**עמוד השער**

תמונה

**יכלול:**

1. לוגו של בית הספר
2. לוגו של האפליקציה/יישום/מערכת
3. נושא העבודה
4. שם התלמיד
5. ת"ז של התלמיד
6. שם בית הספר ועיר
7. שמות המנחים + המורים המלווים
8. מועד הגשת המסמך (חודש ושנה)

**תוכן עניינים וראשי פרקים (אוטומטי לא ידני)**

**פרק א - nexus- ייזום**

1. **תיאור ראשוני של המערכת**

* תקציר כולל ורציונל הפרויקט - נקסוס היא רשת של לקוחות שמוכנים ומעוניינים לתקשר אחד עם השני בשביל האופציה להוריד קבצים אחד מהשני בשביל מהירות הורדה גבוהה יותר .
* מה המוצר המוגמר אמור לבצע – המוצר הסופי אמור לאפשר ללקוחות להוריד כל קובץ שקיים לפחות פעם אחת ברשת אצל לקוח אחר והורדה מהירה יותר של קבצים שקיימים יותר מפעם אחת ברשת.
* למה בחרתי בפרויקט ומה האתגרים שאני צופה לי בפרויקט – בחרתי בפרויקט משום שהתחברתי מאוד לרעיון עצמו ובנוסף לזה אני מרגיש שהפרויקט מאוד מתקשר לחיי היום יום שלי ואני חושב שפרויקט כזה יכול להיות שימושי להרבה מאוד אנשים. אחד האתגרים שצפויים לי בפרויקט הוא להגיע לשליטה עמוקה בפורמט של הקובץ ביטורנט. אתגר נוסף שיכול להיות לי הוא להבין איך לממש את הורדת הקבצים ברקע של התוכנית מבלי להפריע לפעילות הלקוח במחשב.

1. **הגדרת הלקוח**

* למי מיועדת המערכת ומי הולך להשתמש בה? – הפרויקט מיועד לכל לקוח שמעוניין לקצר את זמני ההורדה של הקבצים שלו, כגון לקוחות שצופים בסרטים ארוכים או לקוחות שמעוניינים להוריד קבצים חדשים שיצאו לאחרונה ואינם מעוניינים להוריד אותם מהמקור שלהם כדי לחסוך בזמן ההורדה.

1. **הגדרת יעדים/מטרות**

* מה המטרות המרכזיות של המערכת המוצעת (לפרט לכל לקוח ולקוח) – בראיית הלקוח.

מטרת כל לקוח היא לאפשר הורדת קבצים מהירה יותר ללקוחות אחרים ולשיפור מהירות הורדת הקבצים ברשת בנוסף לזה המטרה האישית של כל לקוח היא להוריד בעצמו קבצים יותר מהר. ומטרה נוספת שמאוד חשובה היא להבטיח ללקוחות שמורידים קבצים דרך המערכת דרכים לאמת את שלמות ומהימנות המידע המורד דרך המערכת.

1. **בעיות, תועלות וחסכונות**

* מה הבעיה/יות ומה אנחנו מנסים להשיג? – בכלליות הבעיה היא שהורדת קבצים היא לא מספיק אופטימלית ויכולה להיות יותר מהירה על ידי הורדה ממקומות שונים. אנחנו מנסים להשיג הורדת קבצים מהירה יותר ויעילה יותר
* מה התועלות שסביר לצפות מהמערכת? – בסיום הפרויקט קבצים יוכלו לרדת הרבה יותר מהר (במידה והקבצים קיימים כמה פעמיים ברשת) ובמידה ולקוח ירצה שיורידו ממנו קובץ חדש לדוגמא סרט שהוא הכין או כל תוצר אחר שהביקוש שלו יכול להיות גבוהה התוצר יאפשר הורדת עומס מאותו משתמש

* אלו שירותים המערכת תיתן? – המערכת תיתן שירות אחד שהוא הורדת קבצים שקיימים ברשת במהירות גבוהה מהרגיל תוך מתן ערובה למהימנות ושלמות הקבצים המורדים.
* השוואת העבודה עם פתרונות ויישומים קיימים– יישומים שקיימים ופותרים את הבעיה הם bit torrent network , internet Download Manager ההבדל הוא שאת הפרויקט אני מממש ולא משתמש בתוצר קיים. מה שמאפשר לי חופש עבודה והוספת יכולות שאני מעוניין להוסיף.

1. **האם צפויים קשיים או מגבלות בהגדרת המערכת**

* האם מדובר בטכנולוגיה חדשה ובלתי מוכרת? – לא מדובר בטכנולוגיה חדשה או מסוכנת אשתמש ברשת תקשורת, קבצים, אממש תהליכים של העלאה והורדה כולל אלמנטים של אבטחת מידע.
* האם קיימים סייגים בהגדרת המערכת?– למימוש אין דרישות של ציוד ספציפי בגלל המגבלה של רשת התקשורת הסגורה בבית הספר – רק משתמשים ברשת הזו יוכלו להשתמש במערכת.

1. **תיחום הפרויקט**

* הפרויקט עוסק בתחומים רבים: בתחום הרשתות במידע שעובר ברשת, בתחום מערכות ההפעלה הפעלת טרדים רבים, תחום ההצפנה המידע מוצפן והמערכת בודקת שהקובץ מגיע בשלמותו.
* המערכת לא מטפלת בתוכן הקובץ, ולא בודקת שלא נשלחים וירוסים או דברים זדוניים. המערכת גם מוגבלת לשרת הפנימית של בית הספר, ולא מאפשרת לקוחות חיצוניים מחוץ לרשת .

1. **קיימות**

הפרויקט מתקשר לקיימות כדור הארץ בכך שהופך את הקבצים הדיגיטליים ליותר נגישים ומהירים ובכך מעודד שימוש בקבצים דיגיטליים ובכך מוריד את השימוש בקבצים על הנייר וזה שומר על העצים ועל כדור הארץ. כמו כן הורדה מהירה יותר של הקבצים חוסכת חשמל, שימוש ברשת ובמחשבים.

**פרק ב' - nexus - אפיון**

1. פרוט המערכת:

* תיאור מפורט יותר של המערכת - נקסוס היא רשת של לקוחות שמוכנים ומעוניינים לתקשר אחד עם השני בשביל האופציה להוריד קבצים אחד מהשני בשביל מהירות הורדה גבוהה יותר. הרעיון להוריד חלק שונה של הקובץ המבוקש במקביל ממשתמשים שונים ולהרכיבו מחדש ככל שיהיו יותר משתמשים שהורידו את הקובץ מהירות ההורדה תלך ותגדל
* מה היא אמורה לעשות -בעת בקשת הורדת קובץ, המערכת תזהה את כל המשתמשים שכבר הורידו את הקובץ המבוקש אליהם והיא תוריד מכל אחד מהם חלק אחר – כך שתהליך ההורדה יבוצע במקביל, תבנה את הקובץ מחדש אצל המשתמש תוך הבטחת שלמות ותציג אותו למשתמש.

1. מה היכולות שהיא תעניק למשתמש , פירוט היכולות:

* הורדת קבצים – המערכת תאפשר ללקוח להוריד קבצים במהירות גבוהה יותר תוך מתן ערובה למהימנות ושלמות הקבצים המורדים.
* אבטחה – המערכת תאפשר לזהות קובץ פגום, כבר במהלך ההורדה, ולא רק בסיום ההורדה, ותתריע בהתאם למשתמש.

1. פירוט הבדיקות ('קופסא שחורה')

(בדיקות לביצוע אך ורק עם קובץ ההרצה, ללא הקוד, בדגש על כך שכל אחד לפי הכתוב בטבלה יצליח לבצע את הבדיקה)

| **מספר** | **שם הבדיקה (שם שיעיד על תוכן הבדיקה)** | **מה אמורה לבדוק** | **איך מתכננים לבדוק**  **(לתאר בפירוט את שלבי הבדיקה)** |
| --- | --- | --- | --- |
| 1 | הורדת קובץ ממחשב אחד | שמחשב מצליח להוריד קובץ | לחבר 2 לקוחות שלאחד מהם אין קובץ מסוים לאחד מהם יש, ולנסות להוריד את הקובץ למחשב שאין לו את הקובץ בסיום ההורדה לפתוח את הקובץ ולבדוק שהוא זהה למקור. |
| 2 | הורדת קובץ מכמה מחשבים | שהמחשב מוריד את הקובץ במהירות גדולה יותר בעזרת כמה מחשבים | לחבר כמה מחשבים שלכולם יש את אותו הקובץ ואז לחבר מחשב נוסף שאין לו את הקובץ ולבקש להוריד את הקובץ  ולמדוד את הזמן שלקח לו להוריד את הקובץ, להשוואת את הזמן לזמן שלוקח ללקוח להוריד את אותו הקובץ בעזרת מחשב אחד |
| 3 | התרעת על קובץ לא שלם | שהמערכת מתריעה ללקוח שחלק מהקובץ או הקובץ כולו שהוא הוריד ממחשב מסוים לא תואם את הקובץ שהוא היה אמור להוריד | לחבר 2 מחשבים לרשת לבקש להוריד קובץ מסוים, המחשב שמורידים ממנו את הקובץ צריך לערוך את הקובץ שהוא לא יהיה שלם, ולבדוק האם המערכת מצליחה לזהות את הבעיה. |
| 4 | בדיקת פירוק ואיחוד קובץ | שיש הצלחה לפרק קובץ לחלק אותו לכמה מחשבים, ואז לקבל את החלקים מהמחשבים חזרה ולחבר | לחבר 2מחשבים ולהוריד את קובץ ולבדוק את הזמן שלוקח לו להוריד  לאחר מכן לפתוח מחשב נוסף להוסיף לו את הקובץ, ולמחוק את הקובץ שהורדת בפעם הראשונה אצל המחשב הראשון ואז לבקש שוב להוריד את אותו הקובץ ולבדוק האם לוקח לו פחות זמן העובדה שזמן ההורדה קטן מעידה על זה שהקובץ ירד מכמה מחשבים ושהפירוק והאיחוד עובד |
| 5 | בדיקה של קישור ממשק לקוח למערכת | שאחרי שהלקוח מבקש קובץ מסוים ההודעה עוברת בתקשורת והוא מקבל את הקובץ | לפתוח לקוח ולבדוק שמופיע הממשק משתמש ולבקש ממנו להוריד קובץ ולבדוק האם הקובץ יורד |
| 6 | עבודה במקביל | שניתן להוריד כמה קבצים במקבלים (כמה משתמשים שונים ) | לפתוח 2 מחשבים ולבדוק כמה זמן לוקח להם להוריד קובץ מסוים, לאחר מכן למחוק את הקובץ ממחשב אחד, ואז לפתוח כמה מחשבים ולבקש להוריד את אותו הקובץ מהפעם הראשונה ובנוסף לזה לבקש קובץ אחר ממחשב אחר ולבדוק שהעבודה במקביל לא משפיעה על הזמן של הורדת הקובץ הראשון |
| 7 | הורדה במקבילים | שניתן להוריד את אותו הקובץ במקביל אצל שני לקוחות או יותר | לפתוח כמה מחשבים ולבקש מ2 מהם קובץ שקיים בכמה מחשבים אחרים |
| 8 | מערכת עדכנית | שלא ניתן להוריד קובץ מהמערכת אם הוא נמחק מהלקוחות שהורידו אותו | לפתוח מחשב ולהוסיף אליו קובץ שקיים רק אצלו ואז לבדוק האם אפשר להוריד את הקובץ לאחר מכן למחוק את הקובץ בכל המחשבים שהקובץ קיים עליהם ולבדוק האם הוא עדיין מופיע במחשב שלישי שלא היה עליו את הקובץ |

