תיכון ברנר פתח תקוה משרד החינוך

**עבודת גמר הנדסת תוכנה**

**הנושא: PTS  
 לוגו הפרויקט**



שם התלמיד: עידו סיגלר

תעודת זהות: 328364526

שם המנחה: אמיר דבי

שם החלופה: הגנת סייבר ומערכות הפעלה

**מאי 2024 תשפ"ד (לבדוק)**

**תוכן עניינים**

[**1.** **מבוא** 4](#_Toc161055730)

[**1.1.** **ייזום** 4](#_Toc161055731)

[**תקציר הפרויקט** 4](#_Toc161055732)

[**הגדרת הלקוח** 4](#_Toc161055733)

[**יעדים ומטרות** היעד של הפרויקט היא אפליקציה שאפשר להוריד ממנה קבצים בין אנשים באופן חלק ועוד פעולות כמו להעלות ולמחוק קבצים. 4](#_Toc161055734)

[**בעיות תועלות וחסרונות** 4](#_Toc161055735)

[**פתרונות קיימים** 5](#_Toc161055736)

[**טכנולוגיית הפרויקט** 5](#_Toc161055737)

[**תחומי הפרויקט** 5](#_Toc161055738)

[**1.2.** **אפיון המערכת** 6](#_Toc161055741)

[**תיאור הפרויקט** 6](#_Toc161055742)

[**מאפיינים ויכולות** 6](#_Toc161055743)

[**בדיקות המערכת** 6](#_Toc161055744)

[**תכנון וניהול לוח זמנים** 6](#_Toc161055745)

[**ניהול סיכונים** 7](#_Toc161055746)

[**8.** **תיאור מפורט של המערכת** 7](#_Toc161055747)

[**2.1.** **פירוט וניתוח יכולות המערכת** 7](#_Toc161055749)

[**פירוט היכולות בצד שרת** 7](#_Toc161055750)

[שרת לקוחות: 7](#_Toc161055751)

[**פירוט היכולות בצד לקוח** 9](#_Toc161055752)

[**9.** **ארכיטקטורה** 10](#_Toc161055753)

[**3.1.** **ארכיטקטורת המערכת** 10](#_Toc161055757)

[**תיאור החומרה** 10](#_Toc161055758)

[**סכימת בלוקים** 10](#_Toc161055759)

[**3.2.** **טכנולוגיית המערכת** 11](#_Toc161055760)

[**שפת תכנות** 11](#_Toc161055761)

[**תקשורת** 11](#_Toc161055762)

[**3.3.** **זרימת המידע במערכת** 12](#_Toc161055763)

[**3.4.** **אלגוריתמים עיקריים** 12](#_Toc161055770)

[**3.5.** **סביבת הפיתוח** 13](#_Toc161055778)

[**כלי פיתוח נדרשים** 13](#_Toc161055779)

[**סביבת הבדיקות** 13](#_Toc161055780)

[**3.6.** **פרוטוקול התקשורת** 13](#_Toc161055781)

[**3.7.** **מסכי המערכת/ממשק משתמש** 14](#_Toc161055791)

[**3.8.** **מסד הנתונים** 14](#_Toc161055802)

[**3.9.** **חולשות ואיומים** 15](#_Toc161055814)

[**10.** **מימוש הפרויקט** 16](#_Toc161055815)

[פירוט מימוש המערכת (המשך שלב העיצוב) 16](#_Toc161055816)

[**4.1.** **מודולים ומחלקות** 16](#_Toc161055818)

[**מודולים ומחלקות מיובאים** 16](#_Toc161055819)

[**מודולים ומחלקות שפותחו בפרויקט** 16](#_Toc161055820)

[**4.2.** **קטעי קוד עבור אלגוריתמים מרכזיים** 17](#_Toc161055826)

[**4.3.** **בדיקות** 17](#_Toc161055833)

[**11.** **מדריך למשתמש** 18](#_Toc161055834)

[**5.1.** **עץ קבצים** 18](#_Toc161055840)

