עבודה בתכנון ותכנות מערכות

התמחות בהגנת סייבר

מספר שאלון 883589 (5 יח"ל)

**Eagle Eye**

מערכת לניטור ובקרת תקשורת



שם התלמיד: אופק ארז

מספר זהות: 214273393

שם בית ספר: קריית חינוך אמירים

עיר: ראשון לציון

שם המנחה: מוטי מתתיהו

מועד הגשת המסמך:

תוכן עניינים

תוכן עניינים 2

מבוא 4

לו"ז פיתוח פרוייקט 5

פונקציונליות המערכת 6

דרישות המערכת 6

דרישות פונקציונליות 6

תיאור בסיס הנתונים במערכת 9

שדות בסיס הנתונים 10

הסבר השדות 10

מודולים 10

קבצי השרת: 11

קבצי הלקוח: 11

הגדרת תקשורת (RFC) 13

תיאור התקשורת בצד הלקוח 13

תיאור התקשורת בצד של השרת 13

פעולות שרת-לקוח 14

צילומי מסך 15

מסך הבית 15

מסך מידע אודות המערכת 16

מסכי ההרשמה 17

מסך ההתחברות 18

18

מסכי איפוס סיסמה 19

19

מסך הקנבס 20

תיעוד מחקר 21

אתגר – הצפנה 22

אתגר – מקביליות מערכת 22

רפלקציה אישית 24

תודות 24

ובליוגרפיה (על פי כללי כתיבה אקדמית) 25

נספחים 26

מבוא

בחרתי לפתח תוכנה לניטור תעבורת רשת מפני שכאשר השתמשתי בתוכנה הקיימת לניטור חבילות מידע – Wireshark, הרגשתי כי ניתן לעשות זאת בצורה נוחה ואינטואיטיבית יותר בשל עומס ועודף הנתונים אשר ברוב המקרים אינם רלוונטיים לבדיקות שגרתיות של תעבורת הרשת.

הפרויקט הוא מערכת אשר מטרתה לאפשר אבחון וניטור של תעבורת הרשת העוברת דרך המחשב.

כמו כן, מטרת הפרויקט הוא להוות כלי נוח ואינטואיטיבי לאנשי תקשורת על מנת לעבור על תעבורת הרשת בארגון, או אף במערכת הביתית שלהם, ולבודקי חוסן (penetration testers) לחקור את התעבורה במחשב מסוים אשר עליו הוא מבצע בדיקה והפורטים הפתוחים בו, על מנת לאבחן האם קיימות פגיעויות אפשריות הנובעות מהגדרת הרשת במחשב או בארגון.

**תקציר הפרוייקט**

Eagle - Eye תאפשר:

1. מיפוי העמדות הפעילות ברשת והצגת כתובות הIP שלהן.
2. ביצוע בדיקות תעבורה על מספר מחשבים במקביל.
3. ראיית תוכן חבילות המידע שנקלטו.
4. הצגה של המידע בצורה מסודרת על פי פרוטוקולים.
5. סינון ומיון הפאקטות על פי זמן הקבלה, כתובת האיי פי ממנה התקבלו, הפורט וכו'..
6. סריקת פורטים בכל המחשבים הנשלטים הן פורטי TCP והן פורטי UDP.
7. בחינה של תוצאות בדיקות אשר נעשו בעבר דרך המשתמש.
8. יצירת Reverse Shell בכל אחד מהמחשבים עליהם מותקנת התוכנה, מה שמאפשר השתלטות מרחוק כל זמן שהעמדה פעילה ומחוברת לController.

**רכיבי המערכת**

1. שרת((Server – השרת מאכסן לי את מסד הנתונים המכיל את נתוני המשתמשים ואת טבלת הבדיקות שכל משתמש עשה עד כה.
2. Controller – מחשב עליו נמצאת התוכנה, וממנו ניתן לבצע פעולות על כל המחשבים הנוספים בLAN המכילים את התוכנה. דרך מחשב זה בלבד מתחברים למשתמש/נרשמים כמשתמש.
3. לקוחות(Clients) – כל המחשבים עליהם מותקנת התוכנה מלבד הController ועליהם מתבצעות הבדיקות בהוראת הController-.

**שלב א'**

ראשית הController מתחבר אל השרת בתור משתמש. לאחר מכן, השרת מחזיר אל הController את רשימת המחשבים הפעילים בהם קיימת התוכנה.

**שלב ב'**

הController בוחר מחשב לבצע עליו בדיקה ושולח זאת כבקשה לשרת.

**שלב ג'**

השרת מעביר את הבקשה אל המחשב שנבחר והתוכנה מופעלת על המחשב.

**שלב ד'**

המחשב מעביר את תוצאות הבדיקה אל השרת, השרת שומר את התוצאות בטבלאות מסד הנתונים הנמצא עליו.

**שלב ה'**

השרת מעביר את תוצאות הבדיקה אל הController על מנת שיאבחנן.

תמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטיתמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטיתמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטי

**Client**

**Client**

**Client**

**Server**

תמונה שמכילה מקורה, שולחן, ישיבה, שולחן כתיבה

התיאור נוצר באופן אוטומטי

**Client-Server connection**

תמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטי

**Controller**

בנוסף, במידה וה-Controller ירצה להתחבר אל אחד מן המחשבים באמצעות Reverse Shell הדבר יראה כך:

**שלב א'**

ראשית הController שולחopcode מסוים שיזוהה עם הפעולה אל המחשב עליו הוא רוצה לשלוט.

**שלב ב'**

המחשב אשר קיבל את הבקשה וזיהה את סוגה ישלח בקשת חיבור TCP אל

ה -Controller אשר תאפשר את שיתוף הshell של המחשב הנשלט.

**שלב ג'**

ה -Controller שולח את הפקודות אותן הוא רוצה לשלוח למחשב הנשלט, המחשב הנשלט מבצע את הפקודות ושולח לController את סטטוס הפקודות.