1. תכנון לוח זמנים לפרויקט

* תכנון לוח זמנים ראשוני
* עבור כל פעילות בלוח: זמן התחלה מתוכנן, זמן סיום מתוכנן , זמן התחלה בפועל, זמן סיום בפועל והערות. יש לכלול את התאריכים שניתנו לכם עבור כל פרק כ'זמן סיום מתוכנן' ואת החודש של הקידוד לפרק לאבנים גדולות ולתת להן זמנים

| **פעילות** | **זמן התחלה מתוכנן** | **זמן סיום מתוכנן** | **זמן התחלה בפועל** | **זמן סיום בפועל** | **הערות** |
| --- | --- | --- | --- | --- | --- |
| יזום | 6.11.2023 | 25.11.2023 | 6.11.2023 | 7.11.2023 |  |
| אפיון | 8.11.2023 | 1.12.2023 | 9.11.2023 |  |  |
| ניתוח | 11.11.2023 | 15.12.2023 |  |  |  |
| עיצוב | 1.12.2023 | 15.1.2023 |  |  |  |
| קוד (לפרוט לאבנים גדולות) | 1) חלוקת קובץ לחלקים במקביל שמירת hash  15.1.2023  2) בניית קובץ מפורק במקביל בדיקת hash  15.1.2023  3)יצירת קובץ torrent  22.1.2023  4)מימוש תקשורת והעברת חלקי הקובץ  1.2.2023  5)בניית ממשק משתמש  14.2.2023    6)ניטור תיקיות  21.2.2023  7)מימוש הודעה דרך ממשק המשתמש ובקשה דרך הממשק  28.2.2023 | 1)  22.1.2023  2)  22.1.2023  3)  1.2.2023  4)  14.2.2023  5)  21.2.2023  6)  28.2.2023  7)  1.3.2023 |  |  |  |
| גרסה ראשונית | 15.1.2023 | 1.3.2023 |  |  |  |
| מסמך בדיקות | 1.4.2023 | 10.4.2023 |  |  |  |
| מדריך למשתמש | 10.4.2023 | 1.5.2023 |  |  |  |
| גרסה סופית | 1.5.2023 | 10.5.2023 |  |  |  |
| סגירת תיק פרויקט | 1.5.2023 | 10.5.2023 |  |  |  |
| מצגת הסבר | 1.5.2023 | 10.5.2023 |  |  |  |

* **לוח הזמנים – הינו מסמך חי ויש לעדכנו לאורך כל הפרויקט**

1. ניהול סיכונים בפרויקט

* זיהוי סיכוני הפרויקט, ניתוחם ומענה עליהם, הטבלה תמוין מסיכון גבוה לנמוך (הגבוהים באדום, בינוני –כתום , קל – צהוב) סיכון שיוסר יצבע בירוק ויעבור לתחתית המסמך

| **הסיכון** | **פירוט הסיכון** | **רמת הסיכון**  **(קל/בינוני/קשה)** | **תיאור דרכים (לפחות 2 ) להתמודדות עם הסיכון ולהקטין אותו** | **מה בוצע בפועל** | **תאריך** |
| --- | --- | --- | --- | --- | --- |
| אי עמידה בזמנים | פרויקט לא יושלם | קשה | * הקדמת לו"זים משימות * ארגון הזמן בצורה יעילה * להתחיל בדברים היותר קשים |  |  |
| אי הצלחה להרכיב קובץ חזרה מחלקים | הפרויקט לא יושלם | קשה | * לבדוק שאפשר לממש את זה * לקרוא באינטרנט על פתרונות של אנשים ועל דרכי הפעולה שלהם |  |  |
| אי הצלחת פירוק הקובץ | פרויקט לא יהיה שימושי ובלי תועלת | קשה | * לבדוק שאפשר לממש את זה * לקרוא באינטרנט על פתרונות של אנשים ועל דרכי הפעולה שלהם |  |  |
| הקובץ ירד לאט | הפרויקט יהיה פחות יעיל | בינוני | כתיבת קוד יעיל מההתחלה  כתיבת אלגוריתם חכם |  |  |
| לא להצליח להבין את הניהול הגרפי | הפרויקט לא יראה טוב ויהיה הרבה פחות מזמין | קל | לקרוא באינטרנט לנסות להגיע להבנה עמוקה של המחלקה  להתייעץ עם חברים שהצליחו והגיעו לתוצרים יפים |  |  |

* **ניהול סיכונים – הינו מסמך חי ויש לעדכנו לאורך כל הפרויקט**

**פרק ג' - 'שם המערכת' - מסמך ניתוח**

פרוט יכולות המערכת

1. לעלות קובץ חדש למערכת - קובץ זה יכול להיות קובץ שקיים במערכת או קובץ חדש שהלקוח יצר והוא מעוניין לשתף אותו עם המערכת
2. להוריד קובץ שקיים במערכת - הלקוח מקבל רשימה עם קבצים שקיימים במערכת והוא יכול לבחור איזה אחד מהם הוא רוצה להוריד, אותו הקובץ ירד במהירות גבוהה.
3. פירוק קובץ – מפרקים את הקובץ לחלקים שווים ושמירת הhash של כל קובץ ושל הקובץ הכללי
4. חיבור קובץ - לאחר קבלת כל החלקים של הקובץ לאחר בדיקת כל חלק קובץ עם הhash הקיים ובדיקה הקובץ הסופי עם הhash הסופי
5. בניית קובץ torrent - בניית קובץ torrent עם ה headerהמתאימים לאחר השגתם מפירוק הקובץ
6. עדכון קובץ torrent - עדכון header בתוך קובץ הtorrent לאחר שלקוח שינה את הסטטוס של שמירת הקובץ אצלו (מחק או הוסיף את הקובץ) או לחילופין משתתפי הקובץ לא מחוברים
7. יצירת רשימה של קבצי המערכת – הרשימה נמצאת אצל השרת לאחר שקיבל את הנתונים מכל הלקוחות, כל שינוי ברשימה ישלח לכל לקוח.
8. שליחת חלקי הקובץ – כל לקוח הוא שרת שיכול לקבל בקשה מלקוח אחר להוריד חלק מקובץ שקיים אצלו, לאחר הבקשה הלקוח ישלח את חלק הקובץ לאותו לקוח
9. קבלת חלקי הקובץ – כל לקוח יכול לקבל מהשרת אצל לקוח אחר חלק מקובץ, לאחר הקבלה הלקוח יוודא שהחלק תקין על ידי hash
10. ניטור תיקייה – כל לקוח מנטר תיקייה מוסתרת (שהיא תיקיית המערכת שבה נשמרים הקבצים המורדים), את תוצאות הניטור הלקוח שולח לשרת.
11. ממשק משתמש – מציג רשימת קבצים קיימים במערכת, מאפשר בחירת קובץ מהרשימה הזו להורדה ומאפשר העלאת כל קובץ מכל מקום במחשב המשתמש.  
    כמו כן הממשק יציג חיווים על הצלחת/כשלון העלאת קבצים ועל הצלחה/כשלון הורדת קבצים למשתמש.
12. התחברות לשרת –  במסגרת פעולת ההתחברות, הלקוח ישלח לשרת את רשימת הקבצים הקיימים אצלו בתיקייה המנוטרת ובמקביל יקבל מהשרת את רשימת הקבצים הזמינים במערכת.
13. תקשורת Peer2Peer – מול כל אחד מאלו שהקובץ המבוקש זמין אצלם

שם היכולת: פירוק קובץ

מהות היכולת: פירוק קובץ לחלקים שווים

אוסף יכולות:

* חישוב hash לכל הקובץ
* פירוק הקובץ לחלקים שווים
* מספור כל חלק קובץ
* חישוב hash לכל חלק
* הצפנת כל חלק

רשימת אובייקטים: הצפנה/פענוח , קובץ , hash

שם היכולת: חיבור קובץ

מהות היכולת: חיבור חלקי קובץ לקובץ שלם

אוסף יכולות:

* פיענוח כל חלק
* השוואת hash של כל חלק
* חיבור החלקים לפי מספור
* השוואת hash של כל הקובץ

רשימת אובייקטים: הצפנה/פענוח, torrent file, קובץ, hash

שם היכולת: העלאת קובץ למערכת

מהות היכולת: הוספת קובץ שניתן להורדה למערכת

אוסף יכולות:

* **ממשק משתמש** – בחירת הקובץ להעלאה
* בניית הודעה על פי פרוטוקול
* הצפנה של הקובץ
* שליחת הקובץ לגורם מרכזי
* קבלת אישור שהעלאה הצליחה
* ברקע – לאחר קבלת האישור מהשרת העתקת הקובץ הנבחר לתיקיית הפרויקט (נסתרת) המנוטרת

רשימת אובייקטים: ממשק משתמש, הצפנה/פיענוח, תקשורת, פרוטוקול, קובץ, תיקייה מנוטרת

שם היכולת: בניית קובץ torrent

מהות היכולת: בניית קובץ Torrent המכיל פרטים רלוונטיים

אוסף יכולות:

* **פירוק הקובץ**
* בניית קובץ ה Torrent עם כל המידע הנחוץ (שם קובץ, כמות חלקים, Hash של כל חלק, hash של הקובץ כולו, רשימת כתובות מהם ניתן להוריד את הקובץ)
* עדכון בסיס נתונים עם שם הקובץ
* שמירת קובץ ה Torrent אצל הגורם המרכז

רשימת אובייקטים: הצפנה/פיענוח, תבנית קובץ torrent, בסיס נתונים, תיקיית קבצי ה Torrent

שם היכולת: עדכון קובץ torrent

מהות היכולת: עדכון פרטים בתוך קובץ הtorrent

אוסף יכולות:

* פתיחת הקובץ
* עדכון רשימת משתתפים

רשימת אובייקטים: קובץ torrent רשימת משתתפים

שם היכולת: יצירת רשימת קבצים להורדה

מהות היכולת: יצרת רשימת קבצים ניתנים להורדה מהמערכת

אוסף יכולות:

* קבלת מכל לקוח ברשת את רשימת הקבצים הקיימים אצלו בתיקייה המנוטרת
* אימות הנתונים מול בסיס הנתונים:
  + שם קובץ שלא קיים בבסיס נתונים משמעותו שלא קיים עבורו קובץ Torrent - נשלחת למחשב הספציפי הודעה למחוק את הקובץ מהתיקייה המנוטרת
  + שם קובץ שקיים שקיים בבסיס הנתונים משמעותו שקיים עבורו קובץ Torrent– הוספת ה IP לקובץ הtorrent (במידה ולא קיים)
* במידה ושם הקובץ נוסף לראשונה והיה קיים קובץ torrent אז לעדכן את הרשימה

רשימת אובייקטים: תיקייה מנוטרת, רשימת קבצים, תקשורת, בסיס נתונים , קובץ Torrent

שם היכולת: הורדת קובץ מהמערכת

מהות היכולת: הורדה מהירה של קובץ מהלקוחות האחרים במערכת

אוסף יכולות:

* **ממשק משתמש** – בחירת הקובץ להוריד
* קליטת נתונים
* בדיקת תקינות
* בניית הודעה על פי פרוטוקול
* הצפנה
* שליחה לשרת
* קבלת תשובה קובץ torrent מהשרת
* בקשת **שליחת כל חלקי הקובץ**
* קבלת תשובה
* **חיבור קובץ**
* שמירת הקובץ בdownload והעתקתו לתיקייה המנוטרת

רשימת אובייקטים: ממשק משתמש, הצפנה/פיענוח, תקשורת, torrent file , פרוטוקול

שם היכולת: שליחת חלקי קובץ

מהות היכולת: שליחת חלק ספציפי מהקובץ

אוסף יכולות:

* קבלת בקשה
* פענוח
* פירוק לפי פרוטוקול
* פתיחת הקובץ המבוקש
* מציאת החלק המתבקש
* בניית תשובה לפי פרוטוקול
* הצפנה
* שליחת חלק הקובץ

רשימת אובייקטים: תקשורת, קבצים, הצפנה, פרוטוקול

שם היכולת: ניטור תיקייה

מהות היכולת: ניטור התיקייה ושליחת עדכונים לשרת

אוסף יכולות:

* יצירת התיקייה במידה ואין
* בדיקת שינויים
* בניית הודעה על פי פרוטוקול
* הצפנת השינויים
* שליחת השינויים לשרת
* קבלת הנחיות מהשרת