[**5.2.** **תרשים מסכים** 18](#_Toc161055841)

[**5.3.** **צילום מסך פתיחה והסבר GUI** 18](#_Toc161055849)

[**5.4.** **צילום מסכי האפליקציה בליווי הסברים על פקדים** 18](#_Toc161055850)

[**5.5.** **הבנת flow עבודה במערכת** 18](#_Toc161055851)

[**5.6.** **התקנת המערכת** 19](#_Toc161055862)

[**5.7.** **משתמשי המערכת** 19](#_Toc161055874)

[**12.** **סיכום אישי/רפלקציה** 19](#_Toc161055875)

[**13.** **ביבליוגרפיה** 19](#_Toc161055876)

[**14.** **נספחים** 20](#_Toc161055877)

# **מבוא**

## **ייזום**

**תקציר הפרויקט**

הפרויקט PTS הוא תוכנה שמאפשרת הורדת קבצים בין משתמשים בדומה לפרוטוקול BitTorrent במערכת סגורה. הפרויקט פועל בארכיטקטורת שרת – לקוח מבוסס sockets של פיתון, והוא נועד לכול קבוצה של אנשים שרוצה לשתף בין כול המשתתפים קבצים בצורה מהירה ובטיחותית. המערכת אמורה לאפשר לקבוצה של אנשים להרים שרת Tracker ולהעלות אליו קבצים ולהוריד קבצים "השמורים" על השרת. הפרויקט יכיל ממשק בצד לקוח שיהיה בflask ויהיה שרת נוסף ששומר את בסיס הנתונים על כול המשתמשים בsqllite. מסד נתונים נוסף נמצא על כול Tracker ומסד הנתונים הזה ישמור מידע על כול קובץ שמעלים אליו. כל מידע שיעבור ברשת יהיה מוצפן.

בחרתי בפרויקט הזה מכיוון שהנושא הזה מאוד עניין אותי והפרויקט הקודם שלי גם התבסס על טכנולוגיה דומה.

האתגר המרכזי שאני צופה לפרויקט הוא לשלב בין הטכנולוגיה של הורדת הקבצים ואיך לשלב אותה בדרך שיראה חלקה עם הממשק בצד לקוח בצורה שתראה טובה

### **הגדרת הלקוח**

המערכת מיועדת לכול חברה, ארגון, סטארטאפ או כול התקבצות מסוימת של אנשים שרוצה להעביר ביניהם קבצים בדרך קלה ומהירה ונוח לשימוש

### **יעדים ומטרות** היעד של הפרויקט היא אפליקציה שאפשר להוריד ממנה קבצים בין אנשים באופן חלק ועוד פעולות כמו להעלות ולמחוק קבצים.

המטרה המרכזית של הפרויקט היא לפתור את הבעיה שלהוריד קובץ היום זה נורא איטי מכיוון שכול מי שרוצה להוריד את הקובץ פונה לאותו שרת מה שיוצר עומס על השרת. האפליקציה פותרת את הבעיה בכך שאין שרת אחד שמחזיק את כול הקובץ אלא כול מי שיש ברשותו כבר את הקובץ יכול לשתף אותו עם עוד אנשים.

### **בעיות תועלות וחסרונות**

הבעיה שאנחנו מנסים לפתור היא שהורדת קבצים היום היא בעייתית, הקובץ נמצא על שרת אחד ואם השרת או איטי או נמצא מתחת להמון לחץ אז ההורדה תהיה איטית.

המערכת שלנו פותרת את זה בכך שהיא גורמת ללקוחות להיות השרתים ולהוריד את הקובץ אחד מהשני במקום משרת אחד מרכזי.

### **פתרונות קיימים**

המערכת עובדת בצורה שמאוד דומה לפרוטוקול הbittorrent ולכן כול מערכת שתשתמש בפרוטוקל הזה היא בבסיסה דומה מאוד למערכת שלנו, האפליקציות הכי מפורסמות שמשתמשות בפרוטוקול הזה הן:

Qbittorrent

Utorrent

bittorrent

### **טכנולוגיית הפרויקט**

טכנולוגית המערכת דומה מאוד לפרוטוקול הbittorrent והיא בעצם גרסה מצומצמת ופרטית של הפרוטוקול הזה.

