



בית הילולת ניסויי ע"ש יצחק רבין א. מונר

מאכל הדקל 64. א. מונר 40600 טאפון: 09-2277400 פקס: 09-2277401

ControlNet

מערכת בקרה ושליטה במעבדת מחשבים

שם: יובל כהן.

ת.ז: 212778419.

שם הפרוייקט: ControlNet.

שם בית הספר: רבין תל מונד.

שם המנחה: ניר סליקטר.

שם החלופה: הנדסת תוכנה 883589.

תאריך הגשה: 1/6/2020.

קישור ל-github: <https://github.com/yuval130402/ControlNet>

קישור להרצת הפרוייקט: <https://youtu.be/us02j0-nnUk>





ד"ר יובל כהן, מנהל תחום חינוך

רחוב הרצל 64, תל אביב 40600 טלפון: 09-7777400 פקס: 09-7777401

תוכן עניינים

3	מבוא
3	רקע וסיבות לבחירת הנושא
3	תהליך המחקר
4	קהל היעד
5	ארכיטקטורה של הפרויקט
5	המוצר
5	פונקציונליות המערכת
7	מבני נתונים בהם נעשה שימוש
7	ארכיטקטורת רשת
8	דרישות המערכת + אילוצים עיקריים
9	סביבת הפיתוח
10	תרשימי זרימה
12	מדריך למשתמש
12	הוראות התקנה
12	מדריך הרצה
13	ממשק המשתמש
18	בסיס הנתונים
20	מדריך למפתח
20	קבצי המערכת
22	קבצים ומודלים עיקריים
26	טיפול בשגיאות
29	אלגוריתמים חשובים
39	רפלקציה
39	אתגרים בפרויקט
40	ביבליוגרפיה
41	נספחים
41	פיצ'ארים לשיפור (גרסאות פרויקט)



- 41הגדרות חשובות להסבר המערכת
- 42מודלים שימושיים בפרויקט

מבוא

תיק הפרוייקט מתאר את הפרוייקט שלי- **שליטה על מחשבים ובקרה עליהם ברשת מקומית (רשת LAN)**. הוא נועד לאפשר לקורא להבין מהו המוצר ומהי הפונקציונליות העיקרית שלו. בתיק פרויקט זה ניתן למצוא הסברים על האופן בו פועל המוצר, על מבנה הפרוייקט ותהליך פיתוח הפרוייקט. בנוסף, הוא כולל מדריך למשתמש כיצד להשתמש במוצר.

רקע וסיבות לבחירת הנושא

עקב התפתחות הטכנולוגיה כיום והצורך בשינוי שיטת הלימוד הקיימת לשיטת לימוד יותר יעילה, לפי דעתי יש להעביר לפחות חלק מהשיעורים לחדרי מחשבים בהם יוכלו התלמידים ללמוד בצורה יותר מהירה ויעילה. במהלך הלימודים שלי בחטיבה ובתיכון רק שיעורי סייבר ומדעי המחשב התקיימו במעבדת המחשבים, והיו פעמים מועטות שלמדנו מקצועות אחרים בשיטה זו. לתהליך למידה שבו לכל תלמיד יש מחשב ללמידה במקום מחברות ולוח כתיבה, יש המון יתרונות: הקלה על התלמידים ועל המורים (אינם צריכים לכתוב המון), קצב ההתקדמות הרבה יותר מהיר, למידה יותר נוחה לתלמידים רבים. אך כאשר לתלמידים מונח מחשב מול עיניהם זה עלול להוציא אותם מריכוז והם יכולים להתעסק בדברים אחרים כמו משחקים או סתם גלישה באינטרנט במהלך השיעורים, דבר העלול לפגוע בתהליך למידה כזה. לכן, בתחילת השנה כשחיפשתי רעיון לפרוייקט, חשבתי לפתח תוכנה עבור מורים/מנהלים שתשלט על מחשבי התלמידים/סטודנטים מהמחשב שלהם, ובכך תהווה מענה לבעיה זו. נוסף על כך, בחרתי בפרוייקט זה כיוון שרציתי לחקור על תחומים כמו רשתות ומערכות הפעלה שעניינו אותי ובעזרת מחקר על נושאים ספציפיים כמו שיתוף מסכים, נעילת עכבר ומקלדת של המחשב והדלקת מחשב מרחוק, להרחיב את הידע ולפתח מוצר שימושי. אני מאוד אוהב לתכנת ולחקור ומהרגע שבחרתי בפרוייקט זה המוטיבציה שלי לעבודה רק גברה.

תהליך המחקר

במהלך הפרוייקט היו לי כמה עבודות מחקר עיקריות. ראשית, חקרתי כיצד לשדר מסך בין מחשבים וכיצד לייעל את העברת המסך. מצאתי אפשרות לשידור מסך באמצעות ספריית mss שבאמצעותה צילום המסך נשמר כמשתנה, ובשביל שהעברת המסך תהיה מהירה יותר דחסתי את פקסלי התמונה למשתנה והשתמשתי בפרוטוקול UDP להעברת התמונה. שנית, עבודת מחקר נוספת הייתה כיצד לנעול את המחשב, כלומר איך לחסום את המחשב מקלט עכבר ומקלדת. חקרתי מגוון פתרונות ומצאתי ספריות וכלים שקיימים כבר הנועלים מקלדת ועכבר, אך הם לא הצליחו להתגבר על מנגנון `Ctrl+Alt+Del`. לכן הייתי צריך כלי עם יותר הרשאות כדי להיכנס לפרטים נמוכים יותר. לאחר חקירת הכלי `Device Manager` (מנהל התקנים), גיליתי שאפשר דרכו לשלוט על כל



ההתקנים שקיימים במחשב – וכך דרכו להשביית את התקן המקלדת/עכבר. כלי של Microsoft בשם *Devcon* מאפשר גישה אוטומטית למנהל ההתקנים עם פקודות *cmd* ובאמצעותו חסמתי את ההתקנים. מחקר נוסף שביצעתי במהלך הפרויקט היה את נושא הדלקת מחשב מרחוק. חקרתי את נושא *wake on lan*, ומצאתי כלי בשם *WolCmd* המאפשר הדלקה באמצעות פקודת *cmd* ממחשב אחר באותו ה-*lan* לפי כתובת *mac* של המחשב המודלק.

- סיקור מצב השוק כיום

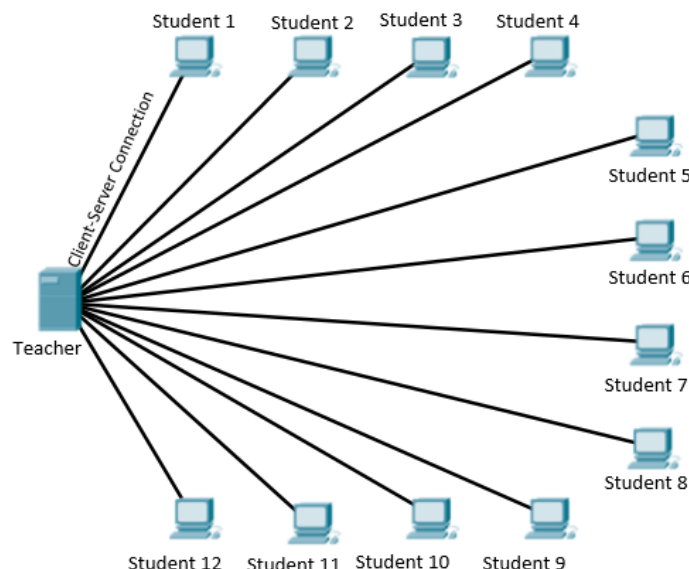
ישנן מספר תוכנות המאפשרות שליטה ובקרה מרחוק, המפורסמות מביניהן הן *VNC*, *NetSupport* ו-*Dameware* המשמשות בתי ספר ומקומות עבודה כאחד. תוכנות אלו מאפשרות מספר רב של פעולות, אך הרבה פעמים הן אינן לגמרי הכרחיות למורים והשימוש בהן לא בהכרח נוח.

- חידושים של הפרוייקט

התוכנה שלי מאפשרת את מרבית הפעולות השימושיות למורים/מנהלים, ועוד פעולות נוספות המונעות מתלמידים "לרמות את המערכת" (כגון כיבוי המחשב) וחוסמות מהם התנתקות מהמערכת, ושומרת על שימוש מהיר ונוח.

קהל היעד

המערכת מיועדת למורים/מרצים או לכל מנהל כיתת לימוד עם מעבדת מחשבים, המעוניינים במערכת שתאפשר בקרה, שליטה, וטיפול נוח במחשבים השונים בכיתה בזמן אמת, על מנת לאפשר מורה/מנהל להיות מודע למה שקורה במהלך השיעור ובמה התלמידים עסוקים במהלך השיעור. בזכות מערכת זו, המורה יוכל לשלוט במחשבי התלמידים ולפקח עליהם וכך ולהפוך את למידת התלמידים ליעילה ולאיכותית ככל הניתן.





ארכיטקטורה של הפרויקט

המוצר

- **שם המוצר: ControlNet.**
- **תיאור המוצר:** המערכת תהיה מותקנת במחשבי הכיתה, ומחשב אחד (של המורה) יהיה המנהל והשרת, וכל מחשב של תלמיד הוא לקוח. המערכת מאפשרת למורה לצפות במסכי התלמידים, לנעול להם את המחשבים, לכבות ולהדליק את המחשבים מרחוק, שידור מסך המורה אל מסכי התלמידים, העברת קבצים ועוד פיצ'ארים נוספים שיקלו על המורה לשלוט במחשבי התלמידים. המחשב של המורה הוא היחיד שיכול לעשות את פעולות אלו, ולמחשבי התלמידים אין יכולת זו.
- **מטרת המוצר ויתרונותיו:**
המטרה היא להעניק למורים מערכת שליטה ובקרה נוחה עם ממשק משתמש נוח לשימוש ולתפעול. המטרה הכללית של המערכת היא לייעל את דרך הלימוד שקיימת כיום בבתי ספר ובהרבה מקומות לימוד, לאור ההתפתחות הטכנולוגיה בתקופתנו ולאור הצורך בשינוי שיטת הלימוד הקיימת. לפי דעתי, שיטת הלימוד הקיימת בימינו פחות מקצועית ויעילה, בעקבות התפתחות הטכנולוגיה אני חושב שיש להעביר לפחות חלק משיעורי הלימוד מלמידה בכיתות לימוד (עם ספרים, מחברות ולוח למידה) ללמידה בחדרי מחשבים ואז ההתקדמות הלימודית תהיה מהירה יותר ואופן הלימוד יהיה יותר יעיל ונוח.