מחיקת קובץ – מוחקים את הקובץ הספציפי מהתיקייה המנוטרת

הוספת קובץ מעתיקים את הקובץ הספציפי לתיקייה המנוטרת

* רשימת אובייקטים: תקשורת, קבצים, הצפנה , פרוטוקול תיקייה מנוטרת

שם היכולת: הצגת רשימת הקבצים

מהות היכולת: הצגת ממשק המערכת למשתמש – העלאת הורדת קובץ

אוסף יכולות:

* קבלת הרשימה של הקבצים מהשרת בכל רגע נתון והצגתם למשתמש
* במידה ונבחר העלאת קובץ - הצגת הקבצים במחשב ואפשרות בחירה למשתמש - **העלאת קובץ**
* במידה ונבחר הורדת קובץ – **הורדת קובץ מהמערכת**
* רשימת אובייקטים: תקשורת, קבצים, הצפנה, גרפיקה

**פרק ד' - 'שם המערכת' - העיצוב**

1. תיאור הארכיטקטורה של המערכת המוצעת

* תיאור החומרה –

שרת: מחשב מבוסס SQL + Windows בסיס נתונים

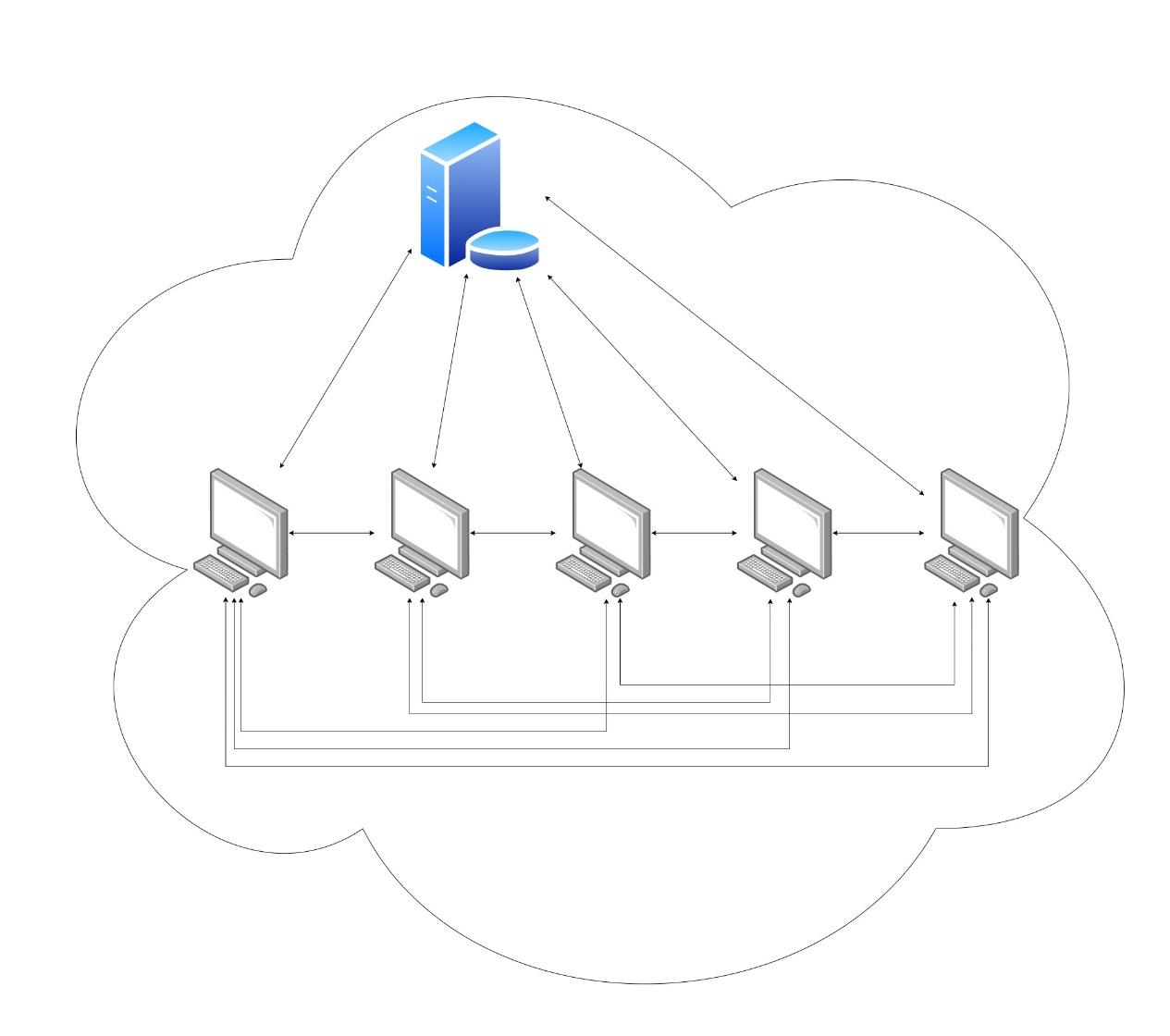
לקוחות: מחשבים מבוססים Windows

כל המחשבים (שרת ולקוחות) נמצאים על אותה הרשת

* רכיבים שונים והקשרים ביניהם

כל הלקוחות מתחברים לשרת(שרת מרובה לקוחות) , שליחת הקבצים בעזרת P2P

לכל לקוח יש רכיב של שרת מרובה לקוחות



1. תיאור הטכנולוגיה הרלוונטית

* פירוט ההחלטות שנלקחו בחשבון בעת בחירת הטכנולוגיה

**שפת תכנות** – Python 3.7, DB SQL

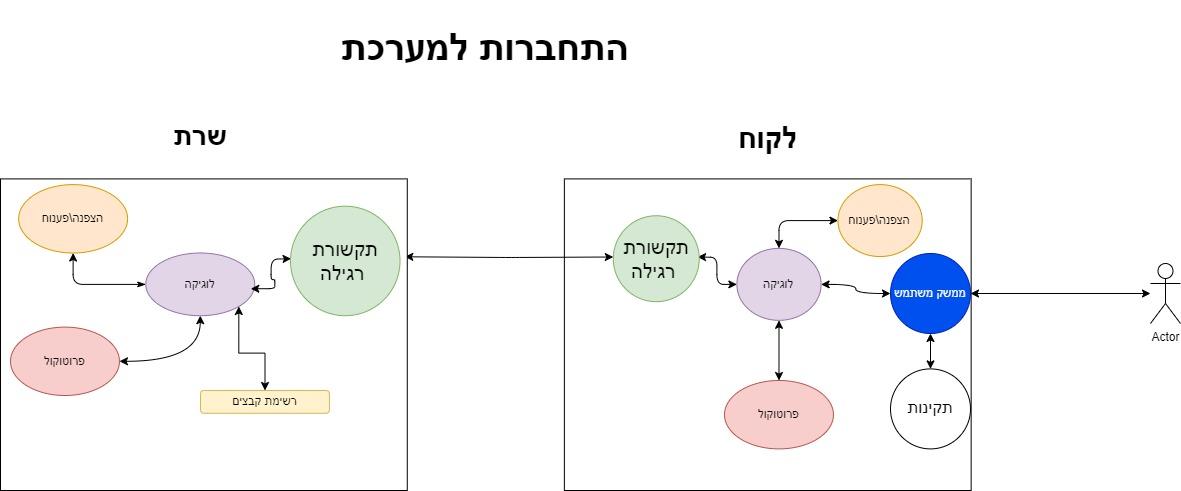
**מערכת הפעלה** - Windows

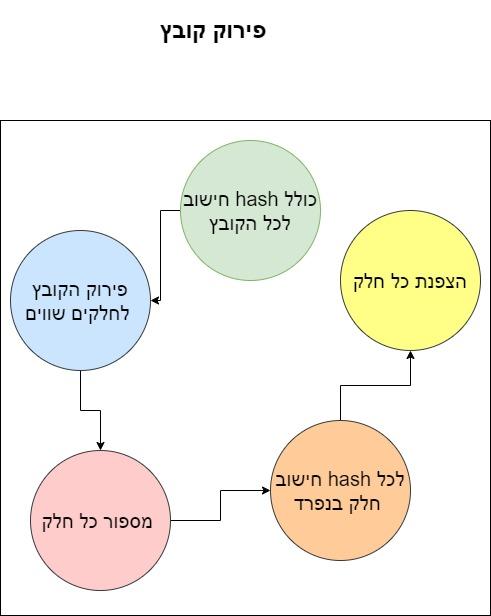
**תקשורת** – תקשורת מבוססת Sockets בעזרת פרוטוקול TCP

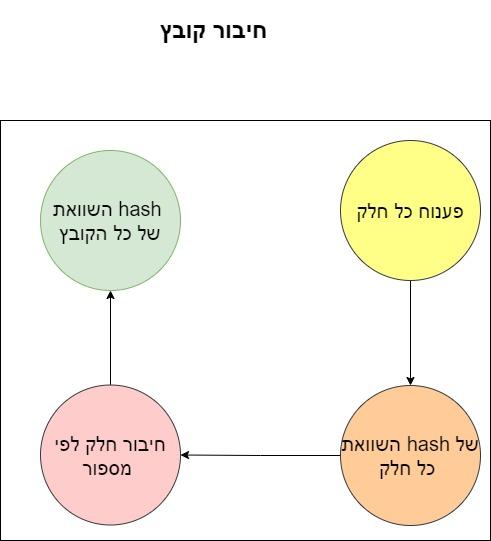
**תחומי עניין** – הצפנה (כל מה שעובר ברשת מוצפן) , בסיס נתונים טבלאי( SQL)

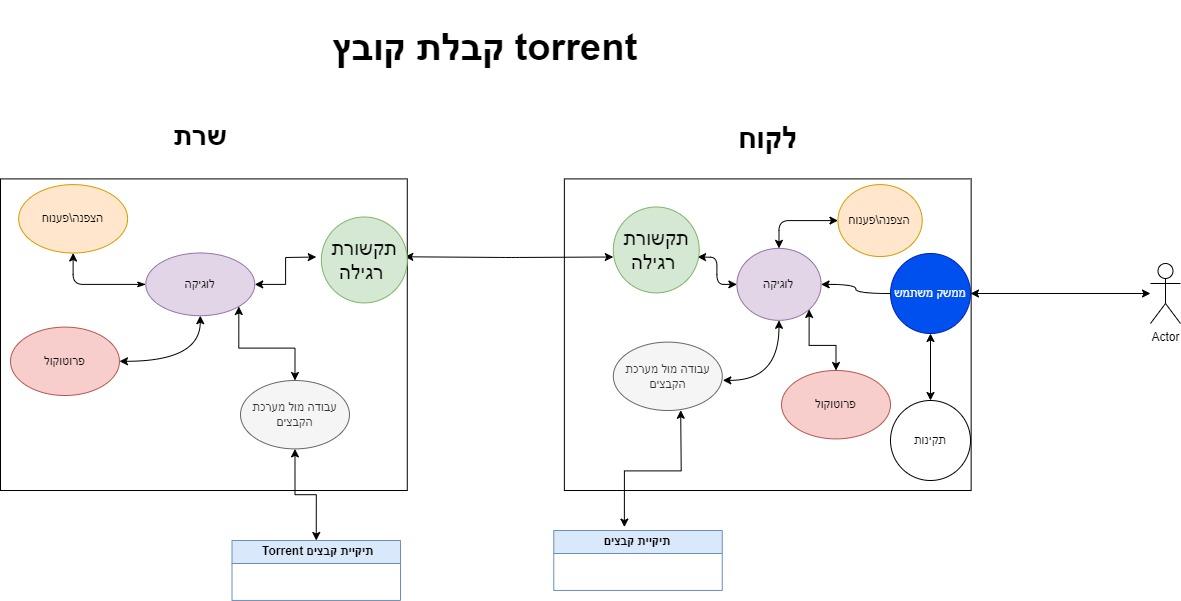
גרפיקה עיצוב בעזרת WXPython , עבודה עם קבצים, עבודה עם hash

