תקשורת מחשוב

פרויקט גמר

הוראות להגשת המטלה:

1. **הגשה ביחידים או בזוגות.**
2. שם קובץ ההגשה (מקווץ) חייב להכיל את ת"ז של הסטודנטים.
3. שימו לב, חובה שהקוד ירוץ בכל מחשב ולכן כל החלטה מקומית (**path**) חייבת להיות כללית.
4. כתיבת קוד נכונה כולל שמות משתנים, אובייקטים ופונקציות קטנות שצריך, אין מספרי קסם**.**
5. חלק נכבד מהציון (לפחות 40%) ינתן על כתיבת קוד נכונה, מבנה, בדיקות וכו
6. יש לכווץ את כל הקבצים באחד מהפורמטים הבאים: ZIP, RAR, 7-ZIP בלבד.
   a. יש להגיש מסמך pdf שמסביר מה עשיתם.
   b. יש להגיש קובץ הקלטה של התעבורה שחלקים ממנו יפורטו במסמך ה-pdf
   c. חובה להשתמש ב-Unit testing .
   d. חלקי הקובץ גדולים מידי, אז תעלו לענן/גיט או כל פתרון אחר ותצרפו לינק למסמך ה-pdf
7. עליכם להתייחס למסמך זה כאפיון המערכת. חובה עליכם להתייחס למקרי קצה ולטיפול בבאגים גם אם לא מופיע במסמך באופן מדויק מה המקרה. אי טיפול בבאגים ובמקרי קצה יגרור הורדה של ניקוד.
8. אופן/צורת/אלגוריתם שתבחרו (לדוגמא RDT, congestion control) יהווה חלק מהציון. אל תנסו לשאול אותנו אם משהו מספיק אלא על פי מורכבות הפתרון ינתן הציון.
9. אסור להעתיק. זאת אומרת:

- מותר לדבר אחד עם השני בנוגע למטלה, להתייעץ איך כדאי לממש, ולשתף בבעיות שצצות, כל זמן שזה נעשה בע"פ. אסור לעזור ע"י העברה של קטע קוד כלשהו מאחד לשני , אפילו לא פונקציה אחת. במידה ומתגלית העתקה 2 הצדדים יקבלו 0 במטלה וייכשלו בקורס, ללא תלות במי העתיק ממי.
- מותר להיעזר באינטרנט, אבל אסור להעתיק קוד שמצאתם כמות שהוא - כתבו את הקוד בעצמכם. בפרט, יש איסור להעתיק קוד מ- github .
- מי שעובד עם github חייב להגדיר repository private .
- מותר להיעזר בחונך או במורה פרטי, אבל אסור שהם יכתבו לכם את הקוד או חלקו

תיאור התרגיל: שפת תיכנות היא ***Python***

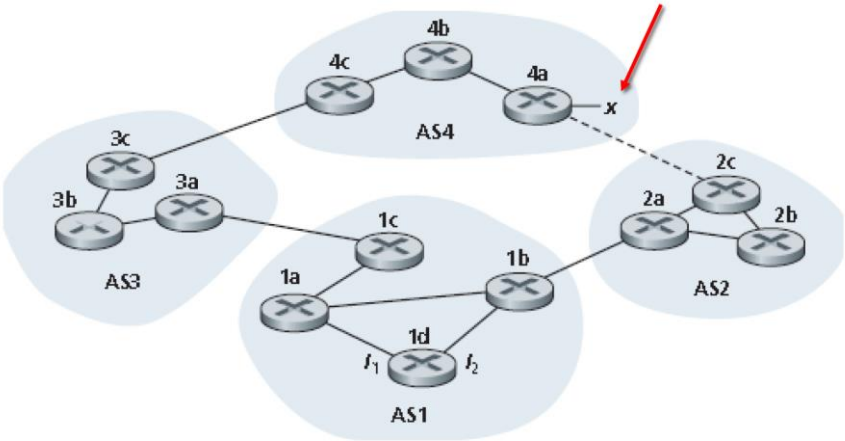
חלק א' – **על בסיס תיאור המערכת:**
עליכם לבנות מערכת מסרים מיידים פרימיטיבית (בדומה ל- messenger) מבוססת על תקשורת .