### **תחומי הפרויקט**

אפליקציה מאובטחת בארכיטקטורת שרת – לקוח

המערכת משתמשת בflask בשביל להציג ממשק גרפי אינטרנטי ללקוח.

המערכת עוסקת עם מסדי נתונים שנמצאים על השרתים

המערכת עוסקת בפרוטוקול tcp ופרוטוקול http

המערכת עוסקת בהצפנה של מידע

המערכת עוסקת בthreading

המערכת עוסקת בשמירת וקריאת קבצים

המערכת עוסקת בהצפנת סיסמאות על ידי hash



## **אפיון המערכת**

### **תיאור הפרויקט**

המשתמש אמור להתחבר לחשבון שלי במידה ויש לו, במידה ואין לו עליו להירשם עם פרטים אישים. משם הוא יכול לבצע את כול הפעולות שהמערכת מאפשרת כגון: הורדת קבצים, העלאת קבצים, מחיקת קבצים, החלפה של טראקר והתנתקות מהמערכת

### **מאפיינים ויכולות**

1. רישום למערכת

1. התחברות למערכת
2. הורדת קבצים
3. העלאת קבצים
4. מחיקת קבצים
5. החלפת טראקר
6. התנתקות

### **בדיקות המערכת**

בדיקה שהמערכת לא קורסת – לנסות להביא את המערכת למצבי קיצון ולראות איך היא מסתדרת

בדיקה שההרשמה וההתחברות עובדת – להירשם ולהתחבר למערכת ולראות שהכול עובד

בדיקה שהורדת הקבצים עובדת – להוריד קבצים ולראות שהכול בסדר.

### **תכנון וניהול לוח זמנים**

וואלה לא זוכר תאריכים

### **ניהול סיכונים**

סיכון: בגלל שהתחלתי את הפרויקט בשלב מאוחר של השנה יש סיכוי שלא אספיק לסיים אותו

פתרון: אני לא אשן בלילה ואעבוד עד שהוא יהיה מוכן

# **תיאור מפורט של המערכת**



## **פירוט וניתוח יכולות המערכת**

**פירוט היכולות בצד שרת**

שרת לקוחות:

|  |  |  |
| --- | --- | --- |
| מספר | שם היכולת | הסבר היכולת |
| 1. | הוספת מידע לבסיס הנתונים | להוסיף מידע לבסיס הנתונים |
| 2. | שליפת מידע מבסיס הנתונים | לשלוף מידע מבסיס הנתונים |
| 3. | הרשמה | לרשום משתמש חדש למערכת במידה ועובר את הבדיקות הנחוצות |
| 4. | התחברות | לבדוק אם פרטי ההתחברות נכונים ומתאימים לנתונים במסד הנתונים |
| 5. | יצירת מסד נתונים חדש | יצירת טבלה חדשה אם לא קיימת אחת |
| 6. | בדיקת משתמש קיים | לקבל מידע על משתמש ולבדוק אם הוא קיים או לא. |
| 7. | קבלת מידע על משתמש | לקבל אימייל של משתמש ולהחזיר את המידע עליו |
| 8. | מחיקת משתמש | למחוק משתמש קיים |
| 9. | שינוי טראקר | לשנות את הטראקר המרכזי של משתמש מסוים. |

