";
63          read_SSID_PASSWORD();
64          bluetoothSerial.flush();
65          t = 0;
66
67      }
```

```
68  void loop() {
69      if (bluetoothSerial.available() > 0 || t!=0)
70      {
71
72          for (char tmp = 0; tmp < 2; tmp++) {
73              turn_RGB(0, 0, 1);
74              delay(300);
75              turn_RGB(0, 0, 0);
76          }
77          if(bluetoothSerial.available())
78              t = bluetoothSerial.read();
79          delay(15);
80          switch (t) {
81
82          case '1':
83              bluetoothSerial.write(1);
84              delay(15);
85              bluetoothSerial.write(4);
86              delay(15);
87              bluetoothSerial.write("SYNC");
88              t = 0;
89              break;
90          case '2':
91              bluetoothSerial.write(1);
92              delay(15);
93              bluetoothSerial.write(strlen(lockId));
94              delay(15);
95              for (char i = 0; i < strlen(lockId); i++) {
96                  bluetoothSerial.write(lockId[i]);
97                  delay(15);
98              }
99              t = 0;
100             break;
101         case '3':
102             bluetoothSerial.write(1);
103             delay(15);
104             bluetoothSerial.write(ssid_len);
105             delay(15);
106             for (char i = 0; i < ssid_len; i++) {
107                 bluetoothSerial.write(ssid[i]);
```

```
105            delay(15);
106            for (char i = 0; i < ssid_len; i++) {
107                bluetoothSerial.write(ssid[i]);
108                delay(15);
109            }
110            t = 0;
111            break;
112        case '4':
113            bluetoothSerial.write(1);
114            delay(15);
115            bluetoothSerial.write(1);
116            delay(15);
117            bluetoothSerial.write(((WiFi.status() == WL_CONNECTED)&&(strlen(pass)>0)) ? '1' : '0');
118            delay(15);
119            t = 0;
120            break;
121
122        case '5':
123            ssidNum = WiFi.scanNetworks();
124            bluetoothSerial.write(ssidNum);
125            delay(15);
126            for (char i = 0; i < ssidNum; i++) {
127                String str = WiFi.SSID(i).c_str();
128                delay(15);
129                bluetoothSerial.write(str.length());
130                delay(15);
131                for (char j = 0; j < str.length(); j++) {
132                    bluetoothSerial.write(str[j]);
133                    delay(15);
134                }
135            }
136            t = 0;
137            break;
138        case '6':
139            ptr = 0;
140            while (bluetoothSerial.available() > 0)
141                t_ssid[ptr++] = bluetoothSerial.read();
142            t_ssid[ptr] = '\0';
143            bluetoothSerial.write(1);
144            delay(15);
```

```
145                 bluetoothSerial.write(strlen(t_ssid));
146                 for (char i = 0; i < strlen(t_ssid); i++) {
147                     bluetoothSerial.write(t_ssid[i]);
148                     delay(15);
149                 }
150                 t = 0;
151                 break;
152             case '7':
153                 ptr = 0;
154                 while (bluetoothSerial.available() > 0)
155
156                     t_pass[ptr++] = bluetoothSerial.read();
157                 t_pass[ptr] = '\0';
158                 bluetoothSerial.write(1);
159                 delay(15);
160                 bluetoothSerial.write(strlen(t_pass));
161                 for (char i = 0; i < strlen(t_pass); i++) {
162                     bluetoothSerial.write(t_pass[i]);
163                     delay(15);
164                 }
165                 t = 0;
166                 break;
167
168             case '8':
169                 bluetoothSerial.write(1);
170                 delay(15);
171                 bluetoothSerial.write(1);
172                 delay(15);
173                 bluetoothSerial.write('1');
174                 delay(15);
175                 write_SSID_PASSWORD(t_ssid, t_pass);
176                 bluetoothSerial.flush();
177                 t = 0;
178                 break;
179             }
180         }
181         else {
182             switch (state) {
183             case MISSING_SSID_PASS:
184                 for (char t = 0; t < 3; t++) {
```