חלק ב'– **על בסיס תיאור המערכת:**
להוסיף לאותה מערכת שכבה חדשה (קוד נוסף) להעברת קבצים מעל UDP אשר נקרא לה FAST reliable UDP
ענו על השאלות הבאות:

- ציירו דיאגרמת מצבים בהם המערכת עובדת
- כיצד המערכת מתגברת על איבוד חבילות
- כיצד המערכת מתגברת על בעיות latency

חלק ג': ענו על השאלות הבאות ללא קשר לחלקים הקודמים. החלק הזה עומד בפני עצמו:

1. בהינתן מחשב חדש המתחבר לרשת אנא תארו את כל ההודעות שעוברות החל מהחיבור הראשוני ל switch ועד שההודעה מתקבלת בצד השני של הצאט. אנא פרטו לפי הפורמט הבא:
   a. סוג הודעה, פירוט הודעה והשדות הבאים
      i. כתובת IP מקור/יעד, כתובת פורט מקור/יעד, כתובת MAC מקור/יעד, פרוטוקול שכבת התעבורה.
2. הסבירו מה זה CRC
3. מה ההבדל בין http 1.0 ,http 1.1, http 2.0, QUIC
4. למה צריך מספרי port?
5. מה זה subnet ולמה צריך את זה?
6. למה צריך כתובות mac למה לא מספיק לעבוד עם כתובות ip?
7. מה ההבדל בין Router Switch Nat?
8. שיטות להתגבר על המחסור בIPv4 ולפרט?
9. נתונה הרשת הבאה.
   a. AS2, AS3 מריצים OSPF
   b. AS1, AS4 מריצים RIP
   c. בין ה-Ass רץ BGP
   d. אין חיבור פיזי בין AS2, AS4
   e. בעזרת איזה פרוטוקול לומד הנתב 3c על תת רשת x
   f. בעזרת איזה פרוטוקול לומד הנתב 3a על תת רשת x
   g. בעזרת איזה פרוטוקול לומד הנתב 1c על תת רשת x
   h. בעזרת איזה פרוטוקול לומד הנתב 2c על תת רשת x



תיאור המערכת:

המערכת תהיה בנויה משרת ולקוחות כאשר:

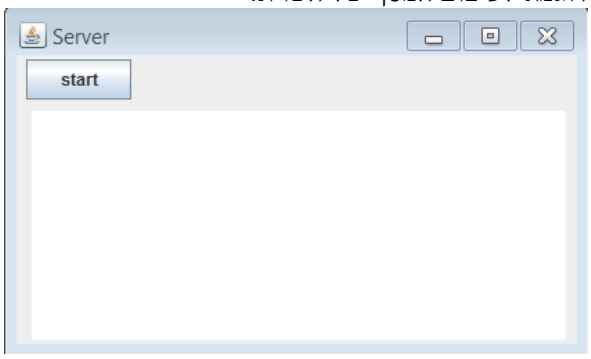
השרת (server) מאתחל את עצמו ו"מקשיב". השרת רץ על גבי שרת IP מסוים ומקשיב ללקוחות בפורט מסוים הידוע ללקוחות לדוגמה 50000, כאשר הוא מאפשר להתחבר למספר לקוחות בו זמנית (לפחות 5). בשביל הסדר הטוב נסכם שבתרגיל זה נעשה בצד של השרת שימוש בפורטים שמספרים 55000-55015. לאחר שלקוח מתנתק הפורט שהתפנה חוזר להיות משאב פנוי.