טראקר:

|  |  |  |
| --- | --- | --- |
| מספר | שם היכולת | הסבר היכולת |
| 1. | הוספת מידע לבסיס הנתונים | להוסיף מידע לבסיס הנתונים |
| 2. | שליפת מידע מבסיס הנתונים | לשלוף מידע מבסיס הנתונים |
| 3. | שליחת הקבצים | שליחה של מידע של כול הקבצים שנמצאים במסד הנתונים. |
| 4. | שמירת קובץ חדש | שמירה של המידע של הקובץ החדש מהלקוח במסד הנתונים |
| 5. | יצירת מסד נתונים חדש | יצירת טבלה חדשה אם לא קיימת אחת |
| 6. | התחלת הורדה של קובץ | שליחה ללקוח את המידע הנחוץ על הקובץ שהוא רוצה להתחיל להוריד |
| 7. | סיום הורדת של קובץ | להוסיף למידע של הקובץ את המידע הנחוץ כאשר הלקוח מסיים להוריד קובץ |
| 8. | מחיקת קובץ | למחוק מידע של קובץ ממסד הנתונים |
| 9. | קבלת מידע של הטראקר | לקבל את המידע על הטראקר. |

# **פירוט היכולות בצד לקוח**

|  |  |  |
| --- | --- | --- |
| מספר | שם היכולת | הסבר היכולת |
| 1. | התחברות | התחברות למשתמש קיים |
| 2. | הרשמה | הרשמה של משתמש חדש. |
| 3. | בקשת חלק מקובץ | בקשה של חלק מקובץ מלקוח אחר |
| 4. | העלאת קובץ | העלאת מידע על קובץ חדש לטראקר |
| 5. | הורדת קובץ | להוריד קובץ מלקוחות אחרים אחרי קבלת מידע מהטראקר |
| 6. | החלפת טראקר | להחליף את החיבור מטראקר אחד לטראקר אחר |
| 7. | התנתקות | התנתקות ממשתמש. |
| 8. | מחיקת משתמש | מחיקה של משתמש קיים |
| 9. | רענון המידע מהטראקר | לרענן ולקבל מחדש את המידע מהטראקר. |

# **ארכיטקטורה**

פירוט ההחלטות שנלקחו למימוש המערכת (עיצוב:(

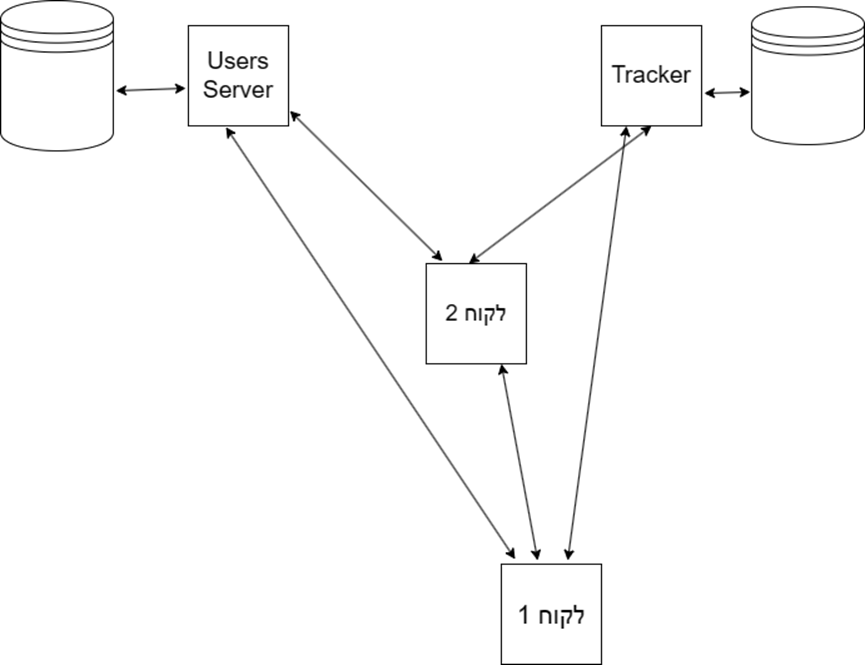


## **ארכיטקטורת המערכת**

**תיאור החומרה**

??????