```
185                 turn_RGB(1, 0, 0);
186                 delay(300);
187                 turn_RGB(0, 0, 0);
188                 delay(300);
189             }
190             read_SSID_PASSWORD();
191             break;
192         case HAVING_SSID_PASS:
193             connect_to_WIFI_and_FB();
194             break;
195         case WIFI_CONNECT:
196             read_lock_status();
197             break;
198         case NEW_LOCK:
199             for (char i = 0; i < 2; i++) {
200                 turn_RGB(1, 1, 1);
201                 delay(300);
202                 turn_RGB(0, 0, 0);
203                 delay(300);
204             }
205             read_lock_status();
206             break;
207         }
208     }
209
210 }
211
212 //connecting to WIFI and FIREBASE
213 void connect_to_WIFI_and_FB() {
214     if (wifiTemp == 0) {
215         wifiTemp++;
216         WiFi.begin(ssid, pass);
217     }
218     //  Serial.print("connecting");
219     if (WiFi.status() != WL_CONNECTED) {
220         turn_RGB(1, 1, 0);
221         delay(300);
222         turn_RGB(0, 1, 1);
223         delay(300);
224         turn_RGB(1, 0, 1);
```

```
        delay(300);
    }
    else {
        Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
        Firebase.reconnectWiFi(true);
        state = WIFI_CONNECT;
        turn_RGB(0, 0, 0);
    }
}

//read SSID and PASSWORD form EEPROM
void read_SSID_PASSWORD() {
    char len = 0;
    char ch;
    ch = EEPROM.read(ssid_ptr);
    while (ch != '\0' && len < 21) {
        ssid[len++] = ch;
        ch = EEPROM.read(ssid_ptr + len);
    }
    ssid[len] = '\0';
    ssid_len = len;
    len = 0;
    ch = EEPROM.read(pass_ptr);
    while (ch != '\0' && len < 21) {
        pass[len++] = ch;
        ch = EEPROM.read(pass_ptr + len);
    }
    pass[len] = '\0';
    pass_len = len;

    if (ssid[0] != '\0' )
        state = HAVING_SSID_PASS;

    wifiTemp = 0;
}
```

```
261     //write SSID and PASSWORD to EEPROM
262    void write_SSID_PASSWORD(char s[20], char p[20]) {
263        char i;
264        for (i = 0; s[i] != '\0'; i++)
265            EEPROM.write(ssid_ptr + i, s[i]);
266        EEPROM.write(ssid_ptr + i, s[i]);
267        for (i = 0; p[i] != '\0'; i++)
268            EEPROM.write(pass_ptr + i, p[i]);
269        EEPROM.write(pass_ptr + i, p[i]);
270        EEPROM.commit();
271        state = MISSING_SSID_PASS;
272    }
273
274     //write lock id to EEPROM
275    void write_lock_id(char id[6]) {
276        char i;
277        for (i = 0; i < id[i] != '\0'; i++)
278            EEPROM.write(id_ptr + i, id[i]);
279        EEPROM.write(id_ptr + i, id[i]);
280    }
281
282     //read lock status
283    void read_lock_status() {
284        if (Firebase.getString(FBdata, path)) {
285            state = WIFI_CONNECT;
286            if (FBdata.stringData().equalsIgnoreCase("open"))
287                turn_RGB(0, 1, 0);
288            else if (FBdata.stringData().equalsIgnoreCase("close"))
289                turn_RGB(1, 0, 0);
290            else if (FBdata.stringData().equalsIgnoreCase("lock"))
291                turn_RGB(0, 0, 1);
292            else
293                state = NEW_LOCK;
294        }
295        else
296            if (WiFi.status() != WL_CONNECTED)
297                state = HAVING_SSID_PASS;
298            else
299                state = NEW_LOCK;
300    }
301
302     //turn leds on/off
303    void turn_RGB(char r, char g, char b) {
304        digitalWrite(red, r == 0 ? LOW : HIGH);
305        digitalWrite(green, g == 0 ? LOW : HIGH);
306        digitalWrite(blue, b == 0 ? LOW : HIGH);
307    }
308
```