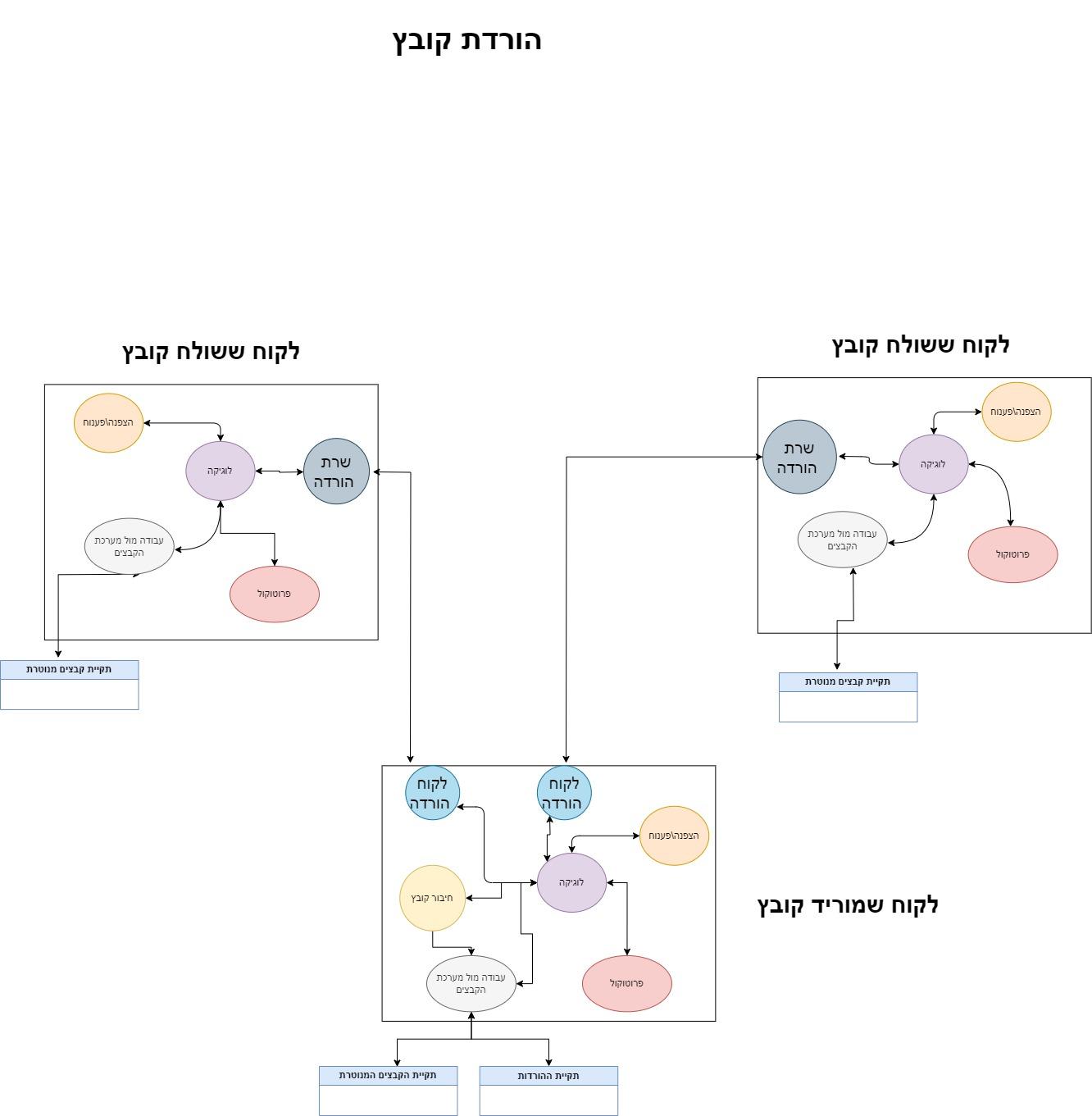
1. תיאור מודולים בהם נעשה שימוש

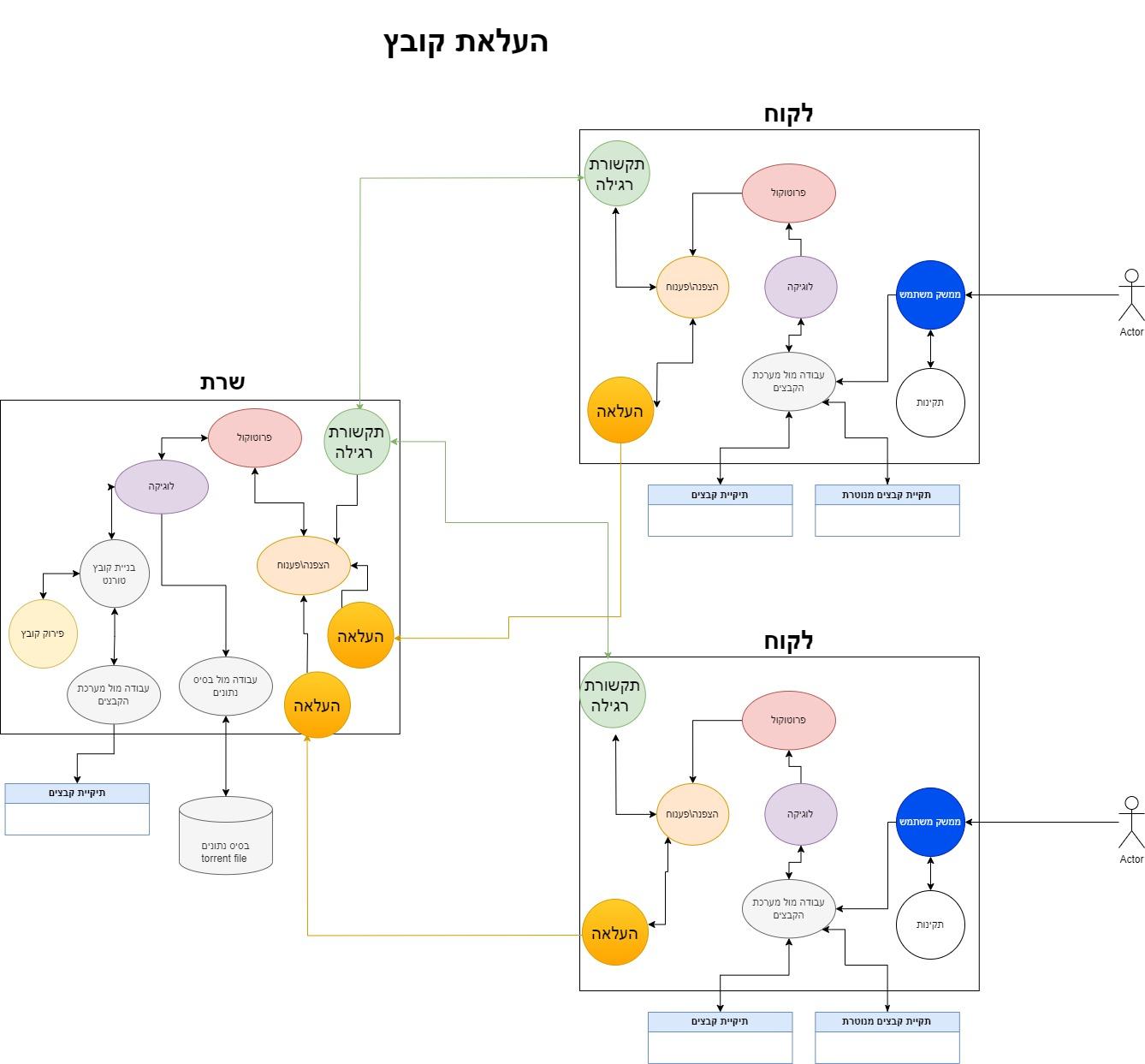


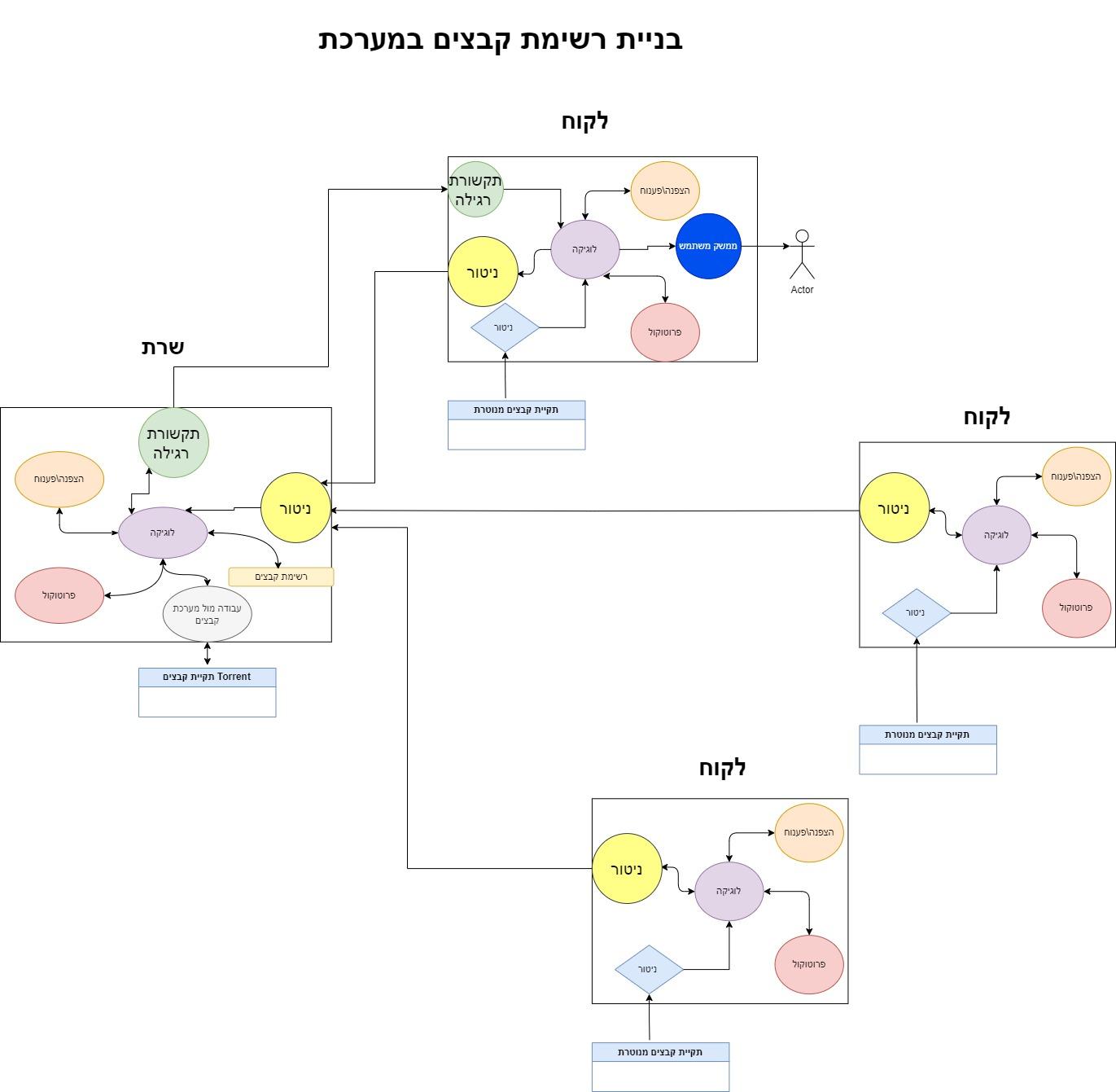


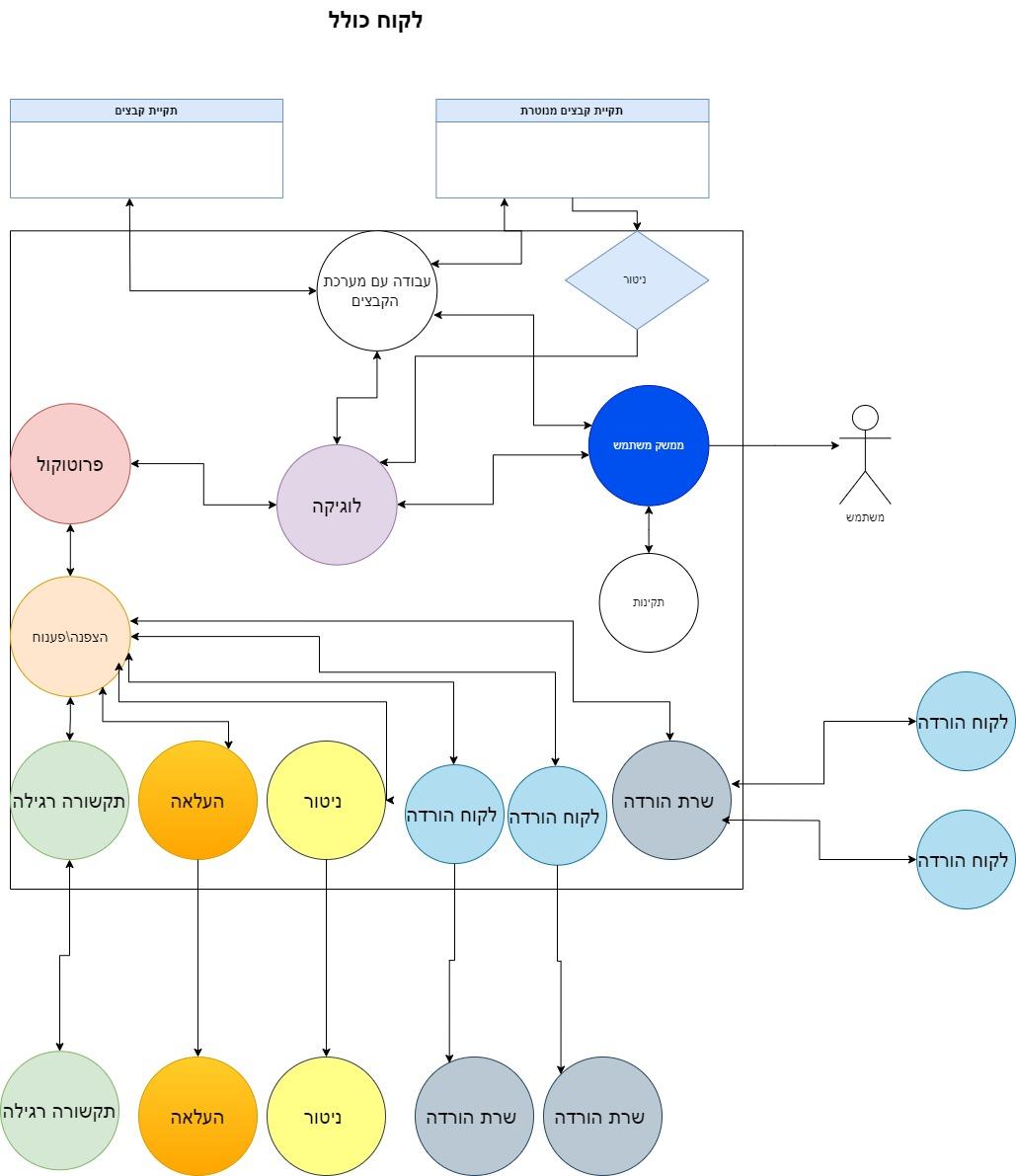


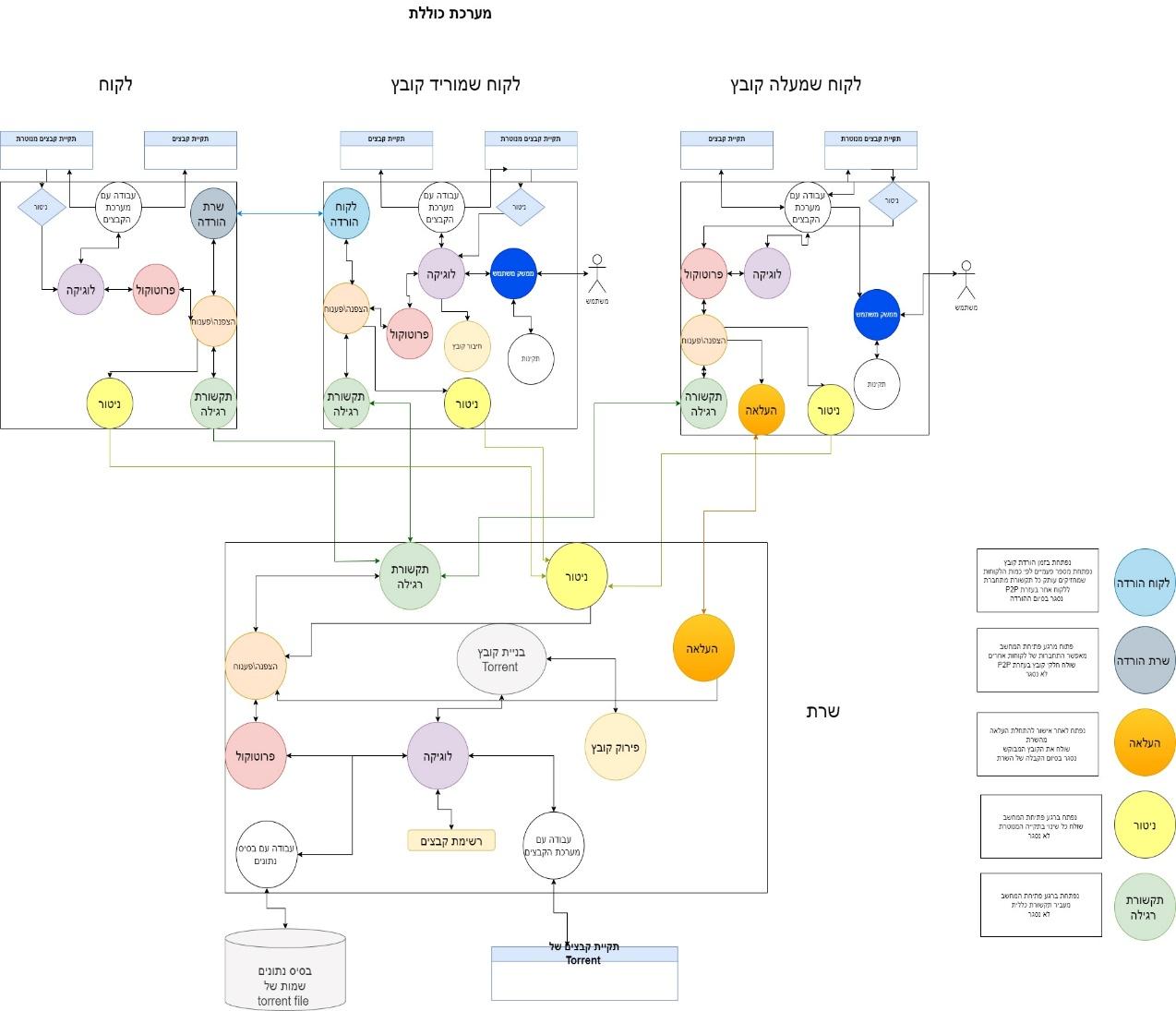












* סקירת כל המודולים המרכיבים את המערכת וקשרי הגומלין ביניהם:

**מודולים מיובאים לפרויקט:**

| **חבילה** | **ייעוד** |
| --- | --- |
| Socket | תקשורת |
| Hashlib | הצפנה |
| Select | שרת מרובה לקוחות |
| Wxpython | Gui |
| Sys |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**מודולים שלי בפרויקט:**

**מודולים בשרת**

| **serverProtocol** | | |
| --- | --- | --- |
| מחלקה מאגדת את כל הפעולות הנדרשות לטיפול בפרוטוקול בצד שרת | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| unpack | data – בתים שהתקבלו בתקשורת | מפרקת את ההודעה ע"פ הפרוטוקול ומחזירה tuple:  opcode – קוד ההודעה  List\_par – רשימת פרמטרים |
| build\_part\_of\_key | b – מספר ראשוני | בונה הודעה העברת מפתח ע"פ הפורמט ומחזיר אותה  00b |
| Response\_for\_torrent\_request | Torrent\_file – קובץ הטורנט בפורמט json | בונה הודעה העברת קובץ טורנט ע"פ הפורמט ומחזיר אותה |
| Response\_for\_upload\_request | File\_name – שם הקובץ  Port – הפורט של הסוקט הנפתח | בונה הודעה פתיחת סוקט העלאה ע"פ הפורמט ומחזיר אותה |
| Response\_for\_upload |  | בונה הודעה אישור קבלת הקובץ ע"פ הפורמט ומחזיר אותה |
| Delete\_file\_from\_folder | File\_name – שם הקובץ | בונה הודעה הוראת מחיקת קובץ ע"פ הפורמט ומחזיר אותה03file\_name |
| Create\_string\_of\_list | list\_of\_open\_files | בונה הודעה ע"פ פרוטוקול ומחזיר אותה |

| **mainServer** | | |
| --- | --- | --- |
| מחלקה שמאגדת את פעולות למימוש הלוגיקה בשרת הראשי | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| server | עצם של השרת הכללי | |
| nitur | עצם של שרת הניטור | |
| general\_queue | תור הודעות שהגיעו לשרת הכללי | |
| nitur\_queue | תור הודעות שהגיעו לשרת הניטור | |
| general\_commands | מילון פעולות כלליות (עבור השרת הכללי) בפורמט general\_commands["opcode"]=proc\_name | |
| nitur\_commands | מילון פעולות כלליות (עבור שרת הניטור) בפורמט nitur\_commands["opcode"]=proc\_name | |
| list\_of\_open\_files | רשימה ששומרת את כל הקבצים שניתן להוריד אותם | |