שלב א:

תמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטיתמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטי

**Reverse shell opcode**

**Client**

**Controller**

שלב ב:

תמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטיתמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטי

**TCP Connection Request**

**Controller**

**Client**

שלב ג:

תמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטיתמונה שמכילה אלקטרוניקה, מחשב, ישיבה, מחשב נישא

התיאור נוצר באופן אוטומטי

**TCP Connection Established**

**Controller**

**Client**

לו"ז פיתוח פרוייקט

|  |  |
| --- | --- |
| יעד (מה יש לבצע) | זמן ביצוע (מתאריך עד תאריך) |
| הגשת הצעת ותיאור הפרויקט כולל MVP | 21.11.2021 |
| תכנון פרוטוקול תקשורת בדומה למסמך בקורס Networks.py |  |
| פיתוח ליבת הקוד  הוכחת יכולת תקשורת לפי הנדרש בפרוייקט שרת ולקוח ((Threads |  |
| פיתוח ממשק גרפי |  |
| בנייה ופיתוח מסד נתונים |  |
| בדיקות |  |
| סיום פיתוח |  |
| מצגת שיווקית לפרויקט + סיום תיקון שגיאות |  |
| הגשה סופית של הפרויקט |  |

פונקציונליות המערכת

1. הרשמה ראשונית למערכת.

2. התחברות משתמש קיים למערכת (Log In).

3. אפשרות איפוס סיסמה.

4. ציור על הקנבס:

* בחירת/שינוי צבע.
* בחירת/שינוי גודל מברשת.
* בחירת/שינוי סוג מברשת.

6. המשך ציור.

7. התנתקות מהמערכת.

**דרישות פונקציונליות**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **דרישה** | **קלט** | **תהליך** | **פלט** | **טיפול בשגיאות** |
| **1** | הרשמה ראשונית למערכת – רישום של משתמש חדש | נתוני המשתמש החדש (אימייל, שם משתמש נבחר, סיסמה נבחרת). | העברת נתוני המשתמש המתקבלים לשרת ושמירתם בבסיס הנתונים. | הודעה המודיעה כי הלקוח נרשם בהצלחה למערכת, ומעבר למסך הבית. | תוחזר הודעת שגיאה במקרה של:  - אימייל לא תקין/תפוס.  - שם משתמש תפוס.  - סיסמה קצרה מידי. |
| **2** | התחברות משתמש קיים למערכת | נתוני המשתמש הקיים (שם משתמש, סיסמה). | העברת נתוני המשתמש המתקבלים לשרת ובדיקתם בבסיס הנתונים. | הודעה המודיעה כי הלקוח התחבר בהצלחה למערכת, ומעבר למסך הבית. | תוחזר הודעת שגיאה אם:  - לפחות מהנתונים שהתקבלו לא מתאים לנתונים הקיימים מראש במסד הנתונים.  - המשתמש כבר מחובר למערכת. |
| **3** | אפשרות איפוס סיסמה | אימייל ושם משתמש. | ביצוע בדיקת התאמה בין האימייל ושם המשתמש הנקלטים. אם הם תואמים זה לזה, כלומר קיים משתמש זה במערכת, ישלח לאימייל קוד לשחזור סיסמה. המשתמש יתבקש להקליד קוד זה, ובמידה והקוד נכון תינתן אפשרות לשינוי סיסמה. | מסך המאפשר איפוס ושינוי סיסמה. | תוחזר הודעת שגיאה במקרה וכתובת האימייל שהתקבלה לא תקינה.  תוחזר הודעה מתאימה במקרה ואין התאמה בין האימייל ושם המשתמש שהתקבלו, כלומר לא קיים משתמש כזה במערכת. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **4.1** | בחירת/שינוי צבע | הצבע הנבחר על ידי המשתמש. | שינוי צבע המברשת לצבע הנבחר. | מימוש בחירת המשתמש. | אין. |
| **4.2** | בחירת/שינוי גודל מברשת | גודל המברשת הנבחר על ידי המשתמש. | שינוי גודל המברשת לגודל הנבחר. | מימוש בחירת המשתמש. | אין. |
| **4.3** | בחירת/שינוי סוג מברשת | סוג המברשת הנבחר על ידי המשתמש. | שינוי סוג המברשת לגודל הנבחר. | מימוש בחירת המשתמש. | אין. |
| **5** | המשך ציור | בקשת המשך ציור של המשתמש. | העברת המשתמש לקנבס נקי חדש. | קנבס חדש. | אין. |
| **6** | התנתקות מהמערכת | בקשת התנתקות מהמערכת של המשתמש. | ניתוק המשתמש מהמערכת. | סגירת הממשק הגרפי של המערכת. | אין. |

תיאור בסיס הנתונים במערכת

השרת יוצר, כותב וקורא ממאגר הנתונים.

שדות בסיס הנתונים

|  |  |
| --- | --- |
| טיפוס שדה | שם שדה |
| text | email |
| text | username |
| text | password |

הסבר השדות

email – כתובת האימייל של המשתמש. משמשת לאימות נוסף של פרטי המשתמש, לאחר שהקליד סיסמה. בעזרת כתובת האימייל המשתמש יכול לאפס סיסמה במידת הצורך. שדה ייחודי.

username – שם המשתמש של המשתמש, בעזרתו המשתמש מתחבר למערכת. משמש לזיהוי מהיר של המשתמש במהלך הפעילות. שדה ייחודי.

password – הסיסמה של המשתמש, בעזרתה המשתמש מתחבר למערכת. משמשת לאימות פרטי המשתמש.

מודולים

קבצי השרת:

Project\_Server.py – הקובץ העיקרי והראשי במחשב השרת. כולל את השרת, את פעולותיו ואת התקשורת שלו עם הלקוחות.

Run\_Server.py – קובץ המשמש להרצת השרת במערכת.

Sign\_Up\_Database.py – קובץ הנוצר על ידי sqlite3 והוא מאגר הנתונים המכיל את נתוני ההרשמה של הלקוחות למערכת.

Login\_Database.py – קובץ הנוצר על ידי sqlite3 והוא מאגר הנתונים המכיל את נתוני ההתחברות של הלקוחות למערכת.

Project\_Email\_Handle.py – קובץ המשמש לשליחת אימייל ללקוחות בעת ההתחברות למערכת.

קבצי הלקוח:

Project\_Client.py – הקובץ העיקרי והראשי במחשב הלקוח. כולל את הלקוח, את פעולותיו ואת התקשורת שלו עם השרת.