פונקציונליות המערכת

פעולות המורה/מנהל המערכת-

- צפייה במסך תלמיד מסוים שהוא בוחר מתוך התלמידים הקיימים במערכת - צילום מסך התלמיד הנבחר ע"י המורה והצגתו על מסך המורה.
- שידור מסך המורה - צילום מסך המורה והצגתו על מסך התלמיד/מסכי התלמידים הנבחרים ע"י המורה. במקרה זה מסכי התלמידים הנבחרים, המקלדת והעכבר ננעלים והם יכולים לצפות רק בשידור מסך המורה.
- עצירת שידור מסך המורה.
- נעילת העכבר, המקלדת ומסכי התלמידים - התעלמות מכל קלט התלמיד ע"י נעילת המקלדת והעכבר ונעילה של המסך שלו באמצעות ראיית תמונה אחת.
- שחרור נעילת העכבר, המקלדת ומסכי התלמידים.
- כיבוי מחשבים מרחוק - מחשב השרת יכול לשלוט על מחשבי התלמידים ולהחליט מתי שהוא רוצה לכבות ע"י לחיצה על הכפתור המתאים.



ראש הדקל 64, גא מור 40600 טלפון: 09-7777400 פקס: 09-7777401

- הדלקת מחשבים מרחוק.
- שליחת קובץ לתלמידים - העברת קובץ ממחשב המורה למחשב התלמיד או התלמידים הנבחרים.
- צפייה ברשימת המחשבים הקיימים במערכת - מוצג למורה בממשק המורה רשימת המחשבים המחוברים למערכת שעליה הוא שולט.
- אפשרות לבצע את הפעולות השונות על כל התלמיד ככתובת *broadcast*, ובנוסף אפשרות לבצע את הפעולות שונות על תלמיד או תלמידים ספציפיים שהמנהל בוחר.

פעולה	הסבר כללי על התהליך	קלט	פלט
1. שידור מסכי התלמידים למורה	צילום מסך התלמיד הנבחר ע"י המורה והצגתו על מסך המורה.	בהעברת המסך מעבירים 2 דברים: גודל התמונה של מסך התלמיד (אורך הפיקסלים).-תמונת מסך התלמיד (הפיקסלים עצמם).	הצגת מסך התלמיד על מחשב המורה.
2. שידור מסך המורה	צילום מסך המורה והצגתו על מסך התלמיד/מסכי התלמידים הנבחרים ע"י המורה. במקרה זה מסכי התלמידים הנבחרים, המקלדת והעכבר ננעלים והם יכולים לצפות רק בשידור מסך המורה.	בהעברת המסך מעבירים 2 דברים: גודל התמונה של מסך המורה (אורך הפיקסלים).-תמונת מסך המורה (הפיקסלים עצמם).	הצגת מסך המורה על מסכי התלמידים.
3. נעילת העכבר, המקלדת ומסכי התלמידים + שחרור נעילה זו	התעלמות מכל קלט התלמיד ע"י נעילת המקלדת והעכבר ונעילה של המסך שלו באמצעות ראיית תמונה אחת.	רשימה של כתובות IP של מחשבי התלמידים אותם נועלים או משחררים מנעילה.	מצב נעילה - הצגת חלון נעילה כמסך מלא על מסכי התלמידים והתעלמות מקלט התלמידים.
4. כיבוי והדלקת מחשבים מרחוק	מחשב השרת יכול לשלוט על מחשבי התלמידים ולהחליט מתי שהוא רוצה לכבות או להדליק אותם ע"י לחיצה על הכפתור המתאים.	רשימה של כתובות IP של מחשבי התלמידים אותם מכבים או מדליקים.	כיבוי או הדלקת המחשב.
5. שליחת קובץ לתלמידים	העברת קובץ ממחשב המורה למחשב התלמיד או התלמידים הנבחרים. שמירת הקובץ ב- <i>desktop</i> אצל מחשב התלמיד בתיקייה <i>ControlNet_files</i> .	הקובץ ורשימה של כתובות IP של מחשבי התלמידים אליהם שולחים את הקובץ.	הקובץ יופיע בתיקייה מסוימת במחשב אליו הוא מועבר.
6. צפייה ברשימת המחשבים הקיימים במערכת	מוצג למורה בממשק המורה רשימת המחשבים המחוברים למערכת שעליה הוא שולט.	אין.	אין.



פעולות שקורות באופן אוטומטי (פעולות רקע)-

- חיבור לשרת של כל לקוח המתחבר לתוכנה.
- שמירת נתוני התלמיד (כתובת IP, שם התלמיד, כתובת MAC) בבסיס הנתונים.
- יצירת התקשורת, מבוססת על חיבור UDP בין השרת לכל לקוח שמתחבר אליו.
- הפרדת תהליכים של תקשורת וממשק המשתמש.
- הדלקת ה-GUI ברקע בזמן אתחול התוכנה אצל השרת/אצל הלקוח.
- הפעלה אוטומטית של תוכנת הלקוח בהדלקת המחשב.
- השארת התוכנה דולקת ברגע שתלמיד מנסה לכבות אותה, הסרת חלון ה-console ונעילת ה-taskmgr כדי שהתלמיד לא יוכל להתנתק מהמערכת.
- הסתרת תיקיית הפרויקט והקבצים שבתוכה כדי שהתלמיד לא יוכל למחוק אותה מהמחשב.
- בדיקה אם התלמיד מחובר לאינטרנט – אם הוא ניתן את כבל הרשת, מחשבו ינעל עד להחזרת הכבל.
- התוכנה פועלת ברקע כך שגם אם התלמיד גולש באינטרנט, ברגע שהמורה נועל את מחשב התלמיד או משדר את המסך שלו, הפעולה מוצגת על מחשב התלמיד (כלומר ממשק הנעילה ושידור המסך נמצאים תמיד ב-topmost).

פעולות התלמיד-

- שליחת קובץ שנבחר ע"י התלמיד אל מחשב המורה. הקובץ נשמר ב-desktop אצל מחשב המורה בתיקייה על שם התלמיד השולח, שנמצאת בתוך תיקיית ControlNet_files.

מבני נתונים בהם נעשה שימוש

בפרוייקט שלי השתמשתי בכמה סוגי מבני נתונים :

- משתנה clients מטיפוס set – טיפוס זה דומה לרשימה, אך כל איבר בסט יכול להופיע פעם אחת בלבד וכך אם תלמיד מסוים מתחבר כמה פעמים יהיה רק מופע אחד שלו ברשימת clients.
- תור queue- עשיתי שימוש בתור בשביל התקשורת בין קבצי ה-gui המוצג למורה/לתלמיד לבין קובץ השרת/לקוח הראשי. יש 2 משתנים מטיפוס תור בקובץ משתנים גלובליים שנקראים ונכתבים מקבצים שונים.
- רשימה selected_clients המכילה את שמות התלמידים שהמורה סימן עליהם תבוצע הפעולה שבחר.

ארכיטקטורת רשת

התקשורת בפרוייקט שלי היא תקשורת של שרת מרובה משתתפים, כלומר שרת המתקשר עם מספר רב של לקוחות. פרוטוקול התקשורת בפרוייקט שלי הוא באמצעות מודל socket הקיים בשפת התכנות python. פרוטוקול התקשורת הראשי שפיתחתי מבוסס קישור UDP בשכבת התעבורה, IPv4 בשכבת הרשת. השרת מאזין להתחברות מחשבי הלקוחות הנמצאים ב-LAN. לצורך העברת מסך ממחשב

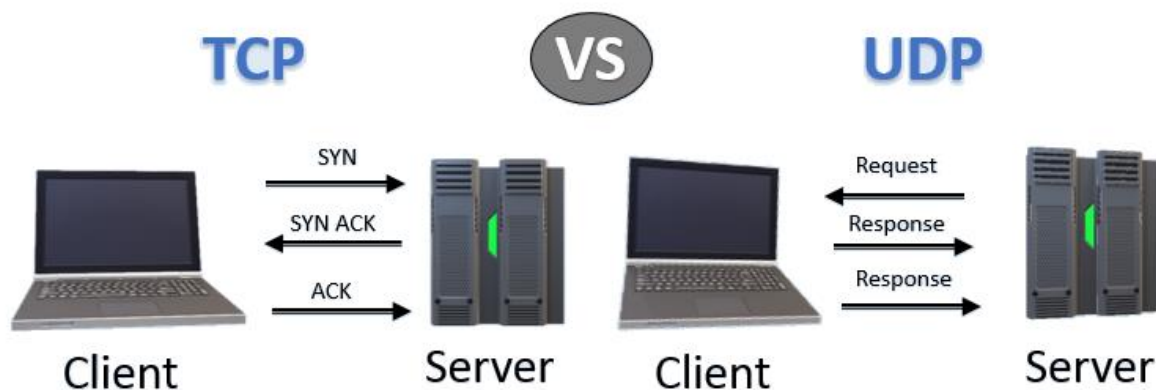


בית הלימודים לניהול מערכות מידע

ראש הדקל 64, גא. מנצ' 40600 טלפון: 09-7777400 פקס: 09-7777401

- למחשב אחר, הייתי צריך לחשוב על דרך יעילה ומהירה של העברת פקסלי כל תמונה כדי ששידור המסך יהיה מהיר ככל שניתן. לצורך כך, חקרתי על ההבדלים בין פרוטוקול UDP ל-TCP:
- **UDP** הוא פרוטוקול להעברת נתונים באופן לא אמין ללא הקמת קישור (*connectionless*), הוא לא מספק תהליך שבו הוא בודק אם המידע אמין או לא אלא מכיל בתוכו את המקור, היעד, אורך החבילה ו-CRC או *Checksum* לבדיקה שהחבילה מגיעה ללא שגיאות, אך אם היא מגיעה עם שגיאות הוא זורק את החבילה וממשיך הלאה. חבילות מידע עשויות להגיע בסדר שונה מזה שבו הן נשלחו, להגיע מספר פעמים או ללכת לאיבוד ולא להגיע כלל.
 - **TCP** הוא פרוטוקול המבטיח העברה אמינה של מידע בין 2 תחנות ברשת בסדר הנכון באמצעות יצירת חיבור מקושר (*Connection Oriented*). TCP מעביר את הנתונים שהועברו באמצעות IP, מוודא את נכונותם ושמירה על הסדר, ומאשר את קבלת הנתונים במלואם או מבקש שליחה מחדש של נתונים שלא הגיעו בצורה תקינה.

למרות אי אמינותו של פרוטוקול UDP, בחרתי להשתמש בו כי עצם היותו של UDP חסר-קישור זה גורם לו לספק דיוור מהיר לחבילות הנשלחות דרכו. המהירות היחסית של הפרוטוקול לעומת TCP הופכת אותו מתאים ביותר לאפליקציות שאינן דורשות אמינות מלאה של המידע. שידור מסך בהחלט אינו דורש אמינות ומהירות העברת המידע חשובה ביותר. בנוסף, הקמתי תקשורת משנית המבוססת על TCP לצורך העברת קבצים ממחשב למחשב. העברת קבצים דורשת אמינות ונדרש שהמידע יגיע בסדר הנכון ובצורה תקינה (גם במקרה של עיכוב), ולכן תקשורת זו מעל פרוטוקול TCP.