כל לקוח יכול לשלוח הודעה לכל מחוברים ב-chat והודעה פרטית למחובר ספציפי תוך ניטור ההעברה באמצעות קשר TCP. כל לקוח יכול ליזום בקשה להורדת קובץ מהשרת באמצעות קשר TCP במקביל לערוץ העברת קובץ ע"י קשר UDP שימו לב שהעברת הקובץ צריכה להיות אמינה ולקחת בחשבון עומס ברשת ולכן אתם צריכים לממשהי FAST reliable UDP (במילים אחרות מימוש RDT מעל חיבור UDP בתוספת Congestion control). בעת סיום, יש להוציא הודעה מתאימה, בצרוף ערך הבית האחרון שנשלח. שימו לב שניתן להוריד בו זמנית הקובץ לכמה לקוחות שונים. כלומר, יתכן שבעת המתנה לקבלת אישור להמשך הורדה מלקוח אחד, הלקוח האחר מבקש ומתחיל לקבל את הקובץ במקביל.

שימו לב, מטרת התרגיל היא הצד התקשורתי ולכן אין חובה לפתח את ה-GUI. קבוצות שיפתחו GUI משלהם או על בסיס הדוגמאות בתרגיל הזה יקבלו בונוס.

שימו לב מספר 2, כל הפיתוח צריך להיות שלכם ולא להתבסס על קוד פתוח מהרשת.

דוגמא לעיצוב המסך של השרת:

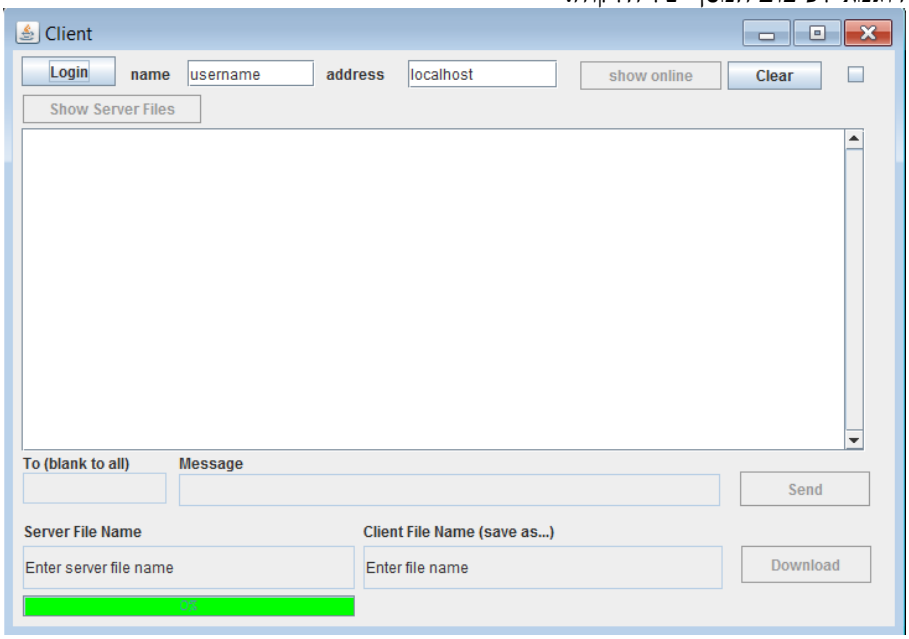


הלקוח (client) יכול לבצע את הפעולות הבאות:

1) להתחבר לשרת,
2) להתנתק מהשרת,
3) לשלוח הודעה ללקוח אחר,
4) לשלוח הודעה לכל הלקוחות המחוברים לשרת כעת,
5) לקבל את שמות הלקוחות המחוברים לשרת,
6) לקבל רשימת קבצים הקיימים בשרת,
7) לשלוח בקשה להורדת קובץ מהשרת,
8) להוריד קובץ מתוך השרת.

כמו כן, במידה והצטרף לקוח חדש או התנתק לקוח קיים יש לשלוח הודעה מתאימה לכל המשתתפים.