Run\_Client.py – קובץ המשמש להרצת הלקוח במערכת.

Project\_Screens.py – קובץ המשמש להפעלת המסך הראשי של המערכת, ולאחר מכן את מסכי ההתחברות/הרשמה ואת מסך הציור המשותף (הקנבס).

הגדרת תקשורת (RFC)

כל הפקודות המועברות בין השרת והלקוח במערכת הן מסוג string או מסוג list.

תיאור התקשורת בצד הלקוח

בכל צביעה על מסך הציור של הלקוח, הלקוח שולח לשרת רשימה של טאפלים, כאשר כל טאפל מכיל נתונים על פיקסל מסוים שנצבע על מסך הלקוח: הצבע בו נצבע הפיקסל, מיקום הפיקסל על המסך, גודל המברשת איתה נצבע הפיקסל, וסוג המברשת איתה נצבע הפיקסל. רשימה זו מועברת לשרת באמצעות מודול pickle.

בנוסף, הלקוח מקבל מהשרת מידע על פיקסלים שעליו לצבוע במסך הציור שלו, הזהה בצורתו לרשימה המועברת מהלקוח עצמו לשרת, בהתאם לצביעה של לקוחות אחרים במקביל. קבלת מידע זה נעשה בחלקים, באמצעות הפונקציה recv\_data\_in\_chunks.

תיאור התקשורת בצד של השרת

בכל צביעה של מסך הלקוח, השרת מקבל מהלקוח רשימה של טאפלים, כאשר כל טאפל מכיל נתונים על פיקסל מסוים שנצבע על מסך הלקוח: הצבע בו נצבע הפיקסל, מיקום הפיקסל על המסך, גודל המברשת איתה נצבע הפיקסל, וסוג המברשת איתה נצבע הפיקסל. קבלת הרשימה נעשית בחלקים, באמצעות הפונקציה recv\_data\_in\_chunks. את הרשימה הזו השרת מעביר לכל הלקוחות האחרים המחוברים אליו (פרט ללקוח ששלח את הרשימה), והוא עושה זאת באמצעות המודול pickle.

פעולות שרת-לקוח

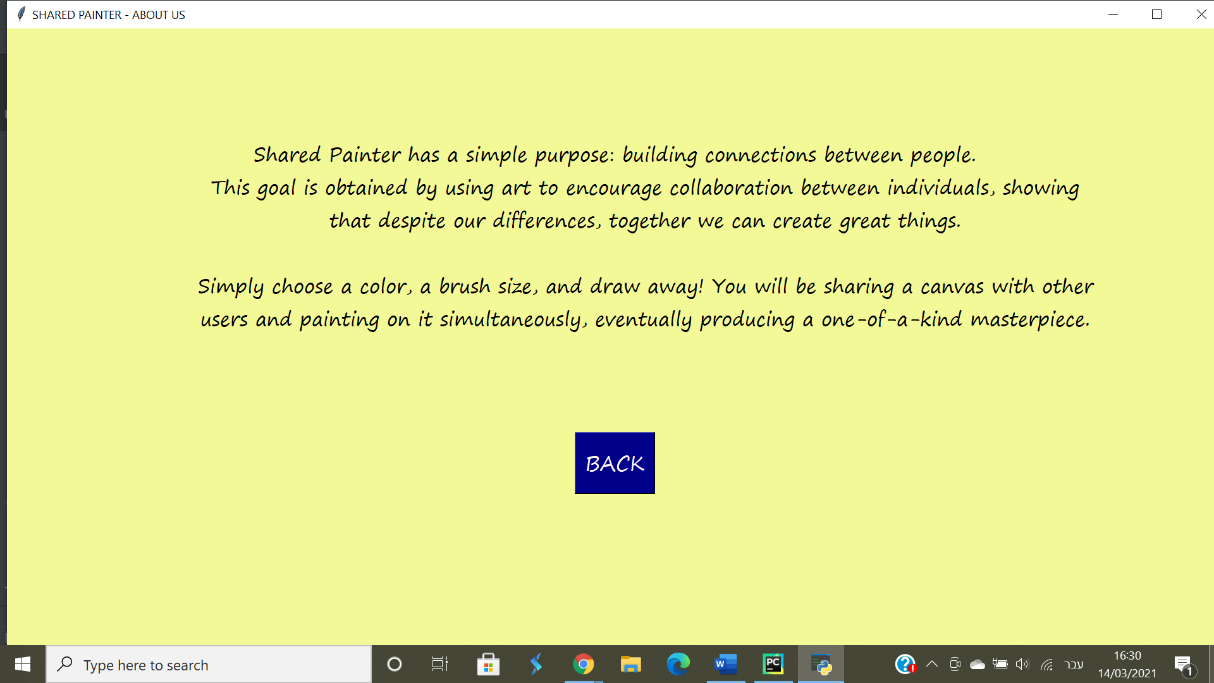
|  |  |  |
| --- | --- | --- |
| פעולה | פרמטרים | הסבר |
| send\_and\_recv\_data  (פעולה הנמצאת אצל הלקוח). | מחלקת הממשק הגרפי של הלקוח (מסכים), הסוקט של השרת. | הלקוח שולח מידע לשרת (פיקסלים שצבע הלקוח עצמו), ומקבל מידע מהשרת על פיקסלים שמשתמשים אחרים צבעו שעליו לצבוע. |
| recv\_data\_in\_chunks  (פעולה הנמצאת גם אצל הלקוח וגם אצל השרת). | אצל הלקוח: הסוקט של השרת, אצל השרת: הסוקט של הלקוח ששולח את המידע. | הלקוח/שרת מקבלים מידע בחלקים (הגודל המקסימלי של כל בלוק מידע הוא 4096 בייטים). |
| handle\_client\_connection  (פעולה הנמצאת אצל השרת). | הסוקט של הלקוח ששולח את המידע. | השרת מקבל מהלקוח מידע (הפיקסלים שצבע הלקוח עצמו) ומעביר אותו לשאר הלקוחות (כל הלקוחות המחוברים פרט ללקוח ששלח את המידע). |