דרישות המערכת + אילוצים עיקריים

- המערכת דורשת חיבור לאינטרנט, המחשבים המתחברים למערכת חייבים להיות מחוברים לאותו ה-LAN (הרשת המקומית).

- ## סביבת הפיתוח

Pycharm - סביבת פיתוח הפרוייקט (כתיבת הקוד והרצה שלו).

Sqlite3 - יצירת מאגר הנתונים, *DB Browser* בשביל לצפות במאגר הנתונים.

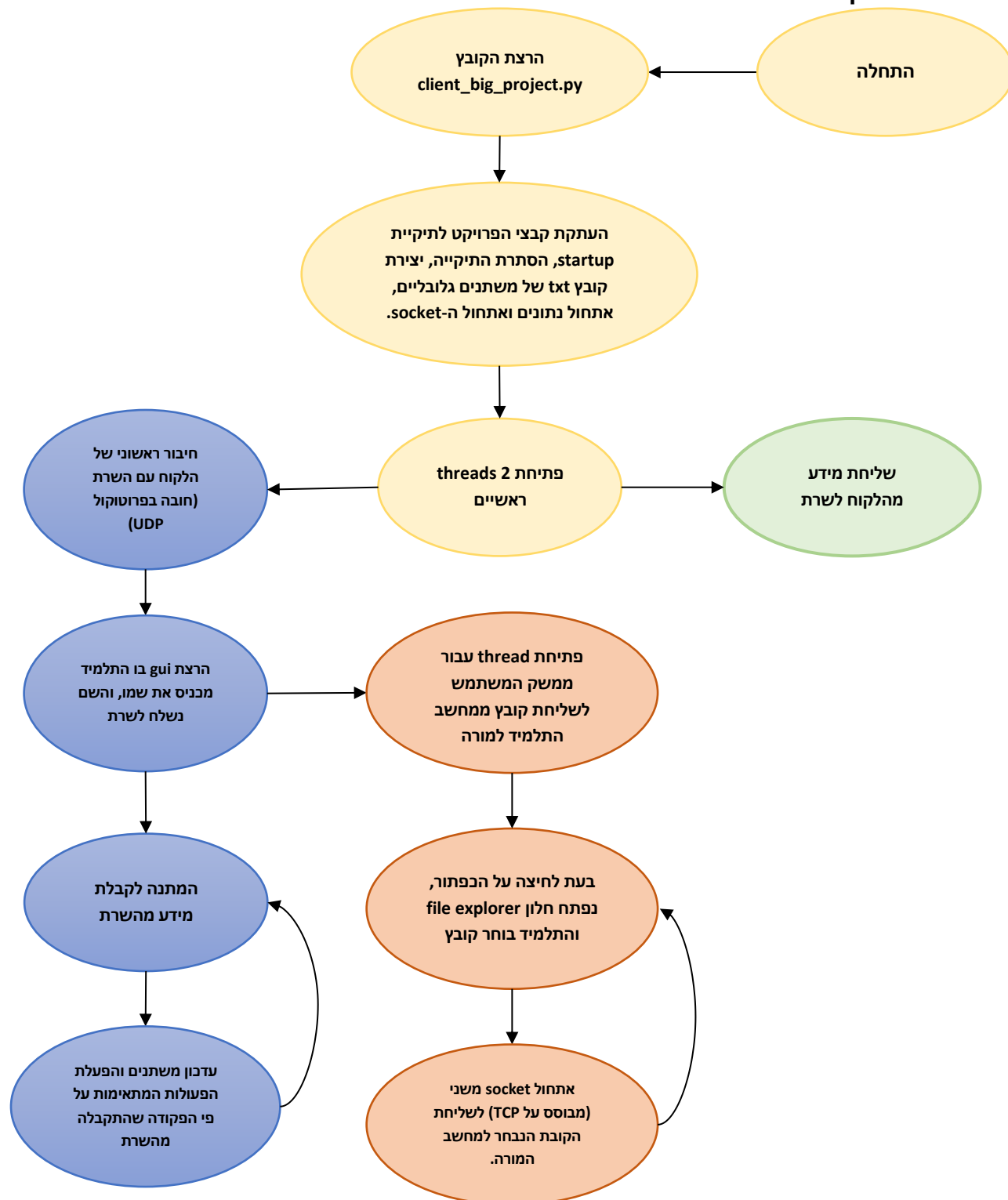
Page - ליצירת ממשק המשתמש באמצעות מודל Tkinter (ה-GUI המוצג למורה/מנהל המערכת).



בית הלימודים לניהול יזמים ורעיון אומץ

ראש הדקל 64, א' אומץ 40600 טלפון: 09-7777400 פקס: 09-7777401

אלגוריתם הלקוח:





מדריך למשתמש

הוראות התקנה

- עבור משתמש אשר רוצה להתקין את השירות, יש צורך לעקוב אחר ההוראות הבאות:
1. להתקין את תיקיית `ControlNet_Server` (כולל כל הקבצים שבתוכה) על מחשב המורה, ואת תיקיית `ControlNet_Client` על מחשבי התלמידים (התיקיות נמצאות בקישור לגיטהאב שצירפתי).
- ב-2 התיקיות הנ"ל קיים קובץ בשם `network_data.txt`. יש להכניס אצל התלמידים בקובץ זה במקום המתאים את כתובת ה-`ip` של כרטיס הרשת של מחשב המורה, אותה ניתן למצוא בעזרת הרצת הפקודה `ipconfig` ב-`cmd`. יש לבדוק שהפורטים ב-2 הקבצים מתואמים.
- לאחר ההתקנה, בשביל שהדלקת מחשבי התלמידים מרחוק תתאפשר יש לוודא במחשבי התלמידים שבהגדרות ה-`BIOS` (ראה הגדרה בנספחים) אפשרות `wake on lan` נמצאת במצב `enabled`. אפשרות `WOL` עובדת רק כאשר המחשב שאותו מדליקים מחובר לאינטרנט באופן קווי.

מדריך הרצה

יש להריץ במחשב המורה את קובץ ההרצה `server_big_project.exe` ובכל אחד ממחשבי התלמידים את קובץ ההרצה `client_big_project.exe` (יש לאפשר לקובץ זה בהרצה הראשונה הרשאות אדמין).
הערה – במחשבי התלמידים בזמן ריצת התוכנית הכניסה ל-`task manager` והתראת `UAC` חסומים כדי לא לאפשר לתלמיד להתנתק ולאפשר הרשאות אדמין, אך ברגע סיום התוכנית הכל חוזר לקדמותו.

המרת קבצי הפייתון לקובץ הרצה:

המרתי את קבצי ה-`python` (`.py`) של הפרויקט שנמצאים בתיקיות השרת והלקוח לקובץ הרצה (`.exe`).
יחיד המכיל את תכני כל הקבצים הקיימים בכל צד. באמצעות פעולה זו ניתן להריץ את התוכנה בלחיצה על הקובץ בלבד ללא צורך בהורדת `python` למחשב ובהורדת הספריות הנחוצות לפרויקט.
את קבצי ה-`exe` יצרתי באמצעות ספרייה ב-`python` בשם `pyinstaller` ע"י הרצת הפקודה הבאה בטרמינל:

בצד הלקוח:

```
# convert to exe file -
# pyinstaller --onefile --uac-admin client_big_project.py -r prog.exe.manifest,1
```

בצד הלקוח יש צורך בהרשאות אדמין על מנת להפעיל את תוכנת ה-`devcon`, לכן יצרתי את קובץ ה-`exe` עם הרשאות אדמין באמצעות שימוש ב-`flag: uac-admin`, וביטלתי את הודעת `UAC` לזמן שהפרויקט רץ כדי שכשמדליקים את המחשב התוכנה תעלה ללא צורך בלחיצה על אישור.









בצד השרת:

```
# convert to exe file -
# pyinstaller --onefile server_big_project.py
```

ממשק המשתמש

ממשק המורה

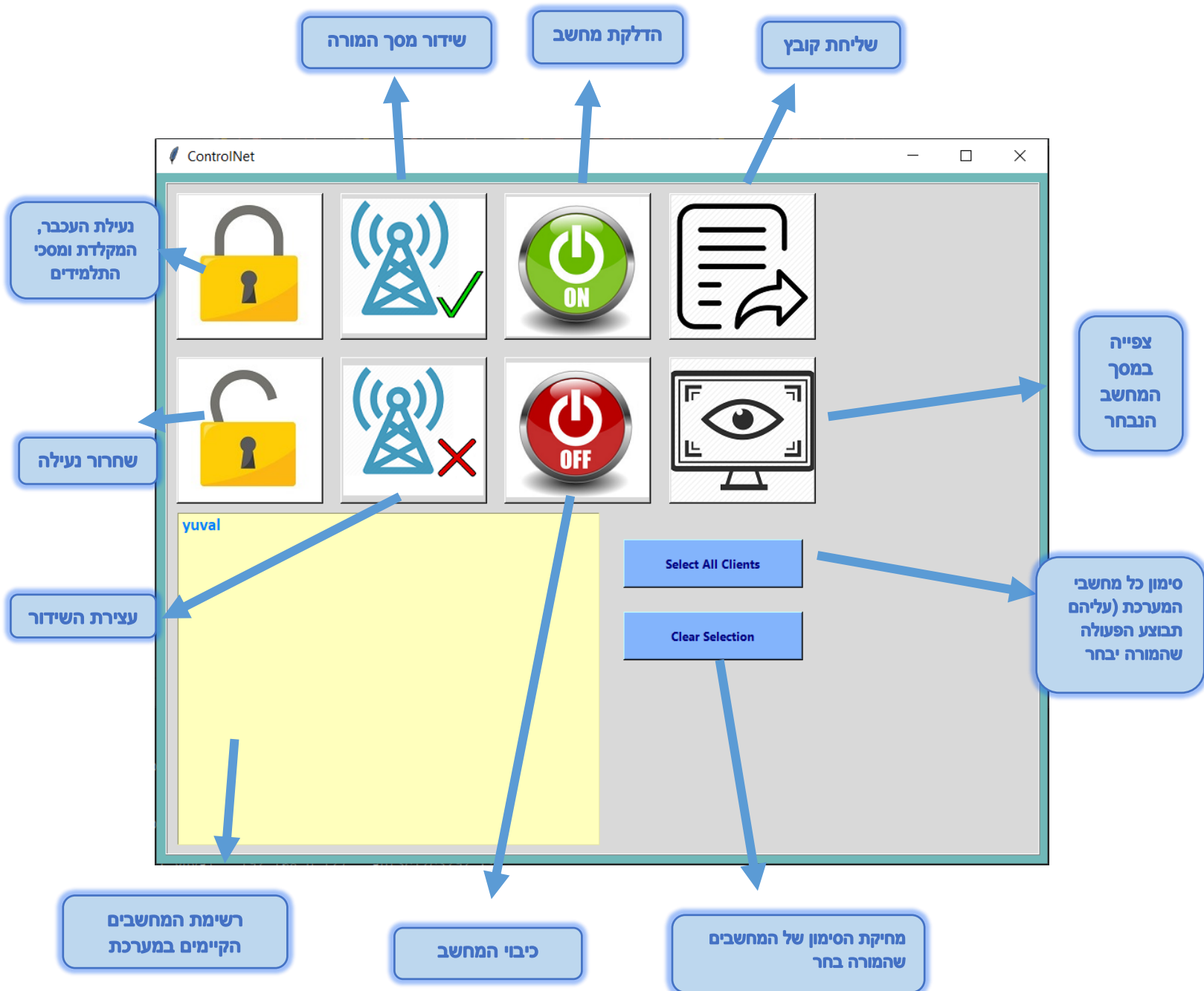
ממשק המשתמש המוצג למורה נבנה ע"י תוכנת העיצוב *PAGE* - זוהי תוכנת עיצוב של ממשקי משתמש, זה ה-*designer* של מודל *Tkinter* בשפת *python*. לאחר הרצת המערכת, מוצג ה-*GUI* למורה. המורה בוחר את התלמידים שרוצה שעליהם תפעל הפעולה ע"י לחיצה על שמם ולאחר מכן לוחץ על הכפתור שרוצה בשביל להפעיל את הפעולה. ממשק המשתמש המוצג למורה מקנה לו אפשרות נוחה לפקח על מחשבי התלמידים באמצעות הכפתורים הבאים:

פעולה	כפתור	פעולה	כפתור
שחרור הנעילה.		נעילת המחברים המסומנים.	
עצירת שידור המסך.		שידור מסך המורה למחשבים המסומנים.	
כיבוי המחשבים המסומנים.		הדלקת המחשבים המסומנים.	
צפייה במחשב התלמיד שסומן.		שליחת קובץ למחשבים המסומנים.	



בית הילול נוסווי ע"ס יצחק רבין גא מונר

ראוב הדקל 64, גא מונר 40600 טאפון: 09-7777400 פקס: 09-7777401



בנוסף, למחשב המורה מוצג חלון *pygame* כאשר הוא צופה במסך של אחד התלמידים.



ממשק משתמש המוצג לתלמיד

ממשק התחלה: כאשר התלמיד נכנס למערכת בפעם הראשונה או כשהתלמיד אינו מוזן בבסיס הנתונים של המורה, יופיע לו החלון הבא. על התלמיד להכניס את שמו/השם שייצג אותו במערכת. אם השם כבר קיים במערכת או שהשם לא תקין (שם תקין – אינו מכיל רווחים ובין 1-15 אותיות) אז תוצג הודעת שגיאה בהתאם. לאחר שהתלמיד מכניס את שמו עליו ללחוץ על מקש *enter* או על כפתור *get started*. המורה יכול להכניס מראש את שמות התלמידים בכל אחד מהמחשבים ואז כשהתלמידים מתחברים למחשב שלהם, שמותיהם כבר קיימים בבסיס הנתונים, התוכנה רצה אוטומטית והתלמידים לא צריכים להכניס את שמם (חלון המערכת מופיע על ההתחלה).

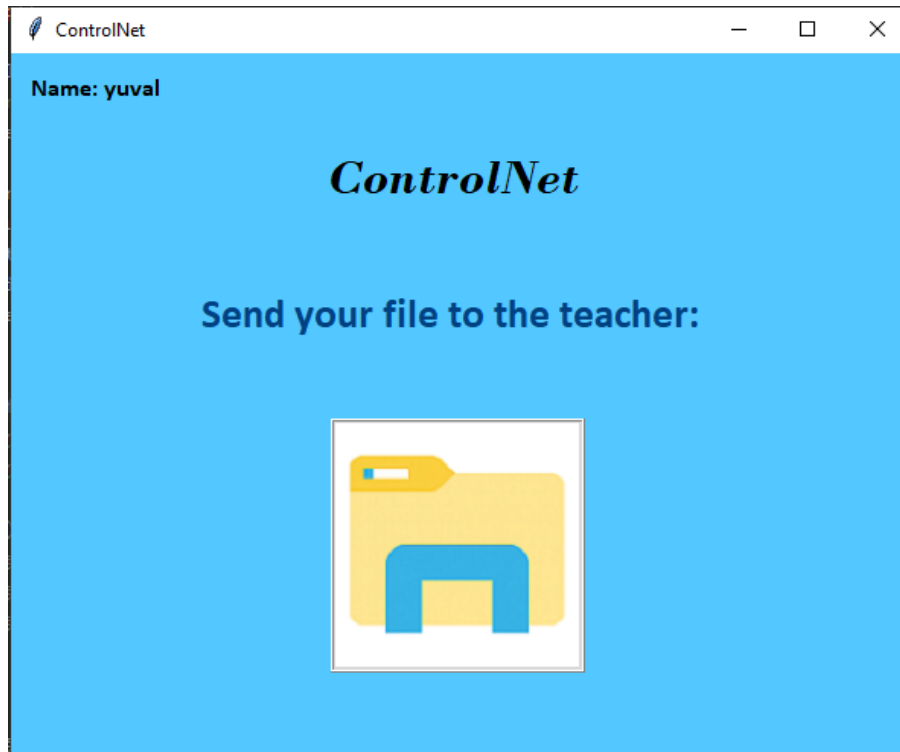
ממשק המערכת: לאחר שהתלמיד מחובר למערכת (הכניס את שמו), יופיע לו החלון הבא. בחלון זה לתלמיד יש אפשרות לשלוח קובץ למורה בכל רגע נתון (כשהוא לא נעול). כאשר התלמיד לוחץ על הכפתור שבמרכז יפתח לו חלון של *file explorer* והקובץ שהוא יבחר ישלח למורה.



בית הילולת נוסח"י יצחק רבין גן מנחם



רחוב הרצל 64, גן מנחם 40600 טלפון: 09-7777400 פקס: 09-7777401



ממשק נעילה: כאשר המורה נועל את מחשב התלמיד, לתלמיד מוצג חלון Tkinter עם הודעת נעילה על כל המסך ומחשבו ננעל (עכבר ומקלדת):

YOUR COMPUTER IS LOCKED



ממשק שידור מסך: למחשבי התלמידים מוצג חלון *pygame* המכיל את המסך של מחשב המורה כאשר הוא משדר מסך אליהם.



משרד החינוך, תל אביב 6100000, טלפון: 09-7777400, פקס: 09-7777401

בסיס הנתונים

כתבתי את בסיס הנתונים באמצעות *sqlite3*.
 במאגר הנתונים יישמרו נתונים על מחשבי התלמידים.
 השרת יוצר, כותב וקורא ממאגר נתונים במחשב המורה בשם *Clients.db*.
 רק לשרת יש גישה לבסיס הנתונים ויכולת לצפות בו.

שדות בסיס הנתונים:

שם השדה	טיפוס השדה	הסבר
Name	<i>String, primary key</i>	השם של מחשב התלמיד (מה שהתלמיד הגדיר). זהו שם התלמיד שהתלמיד מכניס בעת ההתחברות למערכת. שדה זה חובה (<i>NOT NULL</i>).
IP	<i>String</i>	כתובת ה-IP של מחשב התלמיד, משמש לתקשורת. שדה זה חובה (<i>NOT NULL</i>).
MAC	<i>String</i>	כתובת <i>mac address</i> של מחשב התלמיד, זוהי הכתובת הפיזית של ה- <i>network interface</i> המחובר. כתובת זו משמשת להדלקת המחשב מרחוק באמצעות טכנולוגיית <i>Wake On Lan</i> . שדה זה חובה (<i>NOT NULL</i>).



מבט-על *Clients.db* (באמצעות *DB Browser* בשביל לצפות במאגר הנתונים):

Name	Type	Schema
▼ Tables (1)		
▼ CLIENTS		CREATE TABLE CLIENTS(Name TEXT PRIMARY KEY NOT NULL , IP TEXT NOT NULL , MAC TEXT NOT NULL)
Name	TEXT	"Name" TEXT NOT NULL
IP	TEXT	"IP" TEXT NOT NULL
MAC	TEXT	"MAC" TEXT NOT NULL
Indices (0)		
Views (0)		
Triggers (0)		

דוגמא לבסיס נתונים המכיל נתונים:

Database Structure	Browse Data	Edit Pragmas	Execute SQL
Table: CLIENTS			
Name	IP	MAC	
Filter	Filter	Filter	
1 student1	192.168.0.127	a0f3c12c3292	
2 student2	192.168.0.112	a08cfdec6780	



מדריך למפתח

קבצי המערכת

קבצי המורה (נמצאים בתיקיית **ControlNet_Server**):

- **server_big_project.exe** – קובץ להרצת התוכנה אצל המורה.
- **gui_project.py** – קובץ האחראי על ממשק המשתמש המוצג למורה, נבנה באופן אוטומטי על ידי **PAGE** שם עיצבתי את ה-GUI.
- **gui_project_support.py** – הקובץ מכיל את כל הפונקציות, את הפעולות של ה-GUI (עדכון רשימת התלמידים, לחיצה על אחד הכפתורים וכו').
- **server_big_project.py** – הקובץ העיקרי והראשי במחשב המורה. כולל את השרת, את פעולותיו ואת התקשורת שלו עם הלקוחות.
- **server_globals.py** – קובץ המכיל את כל המשתנים הגלובליים, עבור התקשורת בין קובץ השרת לקבצי ה-GUI.
- **WakeOnLan.bat** – קובץ להרצת הדלקת מחשב מרחוק.
- **WolCmd.exe** – תוכנה בנויה שמצורפת עבור אפשרות הדלקת מחשב מרחוק.
- **Clients.db** – קובץ המכיל את בסיס הנתונים של התלמידים במערכת.
- **Images** – תיקיית תמונות.
- **network_data.txt** – קובץ המכיל את נתוני הרשת עליה רץ הפרויקט-כתובת ה-ip אליה מאזין השרת (default-0.0.0.0) והפורט של כל socket שקיים בפרויקט.