| unused\_ports | שומר את כל הפורטים בתוך גנרטור | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| main |  | פעולה ראשית:  יוצרת תורים עבור ההודעות שמגיעות מהשרת הכללי ומשרת הניטור  יוצרת את השרת הראשי  יוצרת את שרת הניטור  מפעילה כthread את handle\_general\_msgs  מפעילה כ thread את handle\_nitur\_msgs |
| handle\_general\_msgs  (מופעל כ thread ) | server  general\_queue | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  מפעילה פעולה רלוונטית ע"פ general\_commands["opcode"]=proc\_name |
| handle\_nitur\_msgs  (מופעל כ thread ) | nitur  nitur\_queue | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  מפעילה פעולה רלוונטית ע"פ nitur\_commands["opcode"]=proc\_name |
| handle\_upload\_file  (מופעל כ thread ) | File\_name  Server\_comm\_upload  sever  queue  Ip | יוצרת עצם בסיס הנתונים db  מוציא את הקובץ מהqueue  מפעיל את הפעולה serverfile.Create\_torrent\_file  נקראת הפעולה db.add\_torrent  נקראת הפעולה serverprotocol. Response\_for\_upload  נשלח הודעת אישור ללקוח בעזרת server.send  ונסגר העצם של ה Server\_comm\_upload |
| Delete\_file\_list\_of\_open\_files | Name\_of\_file  server  ip | מוחק את השם של הקובץ מרשימה  קורא לפעולה serverfile. delete\_ip\_from\_torrent  במידה וחזר true  יוצר הודעה בעזרת serverProtocol. Create\_string\_of\_list  ושולח לכולם את העדכון בעזרת server.sendall |
| Add\_file\_list\_of\_open\_files | Name\_of\_file  server  ip | מוסיף את השם של הקובץ לרשימה  קורא לפעולה serverfile.  Add\_ip\_to\_torrent  במידה וחזר true  יוצר הודעה בעזרת serverProtocol. Create\_string\_of\_list  ושולח לכולם את העדכון בעזרת server.sendall |
| Delete\_file\_in\_client | Ip  File\_name  server | קורא לפעולה serverfile. delete\_ip\_from\_torrent  קורא לפעולה serverprotocol. Delete\_file\_from\_folder  שולח הודעה ללקוח שימחק את הקובץ מהתיקייה המנוטרת בעזרת server.send |
| send\_torrent | filename  ip  server | משתמש ב server\_files. Get\_torrent\_file(filename)  קורא לפרוטוקול Response\_for\_torrent\_request  ואז שולח server.send |
| Create\_upload\_socket | File\_name  server  Ip | יוצרת עצם בסיס הנתונים db  מפעיל את פעולת have\_torrent  במידה וfalse  יוצר queue מקבל פורט מתוך הגרנטור unused\_port יוצר עצם מתוך servercomm  ןמפעיל כ thread את לפעולה  handle\_upload\_file(queue, Server\_comm\_upload)  יוצר הודעה בעזרת  Response\_for\_upload\_request  ושולח הודעה לip שביקש בעזרת server.send |