צילומי מסך

מסך הבית

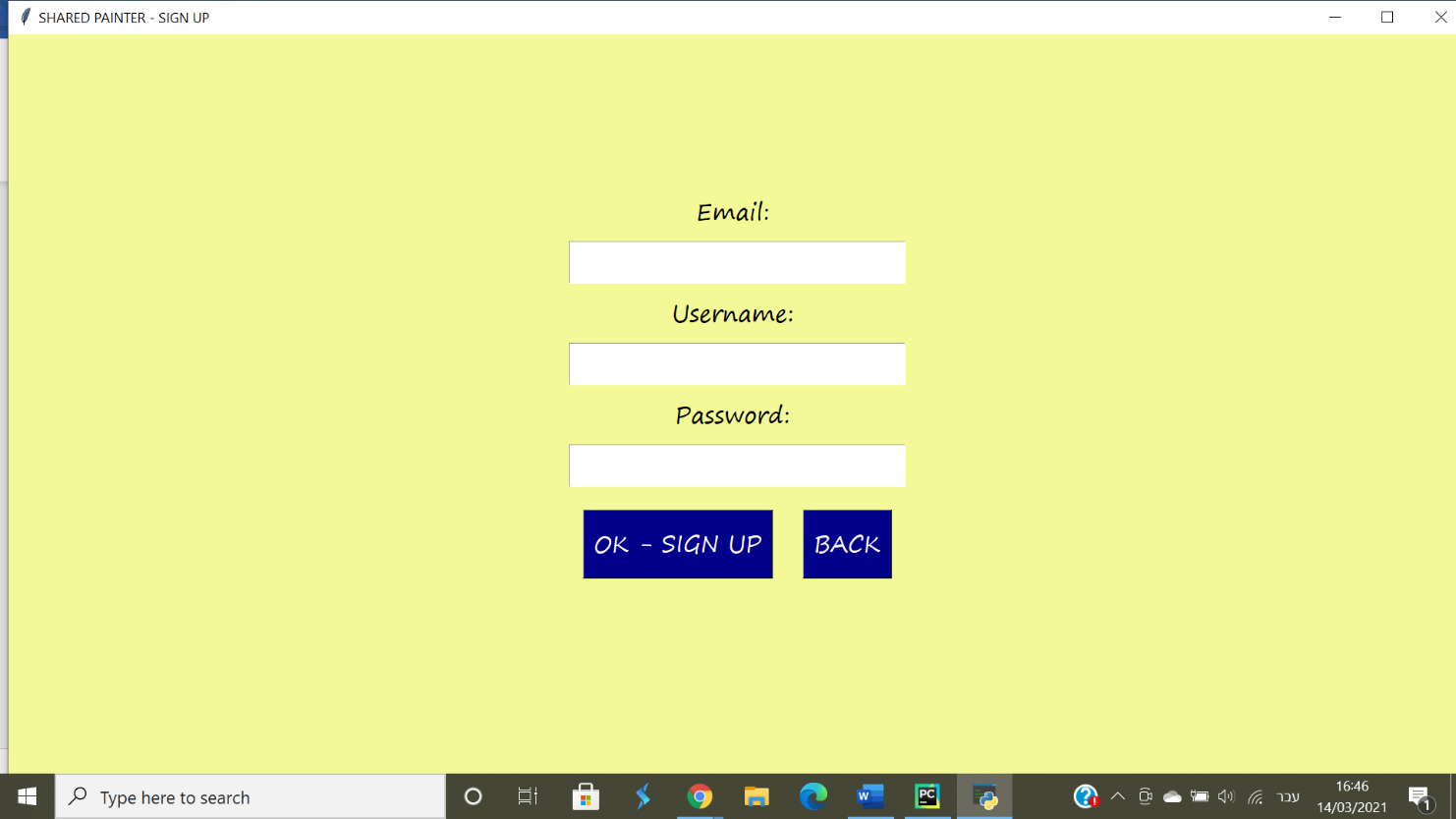
מסך הבית הוא מסך הפתיחה, כלומר המסך הראשון שהמשתמש רואה. ממסך זה ניתן להגיע למסך מידע אודות המערכת, מסך ההרשמה למערכת ומסך ההתחברות למערכת.

מסך מידע אודות המערכת

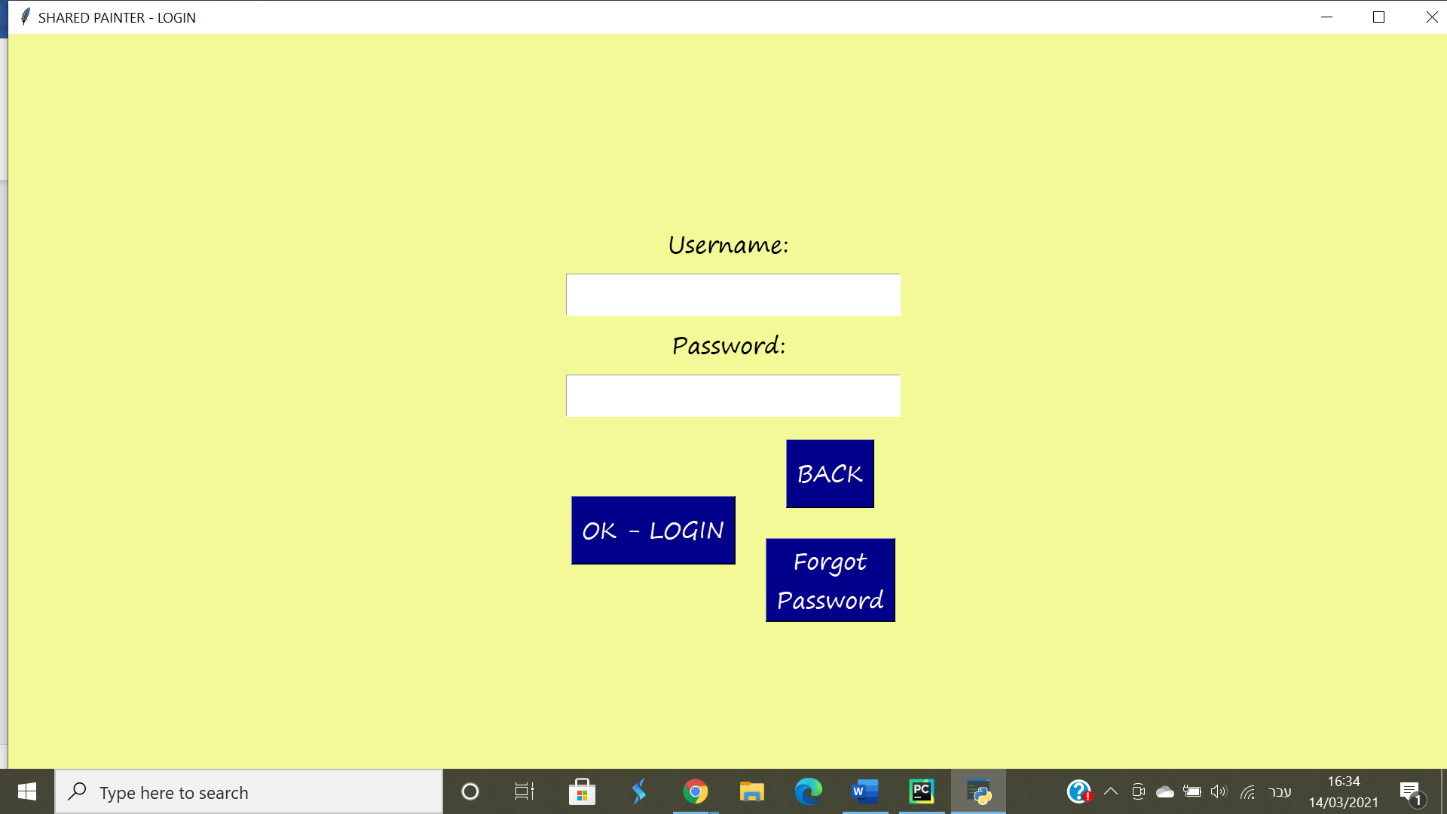
מסך זה מסביר את מטרות המערכת ואת אופן הפעולה בה.