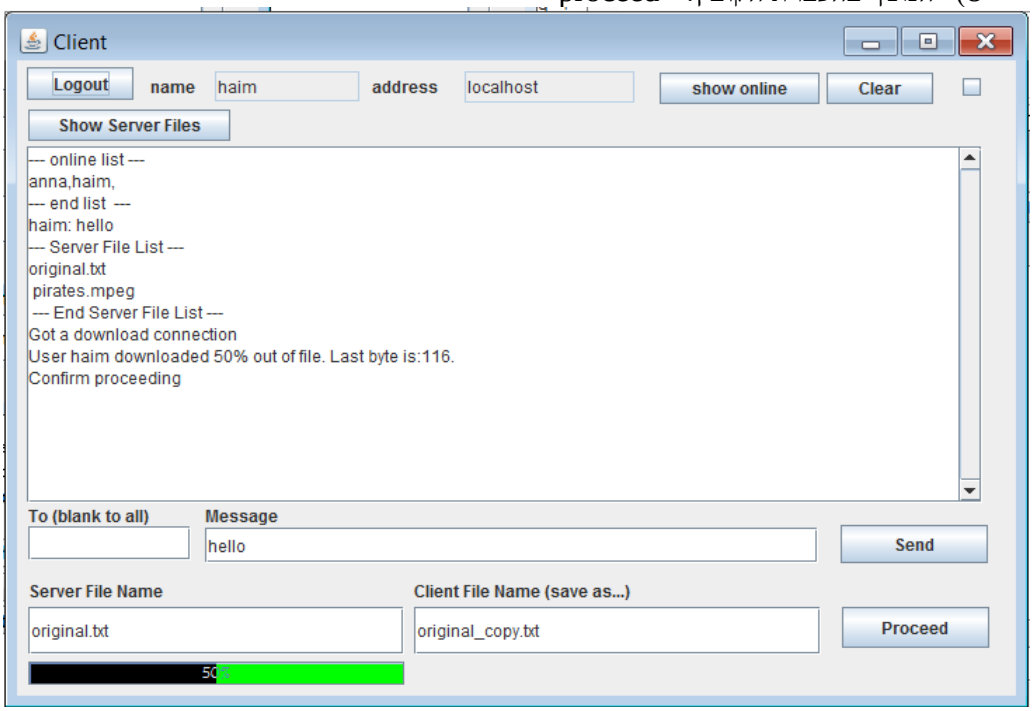
דוגמא לעיצוב המסך של הלקוח:



מומלץ להגדיר פרוטוקול (שפה משותפת) עבור הודעות בין השרת ללקוח. לדוגמה:

רשימת הודעות שנשלחות מהלקוח (לשרת):

1) אני רוצה להתחבר: <connect><name>
2) אני רוצה לקבל את רשימת המחוברים: <get_users>
3) אני רוצה להתנתק: <disconnect>
4) אני רוצה לשלוח הודעה ללקוח: <set_msg><Name>
5) אני רוצה לשלוח הודעה לכולם: <set_msg_all>
6) אני רוצה לקבל את רשימת הקבצים שיש בשרת: <get_list_file>
7) אני רוצה להוריד קובץ: <download><file_name>
8) המשך בהעברת הקובץ: <proceed>



רשימת ההודעות שמתקבלות מהשרת (ללקוח):

1) התחברת <connected>
2) התנתקת <disconnected>
3) ההודעות עבורך <msg_lst><num_of_msgs><”…”>….<”…”><end>
4) רשימת המחוברים <users_lst><num_of_users><”…”>…<”…”><end>
5) רשימת הקבצים <file_lst><”…”>…<”…”><end>
6) קיבלת 100% מהקובץ:

User <name> downloaded 100% out of file. Last byte is: yyy.

הערות כלליות:

- ניתן להניח שכמות ההודעות של כל לקוח לא תעלה על 100 הודעות.
- שימו לב, גודל מקסימלי לחבילה (**datagram**) להעברה באמצעות **UDP** הינו **kB**64. לכן יש להגביל גודל הקובץ להורדה.
- קודם יש לנסות על מחשב אחד גם את הלקוח וגם את שרת. כתובת מקומית של כל מחשב היא 127.0.0.1 או localhost.
- תשובה בסגנון: "לא כתבתם במטלה שצריך לטפל בזה" לא תתקבל.
- אתם אמורים ליצור איבוד חבילות + השהיות על מנת לבדוק את המערכת שלכם

בהצלחה!