| **ServerFiles** | | |
| --- | --- | --- |
| מחלקה מאגדת את כל הפעולות שנדרשות לטיפול במערכת הקבצים בשרת | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| Create\_torrent\_file | File\_name  File\_data | מחושב hash לכל הקובץ ולכל חלק  קורא לפעולה \_Build\_torrent\_file |
| Build\_torrent\_file\_ | File\_data – הdata של קובץ הtorrent  File\_name  Full\_hash  Parts\_hash | יוצר קובץ torrent כפורמט של קובץ json עם הפרמטרים הנחוצים |
| Add\_ip\_to\_torrent | File\_name – שם הקובץ  Ip - הכתובת | במוסיף לשדה של הip הפתוחים בתוך קובץ הtorrent את הקובץ  מחזיר Boolean האם צריך לשלוח עדכון ללקוחות |
| Get\_torrent\_file | file\_name – שם הקובץ | מחזיר מחרוזת של הקובץ torrent בפורמט json ע"פ פרוטוקול הקובץ ואם הקובץ לא קיים מחזיר None |
| delete\_ip\_from\_torrent | File\_name – שם הקובץ  Ip - הכתובת | מוריד מהשדה של הip הפתוחים בתוך קובץ הtorrent את הקובץ במידה בקיים  מחזיר Boolean האם צריך לשלוח עדכון ללקוחות |

|  |  |  |
| --- | --- | --- |
| **DB** | | |
| מחלקה היוצרת עצם לטיפול וניהול בסיס הנתונים של הפרויקט | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| dbName | שם בסיס הנתונים " NexusDB" | |
| tblName | שם הטבלה " torrentTBL " | |
| Conn | מצביע לבסיס הנתונים שנוצר | |
| Curr | מצביע לטבלה | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| \_\_init\_\_ |  | מאתחלת את המשתנים ומפעילה את \_buildDB |
| \_buildDB |  | מתחברת לבסיס הנתונים ויוצרת (אם לא קיימת) את הטבלה עם השדות היעודיים |
| add\_torrent | Torrent\_file\_name – שם הקובץ | מוסיף את שם הקובץ לבסיס הנתונים במידה ולא קיים (שימוש ב have\_torrent) במידה וקיים מחזיר None |
| have\_torrent | Torrent\_file\_name – שם הקובץ | מחזיר Boolean האם השם קיים בבסיס הנתונים |

**מודולים בלקוח**

| **ClientFiles** | | |
| --- | --- | --- |
| מחלקה המאגדת את כל הפעולות שנדרשות לטיפול במערכת הקבצים בלקוח | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| Add\_part\_of\_file | Path of file – הדרך לקובץ  Name of file - שם קובץ  Number of file - מספר חלק | הקובץ נוצר במקרה שהוא לא קיים  החלק נוסף לפי חישוב המיקום שלו בתוך הקובץ |
| Get\_part\_of\_file | Path of file – הדרך לקובץ  Name of file - שם קובץ  Number of file - מספר חלק | Data – החלק  במקרה שהחלק הוא מספר 1- מוחזר כל הData |
| save\_file | Path of file – הדרך לקובץ  Name of file - שם קובץ | מעתיק את הקובץ מהמיקום הנוכחי למחיצת הניטור אחרי הוראת השרת |
| Delete\_file | Path of file – הדרך לקובץ  Name of file - שם קובץ | מוחק קובץ שנמצא בתיקיית הניטור אחרי הוראת השרת  לפי הדרך והשם של הקובץ  או קובץ torrent לאחר שנגמרה ההורדה |
| Save\_torrent\_file | Torrent file- שם הקובץ | הקובץ נשמר בתיקייה מוסתרת |
| Get\_ips\_from\_torrent | Torrent file- שם הקובץ | מחזיר רשימה עם כל הip שיש להם את הקובץ |
| Get\_total\_hash | Torrent file- שם הקובץ | מחזיר את הhash הכולל של הקובץ |
| get\_number\_of\_parts | torrent file שם הקובץ | מחזיר רשימה של מספר החלקים[1,2,3,4,5] |
| Get\_part\_hash | Torrent file- שם הקובץ  Number part – מספר החלק | מחזיר את הhash של החלק |

| **ClientProtocol** | | |
| --- | --- | --- |
| מחלקה מאגדת את כל הפעולות הנדרשות לטיפול בפרוטוקול בצד הלקוח | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| unpack | data – בתים שהתקבלו בתקשורת | מפרקת את ההודעה ע"פ הפרוטוקול ומחזירה tuple:  opcode – קוד ההודעה  List\_par – רשימת פרמטרים |
| build\_part\_of\_key | A– מספר ראשוני | בונה הודעה העברת מפתח ע"פ הפורמט ומחזיר אותה 00A |
| Request\_torrent\_file | File name – שם הקובץ | בונה הודעת בקשה של קובץ טורנט ע"פ פרוטוקול |
| Request\_upload | File name – שם הקובץ | בונה הודעת בקשת העלאה של קובץ חדש למערכת ע"פ פרוטוקול |
| Upload\_file | File name – שם הקובץ  Data – המידע של הקובץ | בונה הודעת העלאת קובץ חדש למערכת ע"פ פרוטוקול |
| Request\_part\_file | File name – שם הקובץ  Number\_of\_part – מספר הקובץ | בונה הודעת בקשת חלק קובץ מלקוח אחר ע"פ פרוטוקול |
| Send\_file\_part | Number\_of\_part – מספר הקובץ  Data – המידע של הקובץ | בונה הודעת שליחת חלק קובץ ללקוח שביקש ע"פ פרוטוקול |
| Added\_file\_nitur | File\_name | בונה הודעה שנוסף קובץ לתיקיית הניטור |
| Removed\_file\_nitur | File\_name | בונה הודעה שנמחק קובץ בתוך תיקיית הניטור |
| Changed\_file\_nitur | File\_name | בונה הודעה שהשתנה קובץ בתוך תיקיית הניטור |

­

| **mainClientUser** | | |
| --- | --- | --- |
| מחלקה שמאגדת את פעולות למימוש הלוגיקה בשרת הראשי | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| client | עצם של השרת הכללי | |
| gui\_class |  | |
| gui\_queue | התור שאליו נכנסות הודעות מהמשתמש | |
| general\_queue | תור הודעות שהגיעו לשרת הכללי | |
| list\_of\_open\_files | רשימה ששומרת את כל הקבצים שניתן להוריד אותם | |
|  |  | |
|  |  | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| main |  | פעולה ראשית:  יוצר את הגרפיקה  יוצרת תורים עבור ההודעות שמגיעות מהשרת הכללי ומהגרפיקה  ומפעיל כthread את הפעולות :  handle\_general\_msgs |
| handle\_general\_msgs  (מופעל כ thread ) | general\_queue | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  מפעילה פעולה רלוונטית ע"פ general\_commands["opcode"]=proc\_name |
| handle\_gui\_msgs  (מופעל כ thread ) | gui\_queue  client | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  מפעילה פעולה רלוונטית ע"פ gui\_commands["opcode"]=proc\_name |
| Request\_upload | Name\_of\_file  client | יוצר הודעה ע"פ פרוטוקול Clientprotocol. Request\_upload  שולח את ההודעה בעזרת client.send |
| request\_download | Name\_of\_file  client | יוצר הודעה ע"פ פרוטוקול Clientprotocol.Request\_torrent\_file  שולח את ההודעה בעזרת client.send |
| create\_socket\_upload\_file | port  name\_of\_file  client | יוצר עצם של clientcomm  client\_upload\_comm  עם הפורט  קורא לפעולה Get\_part\_of\_file עם -1 והשם של הקובץ  קורא לפעולה clientprotocol.Upload\_file  וקורא לפעולה client\_upload\_comm  .send |
| p2p\_download | file\_data | שומר את הקובץ torrent  יוצר רשימה באורך מספר החלקים עם value -0  יוצר queue של הודעות  ויוצר לכל אחד מהip clientcomm  נקראת הפעולה find\_first\_  ושולח בקשת חלק מכל אחד מהip  ומתחילה לולאה שמוציאה מהqueue את החלק משווה את hash אם הhash אינו שווה הלקוח מוסר מרשימת הלקוחות במידה והיא ריקה לאחר ההסרה נגמר תהליך ההורדה  נקראת הפעולה clientfile.add\_part\_of\_file  וקוראת לפעולה  \_find\_first  ומבקשת את החלק  בסיום החיבור משווה את הhash הכללי |
| \_find\_first | List\_of\_parts  Number of clients | מוציא את המספר הנמוך ביותר הראשון שנמצא מעלה את הערך שלו באחד מחזיר את הindex שלו ומתחיל טיימר במידה והמספר שווה למספר הלקוחות או ל3 אם הטיימר נמגר לפני שחלק הקובץ התקבל נגמר תהליך ההורדה |
| update\_file\_list | רשימת קבצים | שומר את רשימת הקבצים ושולח לgui שיעדכון את המשתמש |

| **mainClient** | | |
| --- | --- | --- |
| מחלקה שמאגדת את פעולות למימוש הלוגיקה בשרת הראשי | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| nitur | עצם של הניטור | |
| Nitur\_queue | טור הניטור | |
| Nitur\_comm | עצם תקשורת הניטור | |
| nitur\_comm\_queue | התור שמקבל את ההודעות התקשורת לגבי הניטור | |
| upload\_server | עצם של השרת הודעות הp2p | |
| upload\_server\_queue | התור של ההודעות שמתקבלות בp2p | |
| nitur\_commands | מילון שמאגד את הפעולות | |
|  |  | |
|  |  | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| main |  | יוצר את התורים של הניטור והP2P  יוצר כclientcomm  את עצם תקשורת הניטור  ויוצר עצם monitoring  ויוצר כservercomm את שרת ההודעות הp2p  ומפעיל את הפעולות  Handle\_p2p\_msgs  Handle\_nitur\_comm\_msgs  Handle\_nitur\_msgs |
|  |  |  |
| Handle\_p2p\_msgs  כthread) | Upload\_server\_queue  Server\_upload\_comm | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  ומפעיל את פעולה send\_part\_of\_file  אם הפרוטוקול מתאים |
| Send\_part\_of\_file | Server\_upload\_comm | בונה הודעה ע"פ פרוטוקול ושולח לו הודעה בעזרת הserver\_upload\_comm.send |
| Handle\_nitur\_comm\_msgs  כthread) | nitur\_comm\_queue  Nitur\_comm | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  ומפעיל את הפעולה delete\_file  אם הפרוטוקול מתאים |
| Delete\_file | filename | קורא לפעולה clientfile.delete |
| Handle\_nitur\_msgs  (כthread) | Nitur  Nitur\_queue | מוציאה הודעות מהתור מפרקת ע"פ פרוטוקול  מפעילה פעולה רלוונטית  nitur\_commands["opcode"]=proc\_name |
| Added\_file | Nitur\_comm | בונה הודעה ע"פ פרוטוקול  שולח את העדכון בעזרת הnitur\_comm.send |
| Changed\_file | Nitur\_comm | בונה הודעה ע"פ פרוטוקול  שולח את העדכון בעזרת הnitur\_comm.send |
| Removed\_file | Nitur\_comm | בונה הודעה ע"פ פרוטוקול  שולח את העדכון בעזרת הnitur\_comm.send |

| **monitoring** | | |
| --- | --- | --- |
| מחלקה ליצירת עצם שמנטר את התיקייה | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| path | הדרך לתיקייה | |
| msgs\_queue | תור שאליו נכנסות ההודעות | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| \_\_\_init\_\_\_ | Path - הדרך לתיקייה  msgs\_queue - תור שאליו נכנסות ההודעות | מפעילה thread שפועל על הפעולה \_monitoring\_folder |
| \_monitoring\_folder |  | מנטר את התיקייה ומעלה את השינויים לתור ע"פ פרוטוקול |