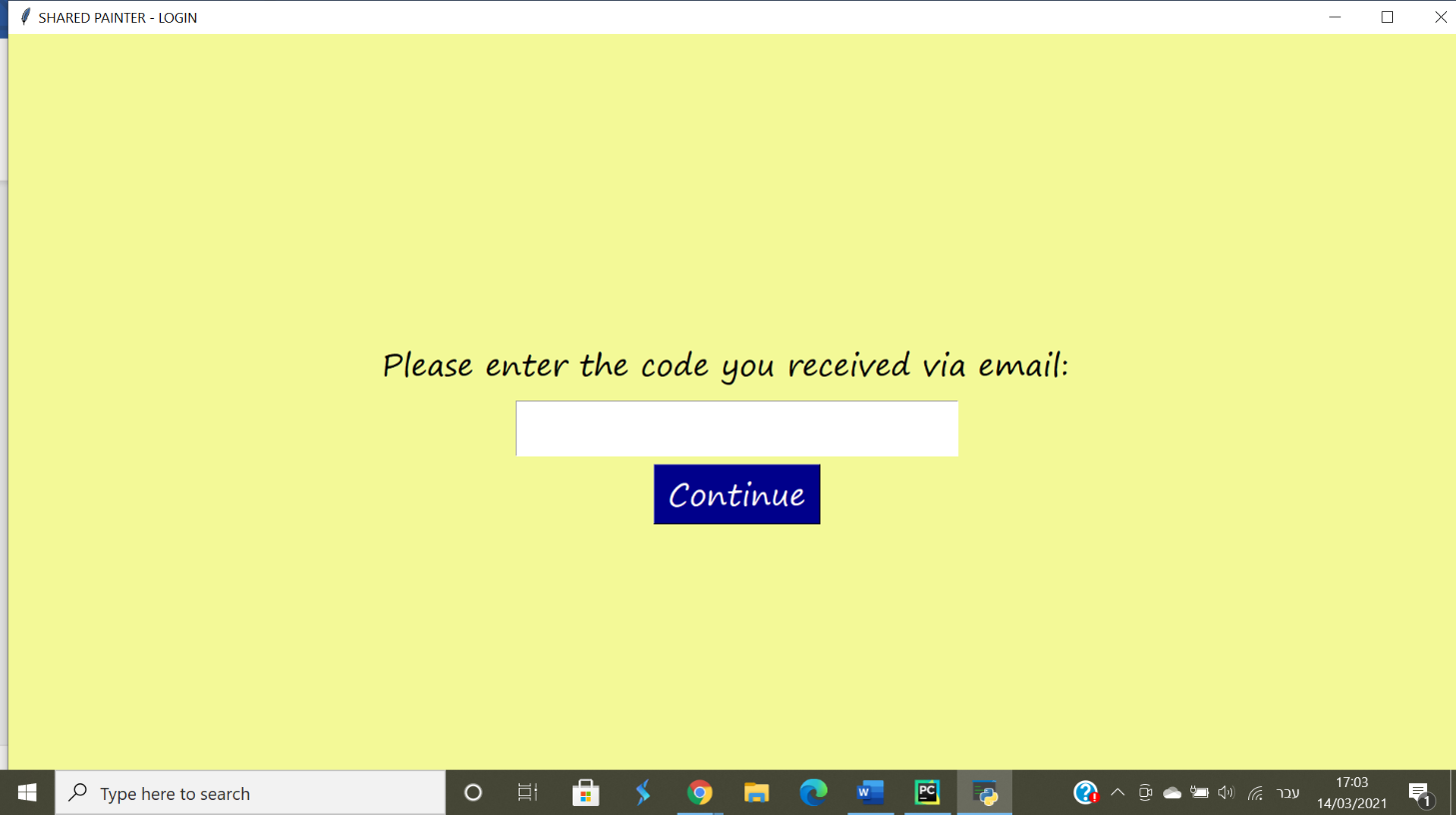
מסכי ההרשמה

במסך זה על המשתמש להכניס כתובת אימייל, שם משתמש וסיסמה על מנת להירשם למערכת, או לחזור אחורה למסך הבית.