קבצי התלמיד (נמצאים בתיקיית **ControlNet_Client**):

- **client_big_project.exe** – קובץ להרצת התוכנה אצל התלמיד.
- **client_big_project.py** – הקובץ העיקרי והראשי במחשב התלמיד. כולל את הלקוח, את הפקודות, את התקשורת שלו עם השרת.
- **block_input.py** – קובץ לנעילת מחשב התלמיד (נעילה של כל קלט שהוא מקבל/ נעילה של התקן ספציפי במחשב).
- **finals.py** – קובץ המכיל משתנים גלובליים קבועים אצל הלקוח המשומשים בכמה קבצים.
- **project_variables.py** – קובץ היוצר בהרצה הראשונית של הפרויקט תיקייה בה נמצא הקובץ **vars.txt** המכיל משתנים גלובליים מעודכנים. בקובץ זה פעולות לקבלת ערכי המשתנים הנ"ל, החלפת ערכם ויצירת משתנים חדשים.
- **vars.txt** – קובץ עבור משתנים גלובליים שערכם האחרון נשמר גם כאשר התוכנה אינה פועלת. קובץ זה הכרחי עבור התגברות על ניסיון של התלמיד להתנתק מהמערכת באמצעות כיבוי המחשב. למשל, במידה והתלמיד מכבה את המחשב בזמן שהמחשב שלו ננעל באמצעות התוכנה שלי, אז בהדלקת המחשב מחדש המחשב ינעל שוב כיוון שבקובץ זה מעודכן משתנה הנעילה.



קובץ זה נמצא בתיקיית vars שנוצרת באתחול התוכנה ומוסרת מהמשתמש כדי שלא יהיה לו גישה לקובץ זה.

- **gui_client.py** – ממשק משתמש של התלמיד עבור הצטרפות למערכת.
- **gui_client_support.py** – פונקציונליות של gui_client.py.
- **send_gui.py** – ממשק משתמש של התלמיד עבור שליחת קובץ למורה.
- **send_gui_support.py** – פונקציונליות של send_gui.py.
- **fe_icon.png** – תמונה עבור ה-GUI.
- **Enable-WOLWindowsNICSettings.ps1** – קובץ הגדרות מתאימות להדלקת המחשב (WOL).
- **network_data.txt** - קובץ המכיל את נתוני הרשת עליה רץ הפרויקט-כתובת ה-ip של השרת והפורט של כל socket שקיים בפרויקט. יש להתאים את הפורטים לפורטים אצל קובץ זה הקיים גם בשרת.



קבצים ומודלים עיקריים

לכל המחלקות והפעולות העיקריות בכל קבצי הפרוייקט הוספתי **תיעוד בקוד** כך שאפשר לקרוא ולהבין מה הפעולה עושה. (בתרשימים המצורפים כאן כחול מסמל מחלקה, ורוד פונקציה וצהוב משתנה)

קובץ `server_big_project.py`

- v `_author_`
- v `clients`
- v `SERVER_IP`
- v `SERVER_PORT`
- v `SECONDARY_PORT`
- v `THIRD_PORT`
- v `TCP_PORT`
- v `TCP_PORT2`
- v `BUFFER_SIZE`
- v `MAX_BYTES`
- v `files_from_clients_path`
- > c `Clients`
- > c `sendfileThread(Thread)`
 - f `send_files()`
 - f `send_broadcast(sock, data)`
 - f `send_selected_clients(sock, cl, data)`
 - f `control_mss(selected_clients_list)`
- > f `mouse_listener()`
 - f `socket_recv(conn_socket, msgsize)`
 - f `recvall(length)`
 - f `recieve_screen(clients_list)`
 - f `get_subnet_mask(specific_ip)`
 - f `turn_on_computers(clients_list)`
 - f `selected_client_address(client_ip)`
 - f `selected_clients_from_their_names(selected_names)`
 - f `send_commands()`
- > c `getfileThread(Thread)`
 - f `get_file()`
 - f `create_vars_folder(path)`
- > c `Server(Thread)`
 - f `main()`

קובץ `client_big_project.py`

- v `_author_`
- v `SetWindowPos`
- v `NOSIZE`
- v `NOMOVE`
- v `TOPMOST`
- v `NOT_TOPMOST`
- v `prog_call`
- v `prog_location`
- v `WOL_SETTINGS_FILE`
- v `BROADCAST_IP`
- v `MAC_ADDRESS`
- v `SERVER_PORT`
- v `SECONDARY_PORT`
- v `THIRD_PORT`
- v `TCP_PORT`
- v `TCP_PORT2`
- v `BUFFER_SIZE`
- v `MAX_BYTES`
- f `add_to_startup()`
- f `control_mss()`
- f `control_mouse(data)`
- f `change_xy(x, y)`
- f `show_mouse()`
- f `exb()`
- f `waitingwindow()`
- f `get_file()`
- f `client_send()`
- f `set_taskmgr(status)`
- f `set_uac_message(status)`
- c `Client(Thread)`
- f `hide_folder()`
- f `define_wol_settings()`
- f `have_internet()`
- f `main()`



צד השרת:

Server מחלקת

המחלקה מייצגת את השרת.
 מכילה תכונות עיקריות של השרת, פעולות לניהול התקשורת
 עם הלקוחות והוספת לקוח חדש לרשימת הלקוחות.

```

C Server(Thread)
  m __init__(self, max_bytes)
  m appear_in_clients_list(self, new_address)
  m add_client(self, new_address, new_name="", new_mac="")
  m run(self)
  f HEIGHT
  f WIDTH
  f address
  f broadcast_ip
  f max_bytes
  f port
  f server_ip
  f server_socket
  
```

Clients מחלקת

המחלקה מייצגת את בסיס הנתונים של התלמידים במערכת.
 מכילה נתונים על מסד הנתונים, ופעולות כמו יצירת טבלת בסיס
 הנתונים (אם אינה קיימת כבר), הוספת תלמיד לבסיס הנתונים,
 מחיקת התלמידים מהמערכת ביציאה מהמערכת, בחירת
 תלמיד מסוים לפי שם/כתובת IP.

```

C Clients
  m __init__(self)
  m __str__(self)
  m get_table_name(self)
  m insert_client(self, name, ip, mac)
  m select_client_by_name(self, name)
  m return_client_by_name(self, name)
  m select_client_by_ip(self, ip_address)
  m return_client_by_ip(self, ip_address)
  m delete_clients(self)
  f __ip
  f __macaddress
  f __name
  f __tablename
  
```

common מחלקת

המחלקה מכילה את המשתנים הגלובליים אצל השרת, עבור
 התקשורת בין קובץ השרת לקבצי ה-GUI. המשתנים נקראים
 ונכתבים מקבצים שונים.

```

C common
  f conn_q
  f gui_q
  f picture_flag
  f selected_clients
  f sharing_screen
  
```



בית הילולת נוסו"י ע"ש יצחק רבין למו"ר

רחוב הרצל 64, למו"ר 40600 טלפון: 09-7777400 פקס: 09-7777401

מחלקת Toplevel1

המחלקה מייצגת את חלון ה-GUI המוצג למורה, נבנתה באופן אוטומטי ע"י תוכנת PAGE.

- Toplevel1
 - `__init__(self, top=None)`
 - `Clear_Selection`
 - `Clients_List`
 - `Frame_Features`
 - `Lock_Button`
 - `Select_All`
 - `SendFile_Button`
 - `Start_Sharing`
 - `Stop_Sharing`
 - `TurnOff_Button`
 - `TurnOn_Button`
 - `Unlock_button`
 - `WatchScreen_Button`
 - `style`

מחלקת getFileThread

המחלקה יורשת ממחלקת `thread`, ומייצגת לקוח ששולח קובץ למורה. ברגע שלקוח חדש מתחבר לשרת ה-TCP, נוצר אובייקט של המחלקה. האובייקט מכיל נתונים על אותו לקוח, ולאחר שנבנה האובייקט מופעלת הפעולה `run`-שמקבלת את הקובץ מאותו לקוח ושומרת אותו על המחשב בתיקייה המתאימה.

- getFileThread(Thread)
 - `__init__(self, ip, port, sock)`
 - `run(self)`
 - `ip`
 - `path`
 - `port`
 - `sock`

מחלקת sendfileThread

המחלקה יורשת ממחלקת `thread`, ומייצגת לקוח שמקבל קובץ מהמורה. ברגע שלקוח חדש מתחבר לשרת ה-TCP, נוצר אובייקט של המחלקה. האובייקט מכיל נתונים על אותו לקוח, ולאחר שנבנה האובייקט מופעלת הפעולה `run`-ששולחת את הקובץ ללקוח זה.

- sendfileThread(Thread)
 - `__init__(self, ip, port, sock)`
 - `run(self)`
 - `ip`
 - `port`
 - `sock`



צד הלקוח:

Finals מחלקת

מחלקה המכילה משתנים גלובליים אצל הלקוח המשומשים בכמה קבצים. המשתנים נקראים ונכתבים מקבצים שונים.

- Finals
 - SERVER_IP
 - USER_NAME
 - active_field
 - check_q
 - client_name
 - command_execute
 - computer_name
 - conn_q
 - end_gui
 - error_message
 - files_from_server_path
 - first_setup_path
 - height_screen
 - main_path
 - path
 - width_screen

Client מחלקת

המחלקה מייצגת את הלקוח. מכילה תכונות עיקריות של הלקוח, פעולות היתקשרות עם השרת, קבלת שידור מסך השרת ועוד.

- Client(Thread)
 - __init__(self, max_bytes)
 - socket_send(self, conn_socket, message)
 - socket_rcv(self, conn_socket, msgsize)
 - recvall(self, length)
 - recieve_screen(self)
 - uMad(self, event)
 - lock_screen(self, lock)
 - command_response(self, command)
 - run(self)
 - HEIGHT
 - WIDTH
 - client_socket
 - height
 - mac
 - max_bytes
 - mouse
 - port
 - server_ip
 - width



טיפול בשגיאות

חסימת האפשרות אצל התלמיד להתנתק מהמערכת:

בפיתוח מערכת של שליטת ובקרת מורה על מחשבי התלמידים עלי לא לאפשר לתלמיד להתנתק מהתוכנה כדי שהמורה ישלט עליו בכל רגע נתון. התלמיד יכול להתנתק מהתוכנה בכמה דרכים: למשל, בלחיצה על כפתור *close* בחלון ה-*console* של קובץ ההרצה/ בלחיצה על *end task* בתוכנת ה-*task manager* / ניתוק כבל הרשת והתנתקות מהאינטרנט/ מחיקת תיקיית הפרויקט וקבצי הפרויקט שעל המחשב/ ריסטרט למחשב. חסמתי בפני התלמיד את אפשרויות אלו באמצעות הרצה אוטומטית של הסקריפטים הבאים באתחול התוכנה:

- הסתרת חלון ה-*console* של קובץ ההרצה כדי שהתלמיד לא יוכל להתנתק:

```
# Hide the Console
window = win32console.GetConsoleWindow()
win32gui.ShowWindow(window, 0)
```

- נעילת ה-*taskmgr* באמצעות פקודה שמשנה ערך ברגיסטרי כדי שהתלמיד לא יוכל להתנתק:

```
def set_taskmgr(status):
    # enable/disable the task manager
    subprocess.Popen(
        "reg.exe add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System /v DisableTaskMgr /t REG_DWORD /d %s /f" % status)
```

- הסתרת תיקיות הפרויקט (תיקיית ה-*startup* ותיקיית *vars* שמכילה את קובץ המשתנים *vars.txt*) וכל הקבצים שקיימים בהן כדי שהתלמיד לא יוכל למחוק אותה:

```
def hide_folder():
    # hide the folder of the project.
    try:
        subprocess.Popen("attrib +s +h \"%s\" % final.first_setup_path)
        subprocess.Popen("attrib +s +h \"%s\" % final.main_path)
        subprocess.Popen("attrib +s +h \"%s\" % WOL_SETTINGS_FILE)
    except:
        pass
```



- באתחול התוכנה, העתקת קבצי התוכנה לתיקיית ה-*startup* כדי שאם התלמיד יבצע רסטרט למחשב התוכנה תרוץ שוב עם אתחול המחשב:

```
def add_to_startup():
    # copy the files of the project to the startup folder
    try:
        # file_folder = str(pathlib.Path(__file__).parent.absolute())
        file_folder = prog_location
        print(file_folder)
        devconMove = final.main_path + "\\devcon.exe"
        file_path_move = final.main_path + "\\client_big_project.exe"
        image_move = final.first_setup_path + "\\fe_icon.png"
        wol_settings_move = "C:\\%s" % WOL_SETTINGS_FILE
        network_data_move = final.first_setup_path + "\\network_data.txt"
        devconCurrent = file_folder + "\\devcon.exe"
        file_path_current = file_folder + "\\client_big_project.exe"
        image_current = file_folder + "\\fe_icon.png"
        wol_setting_current = file_folder + "\\%s" % WOL_SETTINGS_FILE
        network_data_current = file_folder + "\\network_data.txt"
        try:
            shutil.copy(image_current, image_move)
        except Exception as e:
            print(e)
        try:
            shutil.copy(file_path_current, file_path_move)
        except Exception as e:
            print(e)
        try:
            shutil.copy(devconCurrent, devconMove)
            shutil.copy(devconMove, r"C:\Windows\System32\devcon.exe")
            shutil.copy(devconMove, r"C:\Windows\SysWOW64\devcon.exe")
        except Exception as e:
            print(e)
        try:
            shutil.copy(wol_setting_current, wol_settings_move)
        except Exception as e:
            print(e)
        try:
            shutil.copy(network_data_current, network_data_move)
        except Exception as e:
            print(e)
    except:
        pass
```



- *Thread* הפועל במקביל לריצת התוכנה, בו מתבצעת בדיקה כל 3 שניות אם המחשב מחובר לאינטרנט במהלך ריצת התוכנה. במידה והתלמיד ניתק את החיבור לאינטרנט מחשבו ננעל.

```
def have_internet():
    # checks if the computer is connected to the network. if not - Lock the pc.
    condition = 0
    time.sleep(5)
    while final.end_gui is False:
        conn = httplib.HTTPConnection("www.google.com", timeout=5)
        try:
            conn.request("HEAD", "/")
            conn.close()
            if condition == 1:
                client.command_response("unlock_screen")
                condition = 0
        except:
            conn.close()
            if condition == 0:
                client.command_response("lock_screen")
                condition = 1
        time.sleep(3)
```

הפונקציות הנ"ל נמצאות בקובץ *client_big_project.py*.



ד"ר יובל כהן

ראש הדקל 64, גל מנר 40600 טלפון: 09-7777400 פקס: 09-7777401

אלגוריתמים חשובים

שיתוף מסך:

פונקציה בצד המקבל את המסך ומציג אותו בחלון pygame (נמצאת ב- server_big_project.py) ראשית, יצירת socket שאחראי רק לשידור המסך. קבלת גודל המסך השולח, יצירת חלון pygame והתאמתו לגודל המסך המקורי. במהלך הלולאה בכל סיבוב מקבלים את גודל התמונה של המסך ואת התמונה עצמה. את התמונה מוציאים מהדחיסה באמצעות פקודת decompress של ספריית zlib, מתאימים אותה לגודל מסך המחשב ומציגים על חלון pygame.

```
def receive_screen(clients_list):
    # the function gets an array of selected clients,
    # displays on the manager's screen the screen of the client that he selected.
    global watch_server_socket
    global start_watch_socket
    start_watch_socket = 1
    watch_server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    watch_server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
    watch_server_socket.bind((SERVER_IP, THIRD_PORT))
    data, watch_address = watch_server_socket.recvfrom(1024)
    print(data)
    print(watch_address)
    data = data.decode()
    width, height = data.split(",")
    width = int(width)
    height = int(height)
    # open the window
    pygame.init()
    screen = pygame.display.set_mode((manager.WIDTH, manager.HEIGHT))
    clock = pygame.time.Clock()
    watch_server_socket.settimeout(4)
    watching = True
    # print other computer screen
    try:
        while watching:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    print("close")
                    watching = False
                    pygame.quit()
                    send_selected_clients(manager.server_socket, clients_list, "watch_stop".encode())
                    # watch_server_socket.sendto("watch_stop".encode(), watch_address)
                    break
            try:
                size = int.from_bytes(socket.recv(watch_server_socket, MAX_BYTES), byteorder='big')
                print("1")
                while size > 100000000: # checks if it size and not part of the pixels
                    size = int.from_bytes(socket.recv(watch_server_socket, MAX_BYTES), byteorder='big')

                temp_pixels = recvall(size)
                pixels = decompress(temp_pixels)
                # Create the Surface from raw pixels
                img = pygame.image.fromstring(pixels, (width, height), 'RGB')
                picture = pygame.transform.scale(img, (manager.WIDTH, manager.HEIGHT))
                # Display the picture
                screen.blit(picture, (0, 0))
                pygame.display.flip()
                clock.tick(60)
            except:
                pass
        finally:
            print("11111")
            pygame.quit()
            watch_server_socket.close()
            pass
    except:
        pass
```



פונקציה בצד השולח מסך (נמצאת ב-client_big_project.py):

יצירת socket שאחראי רק לשידור המסך. בהתחלה שולחים פעם אחת את גודל המסך ולאחר מכן בלולאה באמצעות ספריית mss() שומרים במשתנה img צילום של המסך, ובאמצעות ספריית Zlib דוחסים את התמונה למשתנה pixels עבור העברתה. שולחים לצד השני בכל סיבוב 2 פרמטרים - גודל התמונה (אורך הפיקסלים) ואת התמונה עצמה כולה.

```
def control_mss():
    # the function photographs the student's screen and sends it to the teacher
    global watch_screen
    watch_screen = True
    global watch_client_socket
    watch_client_socket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
    watch_client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
    screen_size = "{}{}".format(client.WIDTH, client.HEIGHT)
    watch_client_socket.sendto(screen_size.encode(), (SERVER_IP, THIRD_PORT))
    with mss() as sct:
        rect = {'top': 0, 'left': 0, 'width': client.WIDTH, 'height': client.HEIGHT}
        while watch_screen:
            try:
                img = sct.grab(rect)
                pixels = compress(img.rgb, 6)
                size = len(pixels)
                size_len = (size.bit_length() + 7) // 8
                size_bytes = size.to_bytes(size_len, 'big')
                watch_client_socket.sendto(size_bytes, (SERVER_IP, THIRD_PORT))
                sleep = False
                if size > 200000:
                    sleep = True
                while client.max_bytes < len(pixels):
                    part_pixels = pixels[:client.max_bytes]
                    watch_client_socket.sendto(part_pixels, (SERVER_IP, THIRD_PORT))
                    if sleep:
                        time.sleep(0.001)
                    pixels = pixels[client.max_bytes:]
                watch_client_socket.sendto(pixels, (SERVER_IP, THIRD_PORT))
            except:
                pass
            time.sleep(0.01)
    print("end")
    watch_client_socket.close()
```



נעילת עכבר + טאצ'פד + מקלדת (הקוד נמצא בתיקיית התלמיד בקובץ `block_inpot.py`):
 נעילת המחשב מכל קבלת input באמצעות פקודות של הכלי `devcon`. ישנה אפשרות לנעילה של התקן ספציפי אחד או נעילת כל התקני הקלט לפי הפונקציות הבאות.

```
from subprocess import check_output
import subprocess
from project_variables import *
from finals import Finals as final

def parser(msg, keyword):
    """
    msg - send the message got from (devcon find *) into msg
    keyword - send the specific word to be searched
    return value is array with all the id's of the specific hardware
    """
    array = []
    msg = msg.decode()
    lines = str(msg).split("\n")
    for line in lines:
        if line.find(str(keyword)) != -1:
            good_part = line.split(":")
            good_part = good_part[0].strip()
            array.append(good_part)
    return array

def find_all():
    msg = subprocess.check_output("devcon find *", shell=False)
    return msg

def callDevcon(command, array):
    """
    function to call devcon
    command - what to operate
    array - the array of ids to be operated (can be NULL)
    """
    for obj in array:
        c = "devcon " + command + " @" + obj + "\"
        print(c)
        output = check_output(c, shell=False)
        print(output.decode())

def unlock():
    # unlock devices
    subprocess.Popen("devcon rescan")
```

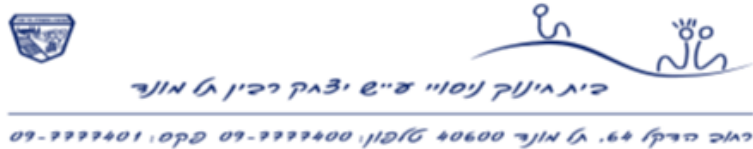


מִיִּשְׁרָאֵל מִסֻּבֵּי ע"ס יִצְחָק רֵבִין לַמּוֹצָי

גִּבְרָאֵל הַדָּק לַמּוֹצָי 64, לַמּוֹצָי 40600 טְלֶפֶן: 09-7777400 פֶּקֶד: 09-7777401

```
def lock_device(KeyWord):
    # Lock devices of the keyword
    try:
        while str(get(final.active_field)) == "1":
            devices = find_all()
            keyword_devices = parser(devices, KeyWord)
            if len(keyword_devices) != 0:
                callDevcon("remove", keyword_devices)
            unlock()
    except:
        unlock()

def lock_all():
    # Lock mouse, keyboard and touch pad of the computer.
    # subprocess.Popen("devcon remove usb*")
    try:
        while str(get(final.active_field)) == "1":
            devices = find_all()
            mouse_devices = parser(devices, "mouse")
            Mouse_devices = parser(devices, "Mouse")
            touch_pad_device = parser(devices, "pad")
            keyboard_devices = parser(devices, "Keyboard")
            # print("mouse")
            # print(mouse_devices)
            # print("Mouse")
            # print(Mouse_devices)
            # print("Keyboard")
            # print(keyboard_devices)
            try:
                callDevcon("remove", keyboard_devices)
            except:
                pass
            try:
                callDevcon("remove", mouse_devices)
            except:
                pass
            try:
                callDevcon("remove", touch_pad_device)
            except:
                pass
            try:
                callDevcon("remove", Mouse_devices)
            except:
                pass
        unlock()
    except:
        unlock()
```

```
def lock_settings():
    lock_option = str(get(final.command_execute))
    if lock_option == "lock_screen":
        print("all")
        if str(get(final.active_field)) == "1":
            lock_all()
        else:
            unlock()
    else:
        print("keyword")
        if str(get(final.active_field)) == "1":
            lock_device("Keyboard")
        else:
            unlock()

if __name__ == "__main__":
    lock_settings()
```

ממשק הנעילה (נמצא בתיקיית התלמיד בקובץ client_big_project.py):

מוצג לתלמיד בעת הנעילה חלון tkinter על מסך מלא, ללא אפשרות לצאת ממנו או להקטין אותו. החלון תמיד נמצא ב-topmost, כלומר הוא יוצג מעל כל המסכים שפתוחים במחשב ברקע. ברגע שמורה משחרר את הנעילה נסגר החלון.

```
def exb():
    pass

def waitingwindow():
    # create a window showing 'Lock computer'
    wa = tk.Tk()
    wa.title('ControlNet - YOUR COMPUTER IS LOCKED')
    wa.state('zoomed')
    wa.focus_set() # <-- move focus to this widget
    wa.protocol("WM_DELETE_WINDOW", exb) # hide close button
    wa.protocol("WM_MINIMIZE_WINDOW", exb) # hide minimize button
    x = wa.winfo_screenwidth()
    y = wa.winfo_screenheight()
    wa.geometry("%dx%d" % (x, y)) # full screen
    lb1 = tk.Label(wa, text="YOUR COMPUTER IS LOCKED\n", font=("Arial Bold", 70), pady=200, fg="RED")
    lb1.pack()
    wa.call('wm', 'attributes', '.', '-topmost', '1') # lift to the top
    wa.attributes("-topmost", True)
    wa.overrideredirect(1)
    # wa.lift()
    while str(get(final.active_field)) == "1":
        wa.update()
        time.sleep(0.1)
    print("destroy")
    wa.destroy()
```



שליחת קובץ (נמצא בתיקיית התלמיד בקובץ `send_gui_support.py`):

ברגע שתלמיד לוחץ על שליחת קובץ נפתח חלון `file explorer`, לאחר בחירת הקובץ נוצר ה-`client` והוא מתחבר לשרת TCP שתפקידו קבלת קבצים מלקוחות. לאחר החיבור לשרת מופעלת לולאה לשליחת הקובץ.

```
def send_files():
    tcp_client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    tcp_client.connect((final.SERVER_IP, client_big_project.TCP_PORT2))
    try:
        tcp_client.send(get(final.client_name).encode())
        print(filename)
        time.sleep(1.5)
        tcp_client.send(filename.encode())
        f = open(filename, 'rb')
        while True:
            l = f.read(client_big_project.BUFFER_SIZE)
            while (l):
                tcp_client.send(l)
                # print('Sent ', repr(l))
                l = f.read(client_big_project.BUFFER_SIZE)
            if not l:
                f.close()
                tcp_client.close()
                break
    except:
        pass

def upload_files(p1):
    file_explorer_root = tk.Tk()
    file_explorer_root.withdraw()
    global filename
    filename = filedialog.askopenfilename(filetypes=[("all files", "*")])
    if filename != "":
        send_files_thread = threading.Thread(target=send_files, args=())
        send_files_thread.start()
    file_explorer_root.destroy()
```



קבלת קובץ (נמצא בתיקיית המורה בקובץ server_big_project.py):

בתחילת הפונקציה `get_file()` נוצר שרת TCP המאזין לחיבורי לקוחות. ברגע שלקוח חדש מתחבר אליו, הוא יוצר עבור `thread` נפרד ששומר את נתוני הלקוח ומפעיל פונקצית קבלת קובץ מלקוח זה. בתחילת פונקציה זו, השרת מקבל את שם הקובץ ושומר אותו בתיקייה שהוא יוצר על שם התלמיד. לאחר מכן הוא מכניס לקובץ זה את התוכן שהוא מקבל.

```
class getfileThread(Thread):
    def __init__(self, ip, port, sock):
        Thread.__init__(self)
        self.ip = ip
        self.port = port
        self.sock = sock
        self.path = files_from_clients_path
        print(" New thread started for "+ip+":"+str(port))

    def run(self):
        client_name = self.sock.recv(BUFFER_SIZE).decode()
        print(client_name)
        file_path = self.sock.recv(BUFFER_SIZE).decode()
        print(file_path)
        file_name = os.path.split(file_path)[1]
        self.path = self.path + client_name + "\\ "
        create_folder(self.path)
        print(file_name)
        print(self.path)
        with open(self.path + file_name, 'wb') as f:
            print('file opened')
            while True:
                # print('receiving data...')
                data = self.sock.recv(BUFFER_SIZE)
                # print('data=%s', (data))
                if not data:
                    f.close()
                    print('file close()')
                    break
                # write data to a file
                f.write(data)
```



```
def get_file():
    global end_server
    end_server = False
    server_getfile_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_getfile_sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    server_getfile_sock.bind((SERVER_IP, TCP_PORT2))
    server_getfile_sock.settimeout(6)
    threads = []

    while True:
        try:
            server_getfile_sock.listen(50)
            print("Waiting for incoming connections...")
            (conn, (ip, port)) = server_getfile_sock.accept()
            print('Got connection from ', (ip, port))
            newthread = getfileThread(ip, port, conn)
            newthread.start()
            threads.append(newthread)
        except:
            if end_server is True:
                break
            pass

    for t in threads:
        t.join()
    print("getfile socket close")
    server_getfile_sock.close()

def create_folder(path):
    # define the name of the directory to be created
    try:
        os.mkdir(path)
    except OSError:
        print("Creation of the directory %s failed" % path)
    else:
        print("Successfully created the directory %s " % path)
```



הדלקת מחשב של תלמיד ממחשב המורה (בתיקיית המורה בקובץ `server_big_project.py`):
 יצירת קובץ batch ובתוכו פקודה עבור כל מחשב שמודלק. פקודה זו כוללת שימוש בכלי `wolcmd` ובנתוני המחשב של התלמיד, היא לפי המבנה הבא:
`wolcmd mac_sddress ip_address subnet_mask port`
 את כתובת ה-subnet mask של הרשת המקומית אני מגלה לפי הפונקציה הבאה.

```
def get_subnet_mask(specific_ip):
    # gets an ip address and returns the subnet mask of the internet adapter.
    msg = subprocess.check_output("ipconfig", shell=False)
    msg = msg.decode()
    adapters = str(msg).split("\r\n\r\n")
    for adp in adapters:
        if adp.find(str(specific_ip)) != -1:
            good_part = adp.split("\r\n")
            for line in good_part:
                if line.find("Subnet Mask") != -1:
                    sbm = line.split(":")[1].strip()
                    return sbm

def turn_on_computers(clients_list):
    # turn on the computers which in the list.
    turn_on_path = ".\\"
    with open(turn_on_path + "WakeOnLan.bat", "w+") as turn_on_file:
        for client_ip in clients_list:
            client_ip = client_ip[0]
            client_data = clients_table.return_client_by_ip(client_ip)
            client_mac = client_data[0][2]
            subnet_mask = get_subnet_mask(manager.server_local_ip)
            turn_on_file.write('wolcmd ' + client_mac + ' ' + client_ip + ' ' + subnet_mask + ' 7')
    os.system(turn_on_path + "WakeOnLan.bat")
```



דירקטוריון נוספים ע"ש יצחק רבין למו"ר

ראש הדקל 64, למו"ר 40600 טלפון: 09-7777400 פקס: 09-7777401

הוספת תלמיד למערכת (בתיקיית המורה בקובץ server_big_project.py):

בדיקה אם כתובת ה-IP נמצאת באחת הרשומות בבסיס הנתונים:
אם כן- התלמיד כבר מחובר למערכת. לכן עושים בדיקה אם שמו מופיע בממשק המורה ובמידה ולא מוסיפים את שמו לממשק המורה בהתאם.
אם לא- התלמיד אינו מחובר למערכת. לכן מבקשים ממנו להכניס את שמו. לאחר שהכניס את שמו, יש בדיקה אם השם קיים בבסיס הנתונים ואם כן המשתמש צריך להכניס שם אחר. אם השם לא קיים נתוני התלמיד נוספים לבסיס הנתונים ושמו מוכנס למשתנה gui_q מטיפוס תור שאחראי על הצגת שם התלמיד החדש על ממשק המורה.

```
def appear_in_clients_list(self, new_address):
    for a in clients:
        if str(a[0]) == str(new_address[0]):
            clients_discard(a)
            clients.add(new_address)
            return True
    return False

def add_client(self, new_address, new_name="", new_mac=""):
    # gets new address and name, insert the client to the DB if it possible.
    if clients_table.select_client_by_ip(str(new_address[0])) is False:
        if new_name == "":
            self.server_socket.sendto("enter a name".encode(), new_address)
        else:
            if clients_table.select_client_by_name(new_name) is False:
                clients.add(new_address)
                clients_table.insert_client(new_name, str(new_address[0]), new_mac)
                print("client was added")
                common.gui_q.put(" " + new_name)
                self.server_socket.sendto("welcome".encode(), new_address)
            else:
                self.server_socket.sendto("This name is exist, enter other name.".encode(), new_address)
    else:
        if self.appear_in_clients_list(new_address) is False:
            clients.add(new_address)
            print("client was added")
            client_data = clients_table.return_client_by_ip(new_address[0])
            common.gui_q.put(" " + client_data[0][0])
            self.server_socket.sendto("connected".encode(), new_address)
```



ד"ר יובל כהן

רחוב הרצל 64, תל אביב 6100000, טלפון: 09-7777400, פקס: 09-7777401

רפלקציה

נהנתי מאוד לפתח את הפרויקט ואני מרוצה מהתוצר. למדתי על עצמי שאני מאוד אוהב לתכנת, לפתח ולחקור נושאים רבים לעומק (למשל, בתוך רשתות). למדתי כיצד לחקור לעומק נושאים שונים שבכלל לא היה לי ידע בהם וכיצד לייעל את החיפוש שלי באינטרנט, דבר חשוב שישמש אותי בעתיד. למדתי לא להתייאש ולחקור לעומק כלים רבים – עד שאני מצליח. למדתי להתמודד עם בעיות המתרחשות במהלך העבודה על הפרויקט. כמו כן, למדתי איך לתכנן מערכת גדולה למשימות וחלקים קטנים יותר ובכך לעבוד באופן מסודר, יעיל ומהיר. נוסף על כך, גיליתי בעקבות המחקר מודלים חדשים, כלים כמו *devcon* ושיטות תכנות יעילות יותר, ולמדתי כיצד לכתוב תיק פרויקט. מסקנה חשובה שהסקתי במהלך העבודה היא חשיבות המחקר בפיתוח פרויקט רציני. מסקנה נוספת היא שחשוב לחלק את הקוד לפונקציות ולמחלקות כדי להקל על כתיבת הקוד וליצור סדר והיררכיה. אילו הייתי מתחיל היום את הפרויקט הייתי מקדיש יותר זמן לתכנון בסיס המערכת.

אתגרים בפרויקט

- הגדלת המהירות ושיפור האיכות של שיתוף המסך בין המחשבים. הפיתרון לכך – שימוש בספרייה *zlib* בשביל דחיסת תמונות המסך. כך מעבירים גודל קטן יותר ממחשב למחשב וזמן הערת התמונה קטן יותר. בנוסף, שימוש בפרטוקול *UDP* להגדלת המהירות.
- נעילת עכבר ומקלדת כך שלא יתאפשר להשתחרר מהנעילה באמצעות *control+alt+delete* ונעילת הפורטים של ה-*usb* (פיתרון לכך באמצעות הכלי *devcon*).
- טיפול במצב בו תלמיד מכבה את המחשב ומדליק מחדש. הפיתרון – הוספת קובץ ההרצה של הפרויקט לתיקיית ה-*startup* ויצירת קובץ *txt* למשתנים גלובליים (למשל, שומר ערך עבור מצב של נעילה, וכך כשהמחשב נדלק מחדש התוכנה מזהה כי על המחשב להיות נעול ולכן נועלת אותו שוב).
- שליחה של מידע שונה בו-זמנית ללקוח. למשל, כאשר המורה משדר את המסך לתלמידים הלקוח צריך לקבל גם את תמונות המסך וגם את מיקומי העכבר של המורה כדי שהעכבר יופיע אצלו באותו מיקום. הפיתרון היה ליצור *socket* נוסף שאחראי רק על פעולה מסוים (למשל-קבלת מיקומי העכבר) שירץ ב *thread* נפרד.



ביבליוגרפיה

- **Devcon -**
<https://docs.microsoft.com/en-us/windows-hardware/drivers/devtest/devcon>
<https://docs.microsoft.com/en-us/windows-hardware/drivers/devtest/devcon-general-commands>
- **General Questions -**
<https://stackoverflow.com/>
- **PAGE (Tkinter *designer*) -**
<http://page.sourceforge.net/html/index.html>
- **Tkinter -**
<https://docs.python.org/2/library/tkinter.html>
- **UDP Communication -**
<https://wiki.python.org/moin/UdpCommunication>
- **Wake On Lan -**
<https://www.youtube.com/watch?v=JMvE-ByZA-c>
<https://www.depicus.com/wake-on-lan/wake-on-lan-cmd>



משרד החינוך, תל אביב 40600 טלפון: 09-7777400 פקס: 09-7777401

נספחים

פיצ'ארים לשיפור (גרסאות פרוייקט)

גרסא סופית (ליום ההצגה)	עדכונים עתידיים
מערכת הכוללת נעילת עכבר+מקלדת, שידור מסך, צפייה במסך תלמיד, כיבוי והדלקת מחשב מרחוק, שליחת וקבלת קבצים.	צפייה במסך של כמה תלמידים בו-זמנית.
אפשרות לבצע את הפעולה הנבחרת על מחשבים ספציפיים / על כלל המחשבים.	הוספת צ'אט כך שהמורה יוכל לשלוח הודעות לתלמידים ולהפך.
טיפול בהתנתקות התלמיד מהאינטרנט, חסימת כיבוי התוכנה, הפעלה אוטומטית של הפרויקט עם הדלקת מחשב.	הוספת אפשרות לתלמיד לשתף את המסך לכולם.
המרת קבצי הפרויקט לקובץ הרצה (בצד הלקוח עם הרשאות אדמין).	

הגדרות חשובות להסבר המערכת

- **Python 3.7** - הגרסה של שפת התכנות *python* בה מימשתי את הפרוייקט.
- **Socket** - ממשק תוכנתי להעברת מידע בין תוכנות שונות. זהו *API* שמסופק בידי מערכת ההפעלה. *Socket* הוא נקודת קצה אחת של קשר דו-כיווני, של חיבור בין שני רכיבים ברשת.
- **Window 10** - מערכת הפעלה **מסדרת Windows** מבית מיקרוסופט למחשבים אישיים, מחשבי לוח, טלפונים ניידים וכו'.
- **Computer network** (רשת אינטרנט) - האינטרנט היא רשת תקשורת המאפשרת העברת נתונים והיא בעלת היקף כלל עולמי. הרשת נוצרה מחיבורים רבים בין רשתות מחשבים אשר איפשרו תקשורת בין מחשבים רבים ברשתות רבות.
- **LAN (Local Area Network)** - רשת מקומית. רשת מחשבים המתפרסת על אזור קטן ומוגבל, לרוב בתוך בניין או כיתה אחת.
- **Client** - מחשב לקוח המחובר למערכת.
- **Server** - מחשב השרת, שולט על הלקוחות ומספק להם שירותים או משאבים שונים.
- **Tkinter** - ספריית פייתון שבאמצעותה ניתן ליצור ממשק גרפי בפייתון.
- **PyCharm** - התוכנה בה פיתחתי את הפרוייקט, תוכנה לעבודה עם קבצי *python*.



ד.ג.ר.י.נ.ק. נ.י.ס.י.י. ע"ש י.צ.ק. ר.כ.י.ן ג. מ.נ.ר.

ד.ג.ר.י.נ.ק. 64, ג. מ.נ.ר. 40600 ט.ל.פ.ן: 09-7777400 פ.ק.ס: 09-7777401

- **PAGE** - התוכנה בה יצרתי את ממשק המשתמש, את ה-GUI של המערכת. זוהי תוכנת עיצוב של ממשק משתמש, זה ה-designer של Tkinter בשפת python.
- **SQLite3** - ספרייה בפייתון של בסיס נתונים עם שימוש ב-SQL, צפייה במאגר הנתונים באמצעות התוכנה *DB Browser (SQLite)*.
- **DevCon** - זהו כלי cmd המציג מידע מפורט על התקנים במחשבים שבהם פועל windows, בעזרתו ניתן לשלוט במנהל ההתקנים (*device manager*) בעזרת כתיבה ל-cmd. באמצעות devcon ניתן להפעיל, להשבית, להתקין, להגדיר ולהסיר התקנים של המחשב.
- **BIOS** - מנגנון של חומרה וקושחה (תוכנה הצרובה בחומרה) המשמש לאתחול המחשב, כמו גם להפעלת שגרות תוכנה אשר תומכות בקלט ובפלט.
- **Wake On Lan** - הפעלת מחשב מרחוק היא טכנולוגיה המאפשרת להפעיל או להעיר מחשב על ידי שליחת הודעת רשת. ההודעה נשלחת בדרך כלל על ידי תוכנית ממחשב באותה הרשת. עיקרון הפעולה: חיבורי רשת, פרוטוקולי רשת והאינטרנט עצמו מבוססים על חבילות מידע שנשלחות בין מחשבים. טכנולוגיית ה-WOL - מבוססת על חבילת מידע מיוחדת הנקראת "חבילת קסם" (*Magic Packet*) שנשלחת לכל המחשבים ברשת, ביניהם המחשב שרוצים להעיר. המחשב המוער אינו יודע האם הבקשה התקבלה מהרשת הפנימית או מהאינטרנט. חבילת הקסם תעבוד רק אם המחשב המוער תומך בפרוטוקול WOL בהגדרות ה-BIOS שלו.

מודלים שימושיים בפרויקט

חבילות/ספריות מיוחדות בסביבת פייתון שהשתמשתי בהן:

- **mss** – ספרייה לצילום ועיבוד תמונה. השתמשתי עבור צילום המסך בפונקציה שידור המסך.
- **zlib** – ספרייה לדחיסת תמונה. דחסתי את תמונת המסך בשביל לחסוך בזמן העברתה לצד המקבל, וכך שידור המסך מהיר יותר.
- **socket** – ספרייה לתקשורת בין רכיבי רשת, עליה התבססתי ביצירת התקשורת בפרויקט.
- **ctypes** – שימוש ב-winapi, השתמשתי בה למשל עבור קבלת גודל המסך הנתון.
- **time** - שימוש בזמן, השתמשתי בשביל הפסקות הכרחיות, למשל בתוך לולאות while.
- **pygame, tkinter** – ספריות GUI, השתמשתי עבור יצירת ה-GUI של המערכת, של שידור המסך ושל ממשק הנעילה המוצג לתלמיד.
- **threading** – שימוש בתהליכים הפועלים במקביל זה לזה. הכרחי מאוד בפרויקט שלי, ישנן הרבה threads מקביליים, למשל על מנת הרצת תוכנת השרת/לקוח במקביל ל-GUI.
- **pynput** – ספרייה לעבודה עם העכבר והמקלדת. השתמשתי בה על מנת להאזין למיקומי העכבר של המורה בעת שידור המסך שלו לתלמידים. במקביל לשליחת המסך גם ישנה שליחה של מיקומי העכבר לתלמידים וכך העכבר אצל התלמידים מצביע על מיקום עכבר המורה.
- **os** – ספרייה לעבודה עם מערכת ההפעלה, השתמשתי בה ליצירת תיקיות וקבצים ולכיבוי מחשב.



- **subprocess** – ספריה להרצת תהליכים, השתמשתי בה בשביל להריץ פקודות *cmd* שונות ולהריץ קבצים.
- **pathlib** – השתמשתי בה לקבלת כתובת ה-*path* של הספריה בה אני נמצא.
- **getmac** – השתמשתי בה בשביל למצוא את כתובת ה-*mac* של מחשב התלמיד.
- **shutil** – השתמשתי בספריה על מנת להעתיק את קבצי הפרויקט לתיקיית *startup* ולהעתיק את קובץ תוכנת *devcon* לספריות הנחוצות.
- **http.client** – השתמשתי עבור ביצוע בדיקות אם המחשב מחובר לאינטרנט.
- **win32console, win32gui** – השתמשתי על מנת להסתיר את חלון ה-*console*.
- **pyinstaller** – ספריה להמרת קבצי פייתון לקובץ הרצה *.exe*.
- **queue** – טיפוס נתונים תור, השתמשתי בתור על מנת העברת מידע בין קבצי ה-*GUI* והקובץ הראשי.