**מודולים משותפים**

| **Clientcomm** | | |
| --- | --- | --- |
| מחלקה שנוצר ממנה מופע על כל סוקט שמתחבר כלקוח | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| Server\_ip | הכתובת של השרת | |
| port | הפורט של הסוקט­ | |
| Message\_Queue | תור שאליו נכנסות ההודעות | |
| Client\_socket | הסוקט שמתחבר לשרת, נוצר בפעולה init | |
| Crypt\_object | אוביקט שמצפין ומפענח | |
| Zfill\_number | המספר בתים של האורך של ההודעה | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| \_\_init\_\_ | Server\_ip – הip של השרת  Port – הפורט של הסוקט  messageQueue - התור שאליו נכנסים ההודעות  zfill\_number – מספר הבתים של האורך של ההודעה | ונקראת הפעולה Recv\_messages\_ כthread |
| recv\_messages\_ |  | נוצר סוקט עם הIp והפורט  מתחבר לשרת  נקראת הפעולה xchange\_key\_ כל הודעה שנכנסת מפוענחת נכנסת לqueue |
| xchange\_key\_ |  | מחליפה מפתחות, יוצרת עצם הצפנה crypto\_object |
| Send | Message | שולח את ההודעה מוצפנת עם האורך שלה לא מוצפן |
| close\_socket |  | סוגר את הסוקט |

| **Servercomm** | | |
| --- | --- | --- |
| מחלקה שמנהל את התקשורת של השרת, בנוסף נמצאת בכל לקוח כל לאפשר ללקוחות אחרים להתחבר, שרת מרובה לקוחות. | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| Port | הפורט של הסוקט | |
| Message\_Queue | תור שאליו נכנסות ההודעות | |
| Server\_socket | הסוקט של השרת, נוצר בפעולה init | |
| Open\_clients | Open\_clients[socket] = [ip,cryptObject] | |
| Zfill\_number | המספר בתים של האורך של ההודעה | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| \_\_init\_\_ | Port – הפורט של הסוקט  Message\_Queue - התור שאליו נכנסים ההודעות  zfill\_number – מספר הבתים של האורך של ההודעה | נוצר המילון  ונקראת הפעולה Recv\_messages\_ כthread |
| Recv\_messages\_ |  | נוצר סוקט עם הIp והפורט,  כל הודעה שנכנסת מפוענחת נכנסת לqueue עם הip  כאשר לקוח חדש נכנס נקראת הפעולה xchange\_key\_  כthread |
| Xchange\_key\_ |  | מחליפה מפתחות, יוצרת עצם הצפנה ומעדכנת את המילון |
| find\_socket\_by\_ip\_ | ip | Socket |
| recv\_file\_ | file\_name  client  file\_length | מקבלת את הקובץ בלולאה ע"פ האורך הנדרש  הכנסה לתור המתאים את שם הקובץ, IP, bytearray |
| Send | Message – ההודעה  - ipלמי לשלוח | שולח את ההודעה לאחר הצפנה עם האורך שלה לא מוצפן לסוקט שהip שלו |
| Send\_all | message | שולח את ההודעה מוצפנת לכל הלקוחות המחוברים |
| Close\_socket |  | סוגר את הסוקט |

| **Encryption\_decryption** | | |
| --- | --- | --- |
| מחלקה שמייצרת את עצם ההצפנה עם הפעולות | | |
| **משתני המחלקה** | | |
| שם משתנה | תפקיד | |
| key | מפתח ההצפנה | |
| **פעולות במחלקה** | | |
| שם פעולה | טענת כניסה | טענת יציאה |
| \_\_init\_\_ | key | יוצרת עצם להצפנה סימטרית עם המפתח שהתקבל |
| Encrypt | message | ההודעה מוצפנת ע"י המפתח הסימטרי |
| decrypt | encrypt\_message | Message - ההודעה מפוענחת |
| pad\_ | message | ההודעה לאחר הוספת מידע |
| unpad\_ | Message | ההודעה לאחר הורדת מידע |
| Get\_key |  | **פעולה חיצונית** - מייצרת מספר ראשוני |
| Set\_key | a,b | **פעולה חיצונית** – מייצרת עצם הצפנה עם המפתח |
| hash | message | **פעולה חיצונית** - ההודעה לאחר גיבוב |

1. תיאור סביבת הפיתוח

* שפת התכנות שנבחרה לכתיבת הפרויקט – python 3.7
* עבודה מול בסיס הנתונים – שפת השאילתות SQL
* פירוט כלי הפיתוח והבדיקות הנדרשים לפיתוח
  + PyCharm בשביל כתיבת הקודDebug
  + DB Browser – בדיקת בסיס הנתונים
  + Wireshark - בדיקה מה עובר ברשת ווידוי קבלת הקבצים וההודעות
  + procmon - בדיקת התוכניות הפתוחות

1. תיאור האלגוריתמים המרכזיים בפרויקט:

סוגיה 1 – תשתית התקשורת מול השרת בפרויקט

| חלופה | תיאור | יתרונות | חסרונות |
| --- | --- | --- | --- |
| 1 | כל התקשורת (בקשות, רשימות קבצים, קבצים וכו')עוברת דרך השרת – סוקט יחיד | טיפול בסוקט יחיד – פשוט | יוצר תהליך סדרתי שמעכב את כל התהליכים  נדרש מיון של המידע |
| 2 | הפרדה:  סוקט ראשי  סוקט קבצים | הפרדה בין הקבצים ושאר המידע  פחות איטי | טיפול יותר מורכב  בסוקט הראשי עדיין נדרש מיון של המידע שעובר |
| 3 | הפרדה:  סוקט ראשי  סוקט קבצים  סוקט ניטור | הפרדה בין הקבצים, לבקשות שונות ולמידע על רשימת הקבצים במערכת  אין עיכובים בין סוגי מידע שונים | יותר סוקטים – יותר עבודה  עדיין מכיוון שקיים רק סוקט יחיד להעלאת קבצים – בצד שרת יהיה עיכוב |
| 4 | הפרדה:  סוקט ראשי  סוקט קבצים – לכל לקוח עבור העלאת קבצים  סוקט ניטור | הפרדה בין הקבצים, לבקשות שונות ולמידע על רשימת הקבצים במערכת  אין עיכובים בין סוגי מידע שונים  אין תלות בין לקוחות שמעלים קבצים לשרת | יותר סוקטים – יותר עבודה |

בפרויקט שלי בחרתי בחלופה מספר 4 – בפרויקט שלי נושא המהירות הוא חשוב (תשתית להורדה מהירה של קבצים) ולכן זוהי חלופה נבחרת, ובכתיבה נכונה ברבה סוקטים זו לא בעיה (כפי שניתן לראות בשרטוט הכולל בסעיף 3)

סוגיה 2 – דרך העברת הקבצים

| חלופה | תיאור | יתרונות | חסרונות |
| --- | --- | --- | --- |
| 1 | התקשורת שדרכה עוברים הקבצים היא השרת, לקוח שרוצה להוריד קובץ מבקש מהשרת את הקובץ והשרת שולח אחרי שביקש מהלקוח האחר | השרת מהווה סמכות ומבטיח שהקובץ המבוקש אכן יגיע ממי שנדרש | איטי ומעמיס על השרת ויותר מזה לא יאפשר הורדת קבצים מהירה  השרת יהווה צוואר בקבוק |
| 2 | בעזרת P2P בין לקוחות לאחר קבלת קובץ הtorrent מהשרת | מהירות ועבודה יעילה מוריד הרבה עומס מהשרת | נדרשת דרך לאמת את הקובץ |

בפרויקט שלי בחרתי בחלופה 2- בפרויקט שלי חושב המהירות והעבודה היעילה והורדת הצוואר בקבוק מהשרת, בנוסף לזה יש לי דרך לאמת את הקובץ בעזרת השימוש בhash

סוגיה 3 – ניהול רשימת הקבצים במערכת (מערכת ניטור)

| חלופה | תיאור | יתרונות | חסרונות |
| --- | --- | --- | --- |
| 1 | השרת מבקש כל זמן מסוים את הרשימה מהלקוחות | מאפשר שליחה של כל השינויים שהתרחשו בפרק זמן מסוים במקום שליחה של כל אחד בנפרד | קבלת העדכון ברגע בקשת השרת ולא ברגע השינוי |
| 2 | הלקוחות מנטרים את תיקיית הקבצים ושולחים כל שינוי לשרת | שם את עבודת העדכון על הלקוחות ומוריד לחץ מהשרת  קבלת הודעה על העדכון באותו רגע של העדכון | מוסיף עבודה לכל לקוח (יצירת הניטור) |

בפרויקט שלי בחרתי בחלופה 2 – בפרויקט שלי חשוב המהירות של עדכון הלקוחות לגבי רשימת הקבצים האפשריים להורדה, חלופה 2 נותנת את הדרישה

סוגיה 4 – מימוש ה Hash בפרויקט

| חלופה | תיאור | יתרונות | חסרונות |
| --- | --- | --- | --- |
| 1 | חישוב ובדיקת hash כולל לקובץ | חוסך בביצוע hash | במידה והקובץ אינו תקין הלקוח יגלה על כך רק בסיום החיבור ולכן הפתרון יכול להיות מאוד לא יעיל מהיר |
| 2 | חישוב ובדיקה hash לכל חלק בקובץ | חוסך בביצוע hash כולל, חוסך חיבור של כל הקובץ במידה וחלק אחד לא תואם | חישוב hash של חלק קטן יכול לצאת שווה לחלק אחר, לכן יכול להיות הצגה של חלק קובץ תקין כלא תקין וההפך, מה שיפגע בכל הקובץ |
| 3 | חישוב ובדיקת hash כולל וגם חישוב ובדיקת hash לכל חלק | חוסך חיבור של כל הקובץ במידה וחלק אחד לא תואם, מאפשר בדיקה נוספת ווידוי נוסף בסיום חיבור הקובץ | דורש הרבה חישוב ובדיקות hash |

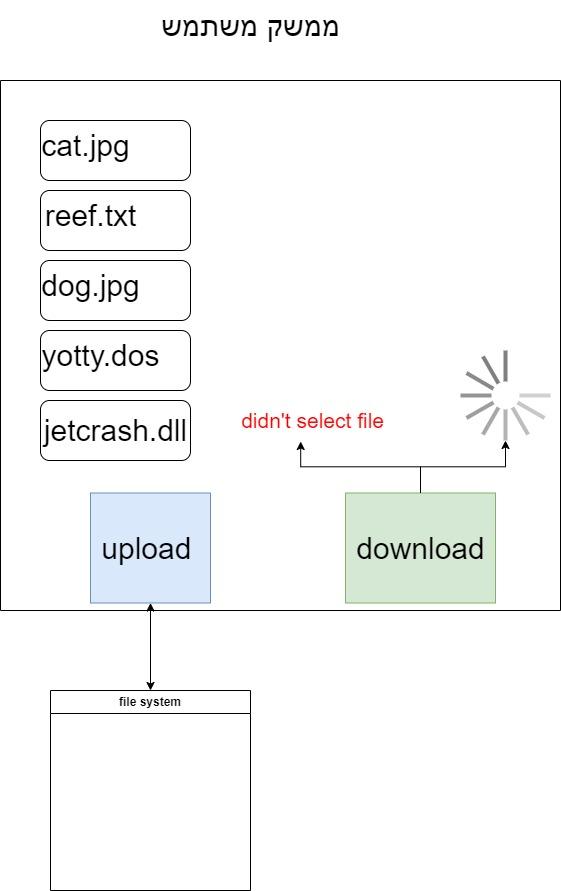
בפרויקט שלי בחרתי בחלופה 3 – בפרויקט שלי יש דגש על ווידוי הגעה של הקובץ הנכון ללקוח בנוסף לזה חשוב לי לייעל את הלקוח ולחסוך ממנו עבודה מיותרת, חלופה 3 נותנת את הפתרון המתאים ביותר.