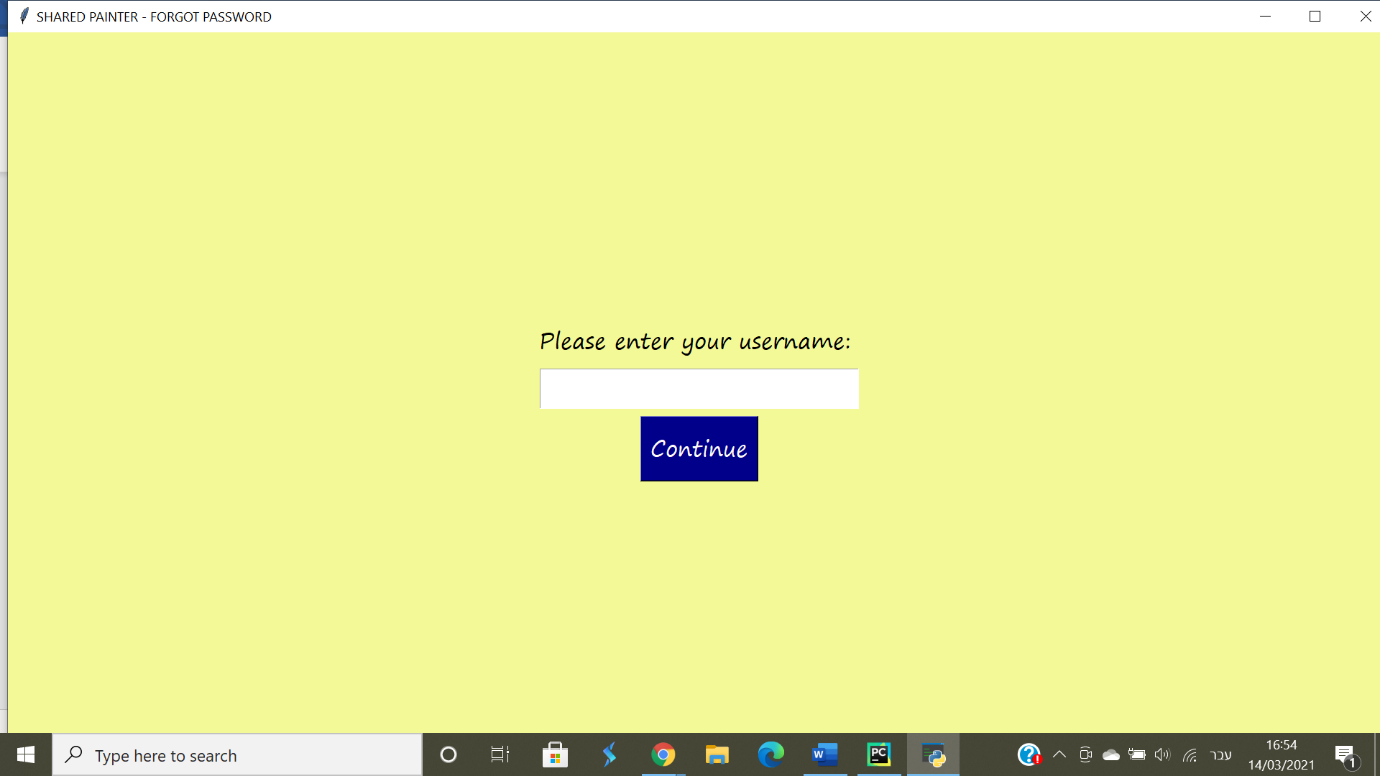
מסך ההתחברות

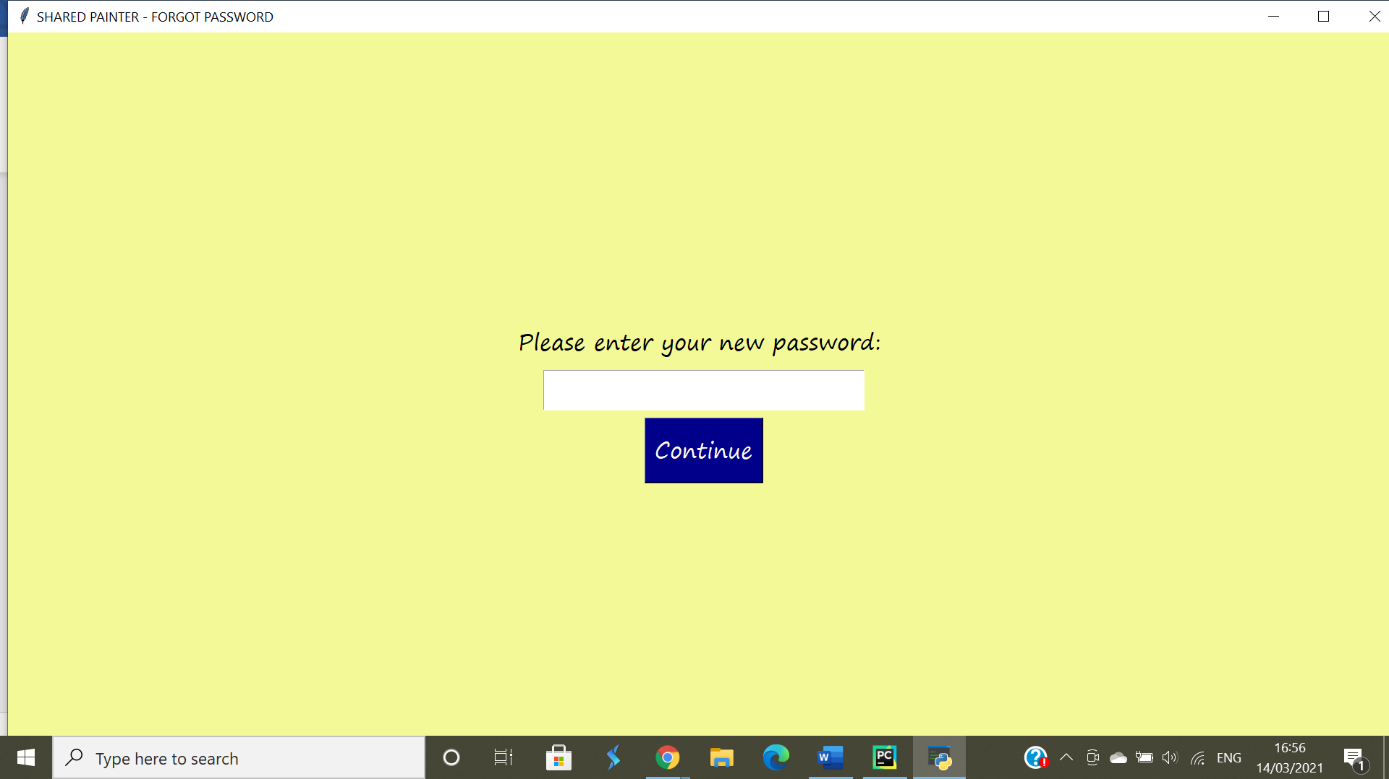
במסך העליון על המשתמש להכניס שם משתמש וסיסמה על מנת להתחבר למערכת. בנוסף, הוא יכול להגיע ממנו למסך איפוס הסיסמה או לחזור אחורה למסך הבית. המסך התחתון מופיע לאחר הכנסת הנתונים במקרה של התחברות למערכת, ובו המשתמש מתבקש להכניס קוד אימות התחברות שנשלח לכתובת האימייל שלו.