**סכימת בלוקים**



## **טכנולוגיית המערכת**

**שפת תכנות**

הפרויקט משתמש בארבע שפות תכנות עיקריות: phyton, JavaScript, html, CSS  
**מערכת הפעלה**

מערכת ההפעלה המרכזית של הפרויקט היא windows. זוהי מערכת העפלה שהפרויקט נבנה עליו ובשבילה. אין לדעת מה יקרה אם ננסה להפעיל את הפרויקט במערכת הפעלה אחרת אבל סביר להניח שהפרויקט לא יעבוד כמתוכנן או בגלל.

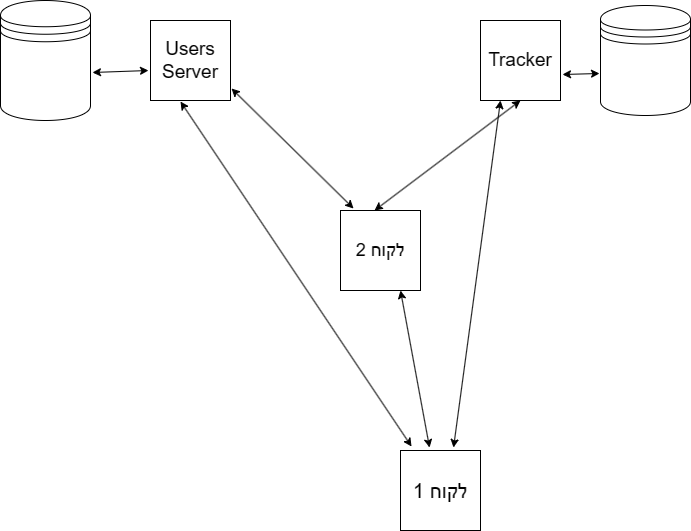
**תקשורת**

התקשורת בין הלקוח לבין כול השרתים מתבצעת על ידי פרוטוקול TCP

התקשורת בין הלקוח לבין הממשק הגרפי נעשה על ידי פרוטוקול HTTP

## **זרימת המידע במערכת**

1. המידע זורם מהממשק הגרפי ללקוח.
2. המידע זורם מהלקוח לשרת.
3. המידע זורם מהשרת למסד הנתונים
4. המידע זורם מהשרת בחזרה ללקוח
5. המידע זורם בין כול לקוח ללקוח אחר.





## **אלגוריתמים עיקריים**

הבעיה האלגוריתמית העיקרית היא איך מורידים חלקים של קובץ מלקוח אחר ואיך מחברים את החלקים האלה לקובץ המלא.

הפתרון שבחרתי הוא שלכול לקוח יהיה גם שרת שרץ איתו שיכול לקבל בקשות של קבצים, ברגע שהוא מקבל בקשה לקובץ הוא הולך לקובץ עצמו ומוציא ממנו ישירות את המידע המבוקש. לאחר שמקבלים את כול המידע המבוקש בודקים שהמידע הגיע כמו שצריך בעזרת hash של כול חלק שקיבלנו מהטראקר.

לאחר מכן אנחנו שומרים כול חלק מהקובץ קקובץ נפרד ובלתי נראה במחשב ואז מחברים בינהם וככה האלגוריתם של הורדת הקובץ עובד



## **סביבת הפיתוח**

**כלי פיתוח נדרשים**

מחשב, חיבור לאינטרנט, תוכנה שיכולה לכתוב ולהריץ קוד של פייתון, פייתון מותקן על המחשב וחשבון אימייל.

**סביבת הבדיקות**

צריך לפחות 2 מחשבים עם חיבור לאינטרנט ופייתון מותקן עליהם

## **פרוטוקול התקשורת**

יש במערכת שלושה פרוטוקלי תקשורת:

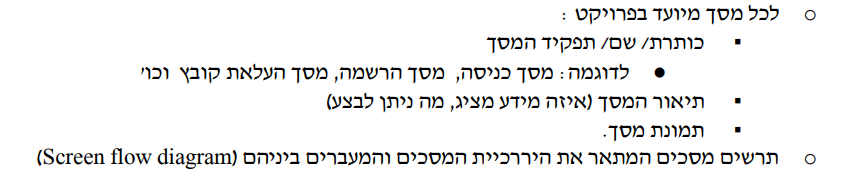
פרוטוקול TCP

פרוטוקול HTTP

פרוטוקול התקשורת שאני פיתחתי



## **מסכי המערכת/ממשק משתמש**





## **מסד הנתונים**

**טבלת משתמשים:**

מפתח משתמש (ID). דוגמה לא מוצפנת – 1

שם פרטי (firstName). דוגמה לא מוצפנת – ido

שם משפחה (lastName). דוגמה לא מוצפנת – sigler

אימייל (email). דוגמה לא מוצפנת – [edosigler3@gmail.com](mailto:edosigler3@gmail.com)

סיסמא מוצפנת (passHash). דוגמה לא מוצפנת – d08971234kljasd

דרגה (rank). דוגמה לא מוצפנת – visitor

"מלח" (salt). דוגמה לא מוצפנת – GjhJHlgGn

טראקר (tracker). דוגמה לא מוצפנת – {ip: "10.10.11.127", port:6987}

**טבלת קבצים:**

מפתח קובץ (ID). דוגמא לא מוצפנת – 1

שם הקובץ (fileName). דוגמא לא מוצפנת – test.txt

גודל הקובץ (fileSize). דוגמא לא מוצפנת – 1203945

גודל חלק (pieceSize). דוגמא לא מוצפנת – 678

כמות של חלקים (amountOfPieces). דוגמא לא מוצפנת – 10

דרגת קובץ (fileVisibility). דוגמא לא מוצפנת – visitor

מחזיקי הקובץ (fileOwners). דוגמא לא מוצפנת: {(ip, port), (ip, port)}

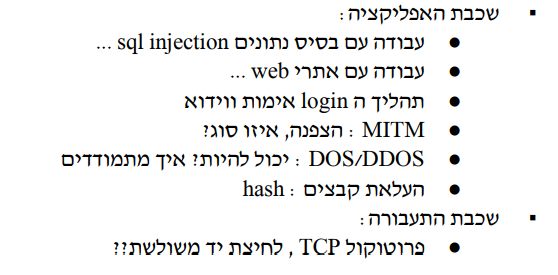
מעלה הקובץ (fileUploader). דוגמא לא מוצפנת –

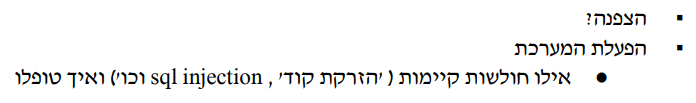
רשימה של האשים (listOfHashes). דוגמא לא מוצפנת –