סוגיה 5 – סוג ההצפנה בפרויקט

| חלופה | תיאור | יתרונות | חסרונות |
| --- | --- | --- | --- |
| 1 | שימוש בהצפנה סימטרית | נותן מהירות בהצפנה והפענוח | המפתח עובר ברשת |
| 2 | שימוש בהצפנה א-סימטרית | המפתח לא עובר ברשת | איטי יותר |
| 3 | שילוב של ההצפנות שימוש בדיפי הלמן לקביעת המפתח מבלי להעביר את המפתח עצמו ברשת, הצפנה סימטרית עם המפתח שנקבע | המפתח לא עובר ברשת, נותן מהירות | דורש ביצוע של 2 חלקים |

בפרויקט שלי בחרתי בחלופה 3 – בפרויקט שלי יש גם דגש על ביטחון המידע וגם על מהירות, בעזרת חלופה 3 אני גם לא צריך להעביר את המפתח ברשת מה שמשפר את ביטחון המידע וגם אני משתמש בהצפנה סימטרית כדי לאפשר מהירות.

1. תיאור מסכי הפרויקט:
   * מה מטרת המסך – להציג למשתמש את הקבצים שהוא יכול להוריד ולאפשר לו לעלות קבצים למערכת



כאשר נלחץ כפתור העלאה נפתחת מערכת הקבצים והמשתמש מתבקש לבחור קובץ

כאשר נלחץ כפתור ההורדה:

* במידה ולא נבחר קובץ נשלחת הודעת שגיאה
* במידה וכן נבחר קובץ נפתחת הצגה של טעינה של הורדת הקובץ

1. תיאור פרוטוקול התקשורת

**פרוטוקול סוקט תקשורת רגילה**

אורך הודעה (מוצפנה) – 4 בתים

הודעה מוצפנת

| הודעה | מ- | אל | opcode | שדות | דוגמא |
| --- | --- | --- | --- | --- | --- |
| העברת מפתחות(דיפי הלמן) | לקוח | שרת | 00 | A-מספר | 0012341321 |
| העברת מפתחות (דיפי הלמן) | שרת | לקוח | 00 | B-מספר | 0012367521 |
| בקשת הקובץ torrent | לקוח | שרת | 01 | שם קובץ | 01cat.jpg |
| תשובת בקשת קובץ torrent | שרת | לקוח | 01 | 0 – שגוי  או  1  ואת הקובץ  קובץ ה torrent | 010  או  011b'0x1a4b 1cd5 ff23' |
| בקשת העלאה | לקוח | שרת | 02 | שם קובץ | 02Cat.jpg |
| תשובה לבקשה העלאה | שרת | לקוח | 02 | 0 – קיים  או  1 - שם הקובץ פורט 4 ספרות | 020  0211500Cat.jpg |
| עדכון רשימת הקבצים | שרת | לקוח | 03 | שמות הקבצים | 03Cat.jpg$%$dog.jpg |

**פרוטוקול סוקט ניטור:**

אורך הודעה(מוצפנת) 2 בתים

הודעה מוצפנת

| הודעה | מ- | אל | opcode | שדות | דוגמא |
| --- | --- | --- | --- | --- | --- |
| נוסף קובץ | לקוח | שרת | 01 | שם קובץ | 01cat.jpg |
| נמחק קובץ | לקוח | שרת | 02 | שם הקובץ | 02cat.jpg |
| השתנה קובץ | לקוח | שרת | 03 | שם הקובץ | 03cat.jpg |
| הוראת מחיקת קובץ | שרת | לקוח | 03 | שם קובץ | 03Cat.jpg |
| העברת מפתחות(דיפי הלמן) | לקוח | שרת | 00 | A-מספר | 0012341321 |
| העברת מפתחות (דיפי הלמן) | שרת | לקוח | 00 | B-מספר | 0012367521 |

**פרוטוקול סוקט העלאה:**

אורך הודעה(מוצפנת) 8 בתים

הודעה מוצפנת

| הודעה | מ- | אל | opcode | שדות | דוגמא |
| --- | --- | --- | --- | --- | --- |
| העלאת קובץ | לקוח | שרת | 01 | שם קובץ  $%$  הקובץ | cat.jpg$%$b'0xaf43'01 |
| תשובה לשליחה | שרת | לקוח | 01 | 1 – עבר | 011 |

**פרוטוקול סוקט לקוח הורדה:**

אורך הודעה(מוצפנת) 2 בתים

הודעה מוצפנת

| הודעה | מ- | אל | opcode | שדות | דוגמא |
| --- | --- | --- | --- | --- | --- |
| העברת מפתחות  (דיפי הלמן) | לקוח | לקוח | 00 | B-מספר | 0012367521 |
| בקשת חלק קובץ | לקוח | לקוח | 01 | מספר חלק - 3 ספרות  שם קובץ | cat.jpg01007 |

**פרוטוקול סוקט שרת הורדה:**

אורך הודעה(מוצפנת) 4 בתים

הודעה מוצפנת

| הודעה | מ- | אל | opcode | שדות | דוגמא |
| --- | --- | --- | --- | --- | --- |
| העברת מפתחות  (דיפי הלמן) | לקוח | לקוח | 00 | A-מספר | 0012341321 |
| שליחת חלק קובץ | לקוח | לקוח | 01 | מספר החלק – 3 ספרות  החלק עצמו | B'0xa0f2 55fa' 01007 |

8. תיאור מבני הנתונים

**שרת**

בסיס נתונים – nexusDB

טבלה torrentTBL - מכילה שמות של קבצים הtorrent הקיימים (שם הקובץ הוא המפתח – לא יתכנו שני קבצים עם אותו השם)

| Names |
| --- |
| STRING עד 30 |
| Cat.jpg.torrent |
| Clients.py.torrent |
| Imri.txt.torrent |

תיקיית קבצים – Torrent files

מכילה את קבצי הtorrent

רשימת לקוחות מחוברים-open\_sockets\_clients

שומר socket – ip, crypt\_objectלכל לקוח שמחובר למערכת

רשימת קבצים – open\_file\_names

שומר את שמות הקבצים השמורים אצל הלקוחות המחוברים כרגע למערכת, מתעדכן כל רגע נתון

**מבנה קובץ Torrent - Json**

| **דוגמא** | **value** | **Key** | **הסבר** |
| --- | --- | --- | --- |
| Cat.jpg | String | File name | השם של הקובץ |
| ["192.168.4.97","192.168.4.96"] | List of strings | Open IP's | כל הip של הלקוחות שיש להם את הקובץ |
| 25600000 | Int | Length of file (bytes) | האורך של הקובץ בבתים |
| 1000 | Int | Number of pieces | מספר החלקים שיש לקובץ |
| ["0xa878b765a","0x1626ab71ab7"] | String | Hash of each piece in order | הhash של כל לקוח |
| 0x123b1a1b3c131"" | String | Full hash | הhash של כל הקובץ |

**לקוח**

תיקייה קבצים מנוטרת – Nexus(hidden)

מכילה קבצים שהמשתמש מאפשר לשתף עם משתמשים אחרים

רשימת קבצים – open\_files\_name

שומר את שמות הקבצים השמורים אצל הלקוחות המחוברים כרגע למערכת, מתעדכן כל רגע נתון

9. סקירת חולשות והאיומים

|  |  |  |
| --- | --- | --- |
| האיום | תיאור | פתרון |
| DOS | לקוח יחיד מתחבר לשרת ומתנתק מספר פעמיים גדול מספיק בשביל לתקוע את השרת | השרת אינו יאפשר התחברות מאותו הip מספר בלתי מוגבל של פעמיים |
| DDOS | לקוחות רבים מתחברים לשרת ומתנתקים | הפרויקט אינו מציג פתרון |
| SQL injection | כתיבת קלט שהוא שורות קוד לתוך הבסיס נתונים | כתיבת שאילתה נכונה, שלא מאפשרת הכנסת קוד זדוני |
| Man in the middle | אדם עם כוונות רעות מאזין להודעות העוברות ומתחיל להתחזות לאחד מהגורמים המתקשרים | כל התקשורת מוצפנת, גם אם האדם מאזין הוא אינו יכול להבין מה משמעות ההודעות, שימוש בפרוטוקול tcp בשכבת התעבורה עם לחיצת יד משולשת |
| קובץ לא תקין | שליחה של לקוח ללקוח אחר קובץ שאינו תקין | השוואת הhash של חלק הקובץ ושל הקובץ הכולל עם הhash המאומת על ידי השרת |
| וירוס | העלאת קובץ שמכיל וירוס | הפרויקט אינו בודק את תוכן הקובץ |

**פרק ה' - 'שם המערכת' - הקוד**

1. עבור (שעושים קטעי קוד מיוחדים משהו מיוחד, משהו מסובך, משהו בדרך שונה, משהו יפה בעיניכם ויש לכלול בתוכם את קטעי הקוד הרלוונטים לאלגוריתמים המרכזיים שציינתם בפרק הקודם):

* הסבר על היכולת
* הקוד עצמו (כתוב ע"פ כללי התכנות הנכון ומלווה בהערות כנדרש)
* (למצוא לפחות 7 קטעי קוד מעניינים ...)
* רצוי תמיד לבחור את אלא שהסברתם קודם לכן את האלגוריתם שלהם

1. בנוסף יש לצרף את **שאר קוד המערכת** , לא נדרש הסבר לצד כל קוד אבל כן את שם הקובץ.  
   לבצע 'שמירה' של הקוד לקבצי PDF אן העתקה לתוך קבצי word **ולא לצרף כתמונה**

**פרק ו' - 'שם המערכת' – בדיקות ('קופסא לבנה')**

1. פירוט הבדיקות

* פירוט הבדיקות שהופיעו בפרק האפיון ברמת הביטים – לתאר בפירוט מה נדרש לבצע ומה בוצע בפועל. וכמובן על בדיקה צריכה להתבצע יותר מפעם אחת ועד שהיא עוברת

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **שם הבדיקה** | **מטרת הבדיקה** | **מה נדרש לבצע** | **מתי** | **מה בוצע בפועל** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

* יש לתעד כל בדיקה שמבוצעת, חייבים לחזור לפחות פעמיים על כל בדיקה – כל חזרה תתועד בשורה נפרדת עם תוצאות הבדיקה

**פרק ז' - 'שם המערכת' – מדריך למשתמש**

1. מדריך למשתמש הכולל עבור כל תהליך/יכולת במערכת:

* הסבר על התהליך/יכולת
* תרשים זרימה (במקרה של תהליך)
* צילומי מסך מלווים בהסברים בשפה פשוטה

במידה ויש יותר ממשתמש אחד למערכת , לבנות פרקים שונים לדוגמא: מדריך למשתמש קצה, מדריך למנהל מערכת וכו'

**פרק ח' - 'שם המערכת' – רפלקציה**

1. מבט אישי על העבודה ועל תהליך פיתוחה:

* אתגרים שהיו לי בדרך
* אירועים מעניינים שקרו במהלך הפיתוח
* התמודדות עם קשיים, איך מה עשיתי
* הערכת הפתרון לעומת התכנון והמלצות לשיפורו
* תודות חשוב מאוד לחברים, למשפה, למורים...

יש למלא עד עמוד ולא להתבייש בכתיבה – כאן לא המקום לצניעות.

**פרק ט' - 'שם המערכת' – ביבליוגרפיה**

1. רקע תיאורטי
2. ספרות מקצועית ספציפית לנושא העבודה (רשימת ספרים, מאמרים והפנייה לכתובות אתרים המכילים חומר רקע ששימש לפיתוח העבודה)
3. הערה: לא מספיק להתייחס לספרים המיועדים ללימוד שפה או כלים יישומיים ומדריכים למיניהם.
4. קישורים לאתרים באינטנרט – קישורים ל stackOverflow עם הסבר על הנושא שנחקר (לא תחביר...)

**נספח א – החקר שבוצע בפרויקט**