מסכי איפוס סיסמה

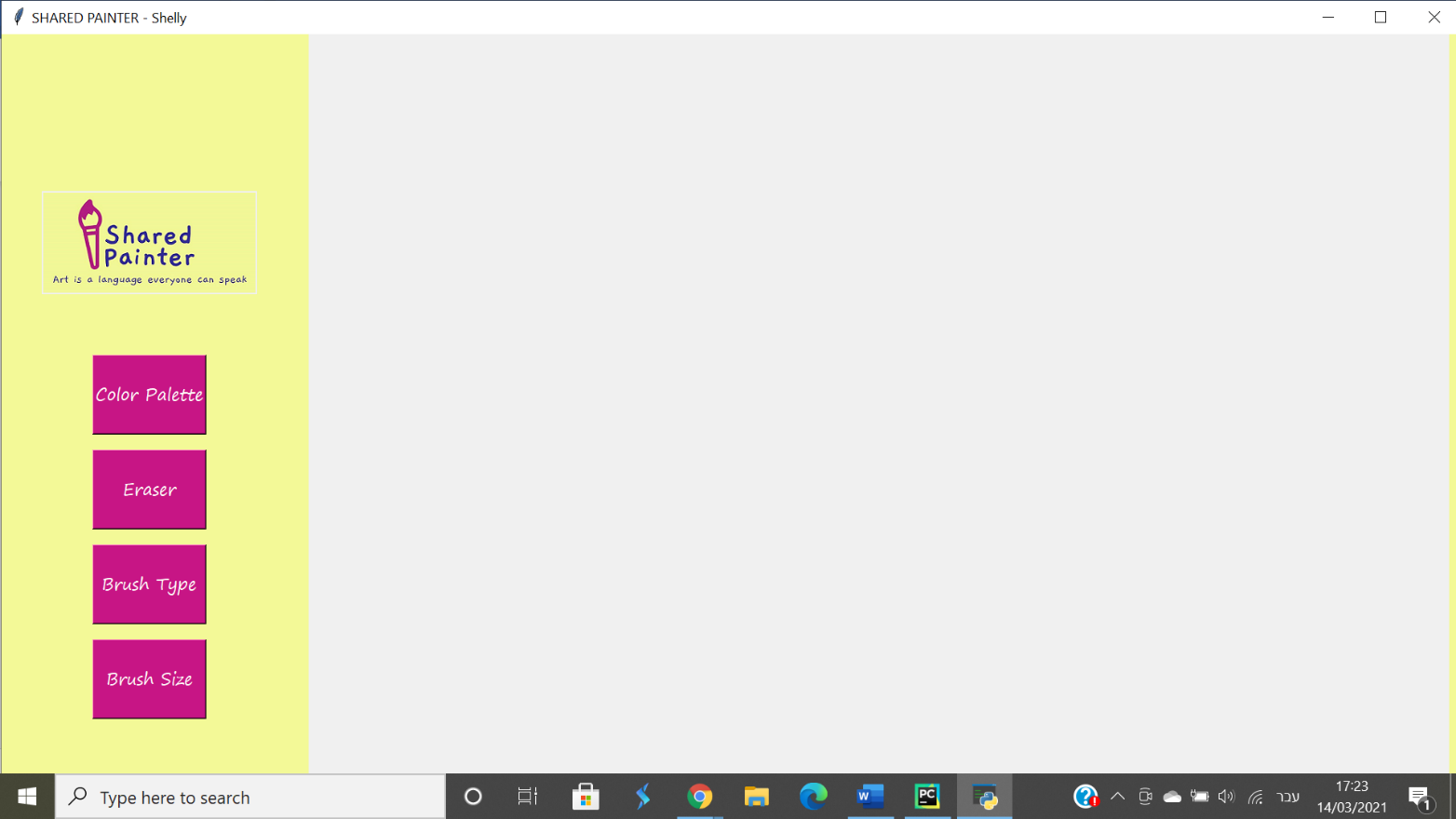
במסכים אלה המשתמש מקבל אפשרות לאפס את סיסמת כניסתו למערכת לסיסמה חדשה. במסך העליון המשתמש מתבקש להכניס את שם המשתמש איתו נרשם למערכת. במסך התחתון המשתמש מתבקש להכניס סיסמה חדשה, ולאחר לחיצה על כפתור "Continue" הוא מוחזר למסך ההרשמה למערכת, אותה יבצע עם סיסמתו החדשה.





מסך הקנבס

המשתמש מגיע למסך הקנבס לאחר שביצע בהצלחה תהליך של הרשמה/התחברות למערכת. זהו מסך הציור המשותף. במסך זה ניתן לבחור צבע לציור, סוג מברשת או מחק, וגודל מברשת.