כמות הורדות (numberOfDownloads). דוגמא לא מוצפנת - 3



## **חולשות ואיומים**





# **מימוש הפרויקט**

# פירוט מימוש המערכת (המשך שלב העיצוב)



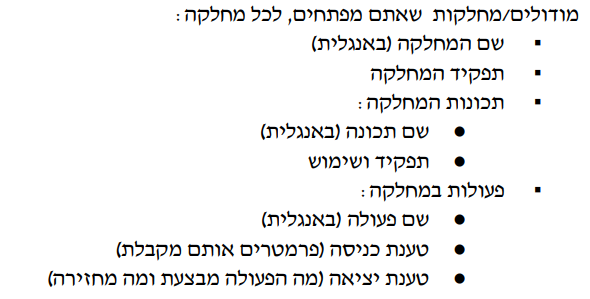
## **מודולים ומחלקות**

סקירת כל המודולים / מחלקות המרכיבים את המערכת וקשרי הגומלין ביניהם

**מודולים ומחלקות מיובאים**



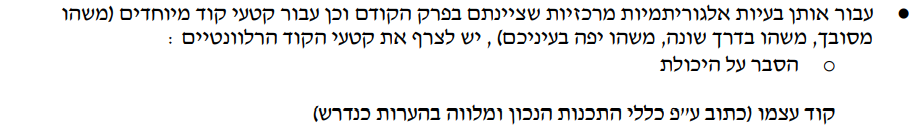
**מודולים ומחלקות שפותחו בפרויקט**



**להוסיף תרשימי UML לכל מחלקה ותרשים קשרי גומלין בין המחלקות.**

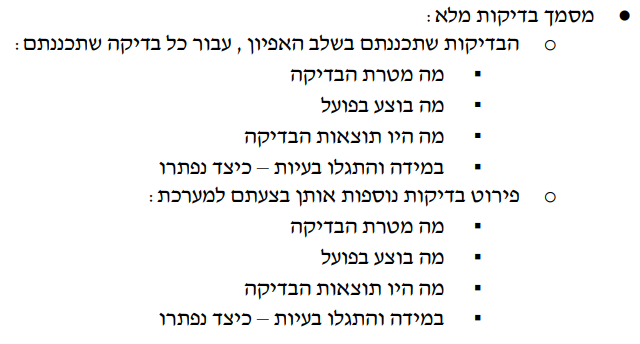


## **קטעי קוד עבור אלגוריתמים מרכזיים**





## **בדיקות**



# **מדריך למשתמש**



## **עץ קבצים**

פירוט כלל קבצי המערכת – עץ קבצים

## **תרשים מסכים**

תרשים המתאר את היררכית המסכים



## **צילום מסך פתיחה והסבר GUI**

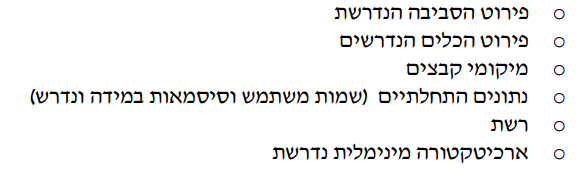
## **צילום מסכי האפליקציה בליווי הסברים על פקדים**

## **הבנת flow עבודה במערכת**

הסבר מפורט המתאר איך המשתמש מפעיל את כל היכולות באפליקציה דרך המסכים (ממשק משתמש)

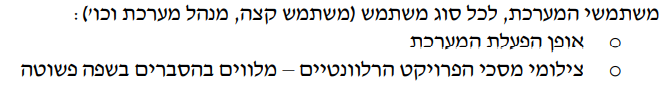


## **התקנת המערכת**

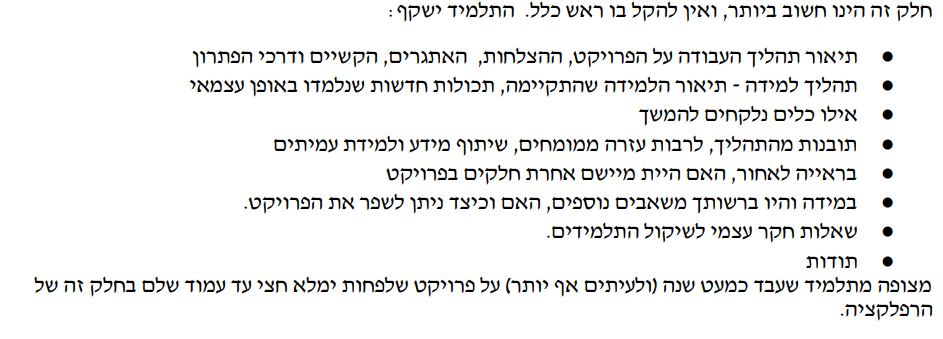




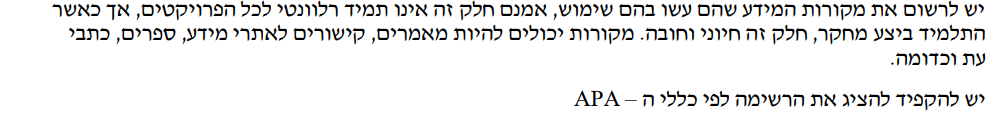
## **משתמשי המערכת**



# **סיכום אישי/רפלקציה**



# **ביבליוגרפיה**



# **נספחים**