תיעוד מחקר

|  |  |
| --- | --- |
| תאריך |  |
| תיאור הבעיה למחקר |  |
| מקור מידע רלוונטי |  |
| מסקנת המחקר |  |
| יישום בפרויקט |  |

אתגר – הצפנה

|  |  |
| --- | --- |
| תאריך |  |
| תיאור הבעיה למחקר | בחירת האלגוריתם למימוש ההצפנה. |
| מקור מידע רלוונטי | https://betterprogramming.pub/how-to-hash-in-python-8bf181806141 |
| מסקנת המחקר | לאחר מחקר על הצפנות שונות, גיליתי שהצפנת hash היא הצפנה חזקה ויעילה המאפשרת אבטחת מידע ברמה גבוהה, כמו גם פשוטה ליישום, לכן בחרתי להשתמש באלגוריתם ההצפנה הזה בפרויקט.  \* הצפנת hash נחשבת חזקה ומוגנת מאוד, מכיוון שלא ניתן לקרוא אותה או להפוך את ההצפנה (זהו הליך חד-כיווני). |
| יישום בפרויקט | כשהלקוח מכניס את פרטיו האישיים במקומות המתאימים בממשק הגרפי, הם נשלחים לשרת. השרת מצפין את הסיסמה שהכניס הלקוח ומכניס אותה מוצפנת למסד הנתונים להרשמה/התחברות. |

אתגר – מקביליות מערכת

|  |  |
| --- | --- |
| תאריך | 12.1.2021 |
| תיאור הבעיה למחקר | על השרת לטפל ולהעביר מידע למספר לקוחות במקביל, ועל הלקוח לצייר על מסך הקנבס שלו תוך צביעת פיקסלים בהתאם למידע שמתקבל מהשרת. |
| מקור מידע רלוונטי | https://realpython.com/intro-to-python-threading/ |
| מסקנת המחקר | לאחר מחקר על פעולות מקביליות של מערכת שרת-לקוח, גיליתי שהשימוש במודול threading יענה על צרכי המערכת שלי ויהיה יעיל ונוח לשימוש, לכן בחרתי להשתמש בו. |
| יישום בפרויקט | שימוש בthreading.Thread- בחיבור לקוחות לשרת וטיפול בהם בשרת, כמו גם בצביעת פיקסלים על מסך הלקוח, שקיבל מהשרת בהתאם לציורים של המשתמשים האחרים. |

רפלקציה אישית

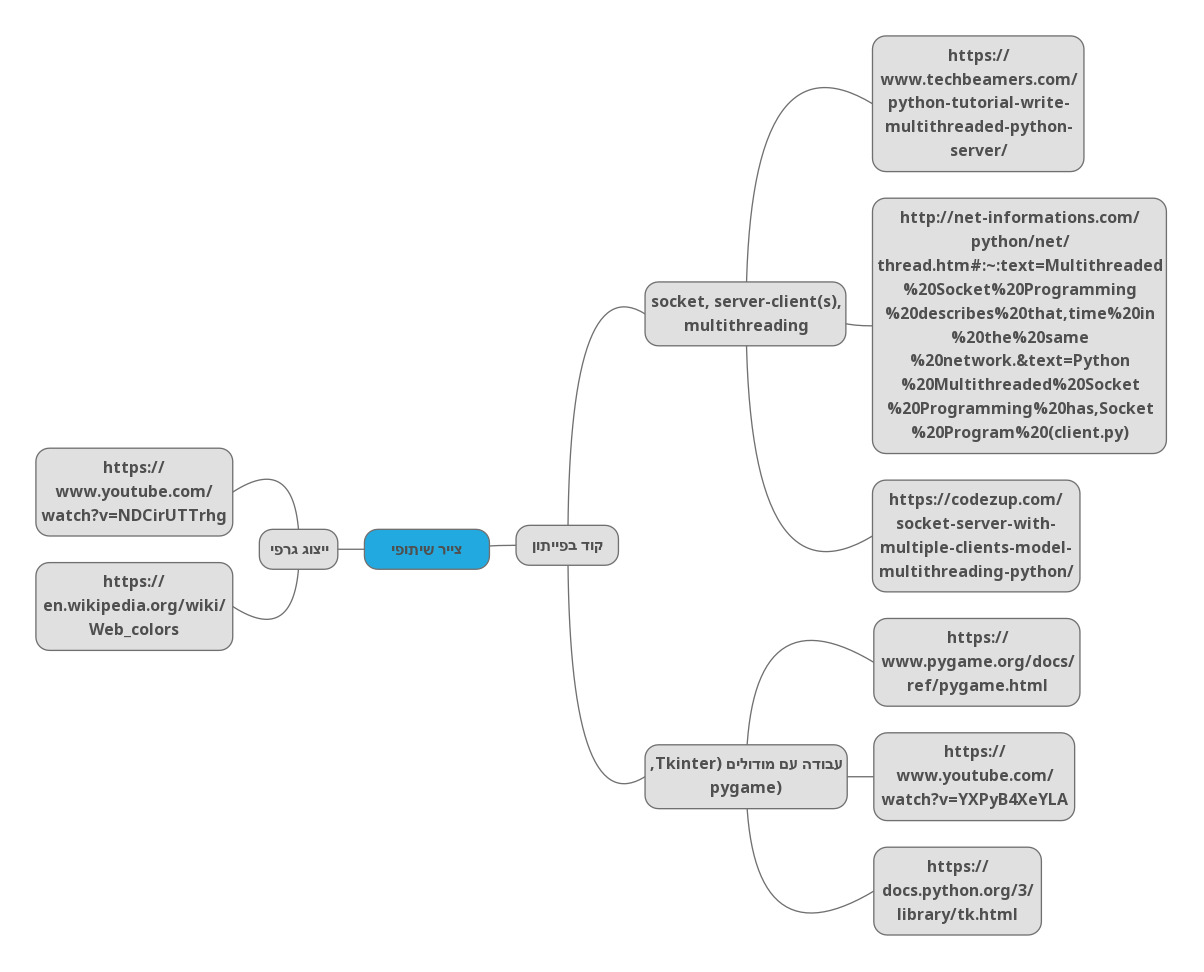
תודות

ובליוגרפיה (על פי כללי כתיבה אקדמית)

\* “Python Socket Receive Large Amount of Data”. Answered by JadedTuna July 2013. Stack Overflow.

<https://stackoverflow.com/questions/17667903/python-socket-receive-large-amount-of-data>

נספחים

\* מפת חשיבה: