**תרגיל תכנות PYTHON  
yaniv hajaj  316411578**

  
  
**PYTHON + Windows10**

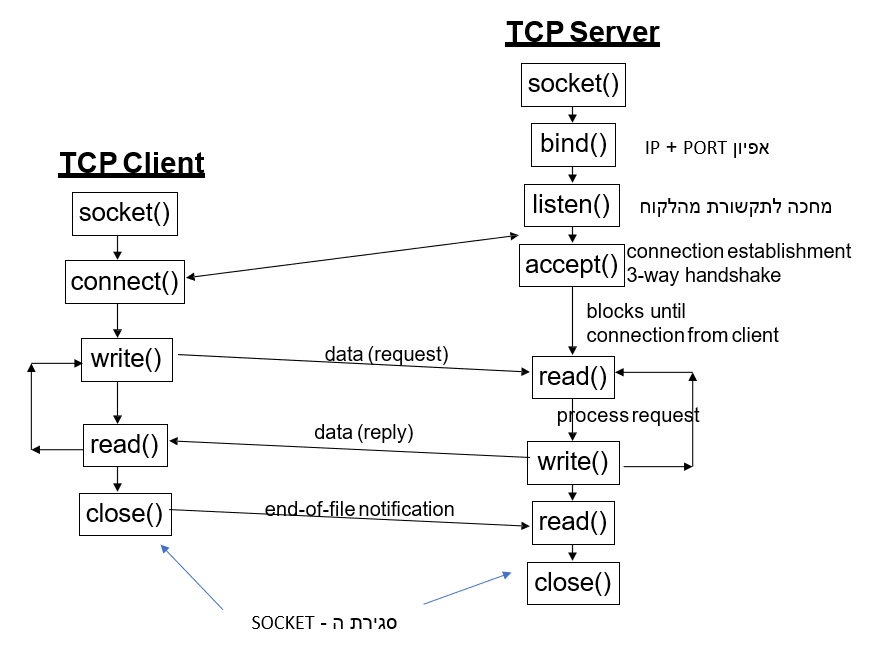
  
רקע תיאורטי:

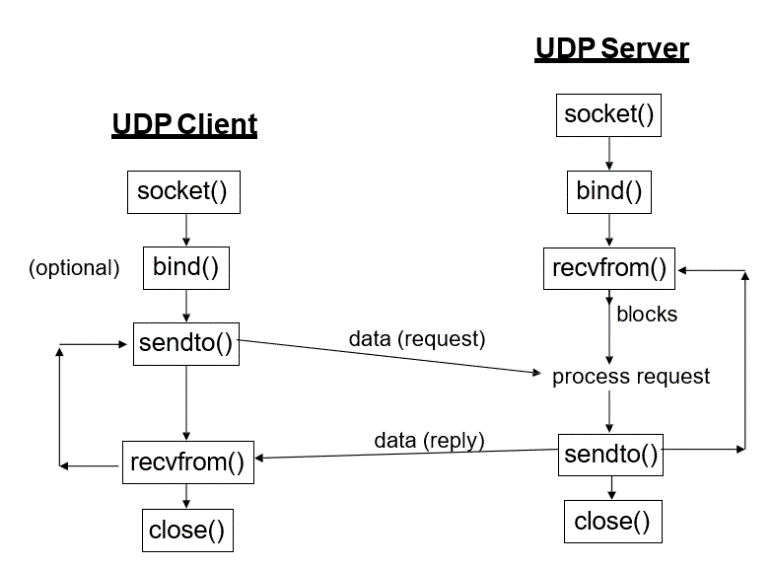
מודל חמש השכבות נועד לספק את ההבנה הבסיסית ביותר לקיום תקשורת תקינה בין כל שני מערכות מחשב. השכבות במודל הן: השכבה הפיזית, שכבת הקו, שכבת הרשת, שכבת

התעבורה ושכבת האפליקציה (סדר השכבות: מהפיזית ועד לאפליקציה).

במטלה זו נבחן לעומק את שכבת התעבורה ואת שכבת האפליקציה.   
שכבת התעבורה: מטרתה לקבוע את יעד ההודעה בתוך המחשב. היא אחראית על  
האופציה לפנות אל מספר שירותים הנמצאים על אותה ישות וכמו כן גם דואגת שהחבילות יגיעו בסדר שנשלחו אל היעד המבוקש.  
בשכבה זו ישנם שני פרוטוקולים עיקריים- פרוטוקול TCP (אמין ואיטי יותר) ופרוטוקול UDP (מהיר אך לא אמין).  
שכבת האפליקציה היא האחראית על התקשורת עם משתמש הקצה והיא שתקבע את  
סוג התקשורת בין מחשבים. כמו כן גם אחראית על תהליך עיבוד הנתונים שהתקבלו וניהול התהליכים.  
במטלה להלן נעבוד במודל "שרת – לקוח" . מודל זה מחלק את המשימות בין השרת, לבין הלקוח.במטלה התבקשנו ליצור צ'אט קבוצתי, השרת הוא למעשה המחשב שאחראי על ניהול הצ'אטים הקבוצתיים.

נראה את אופן פעולת הפרוטוקולים UDP,TCP :





פרוטוקול TCP:

פרוטוקול הנמצא בשכבת התעבורה אשר עובד לפי IP.   
הפרוטוקול אמין (לעומת UDP שאינו אמין), כלומר הוא משתמש בוידוא Windowing כדי לוודא את הגעת החבילות באופן תקין וכדי להגביר את מהירות ההעברה ככל שניתן (בהתאם למהירות הרשת).

התקשורת ברשת מתבצעת בין שרת ששולח מידע ומחשב (לקוח) שמקבל מידע. השולחים והמקבלים ברשת האינטרנט אורזים את המסרים שלהם באמצעות פרוטוקול  **TCP** לחבילות קטנות הנקראות "פאקטות". הפרוטוקול ישלח את הפאקטות באמצעות פרוטוקול **IP**  המנחה את הפאקטה ליעדה.

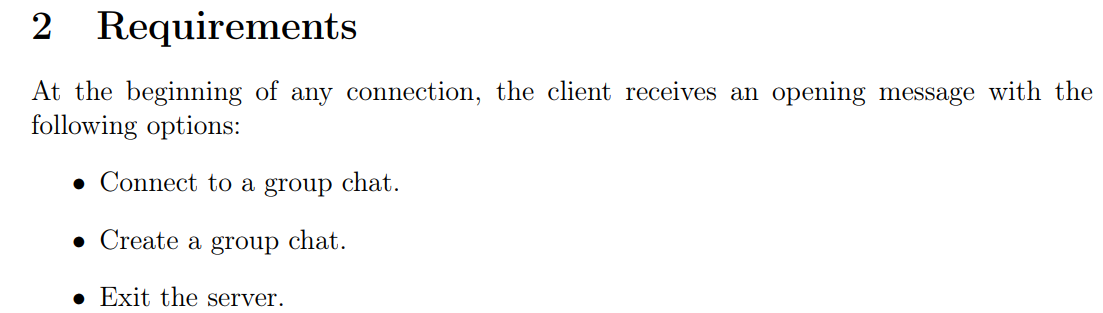
הSOCKET משמש ממשק בין שכבת האפליקציה לשכבת הרשת, בעזרתו ניתן לשלוח/לקבל נתונים ומידע אל הרשת או מהרשת וכך לייצר למעשה תקשורת.

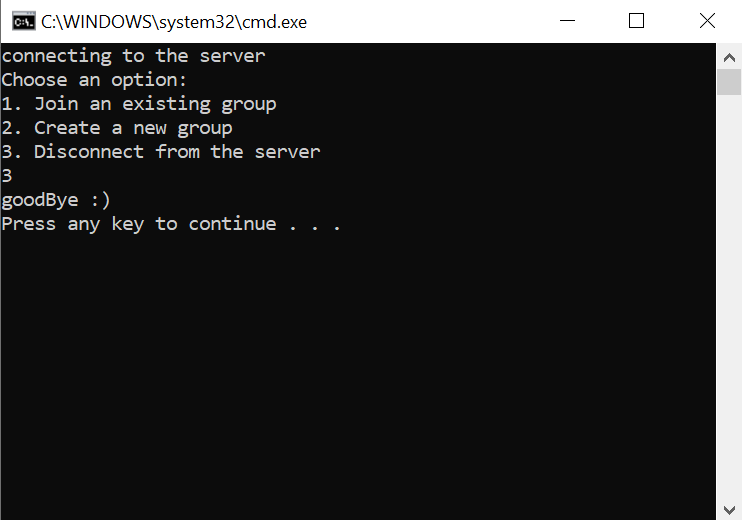
Network socket הוא כמו מבנה וירטואלי- תוכנתי בתקשורת המחשב המשמש כנקודת קצה לשליחת וקבלת נתונים ברחבי הרשת.

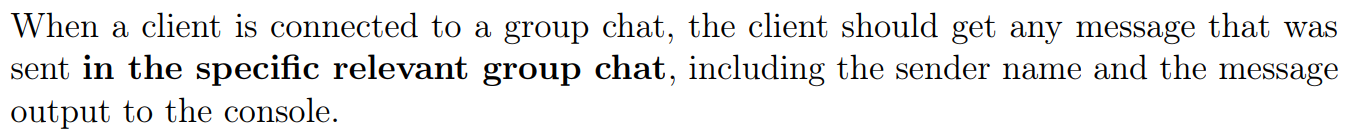
Port הוא מבנה לוגי שמזהה תהליך ספציפי או סוג שירות ברשת. ישנם כמה יציאות ספציפיות המשמשות לזיהוי שירותים ספציפיים, באמצעותם ניתן להעביר חבילות שמגיעה לאפליקציה הנמצאת בהרצה.



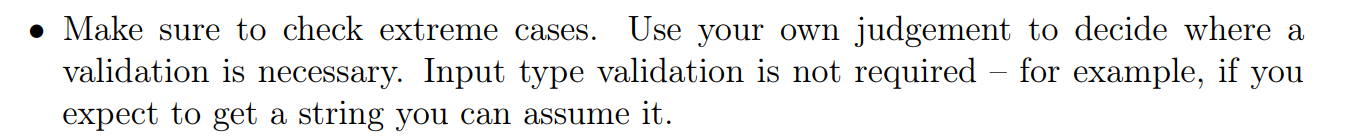
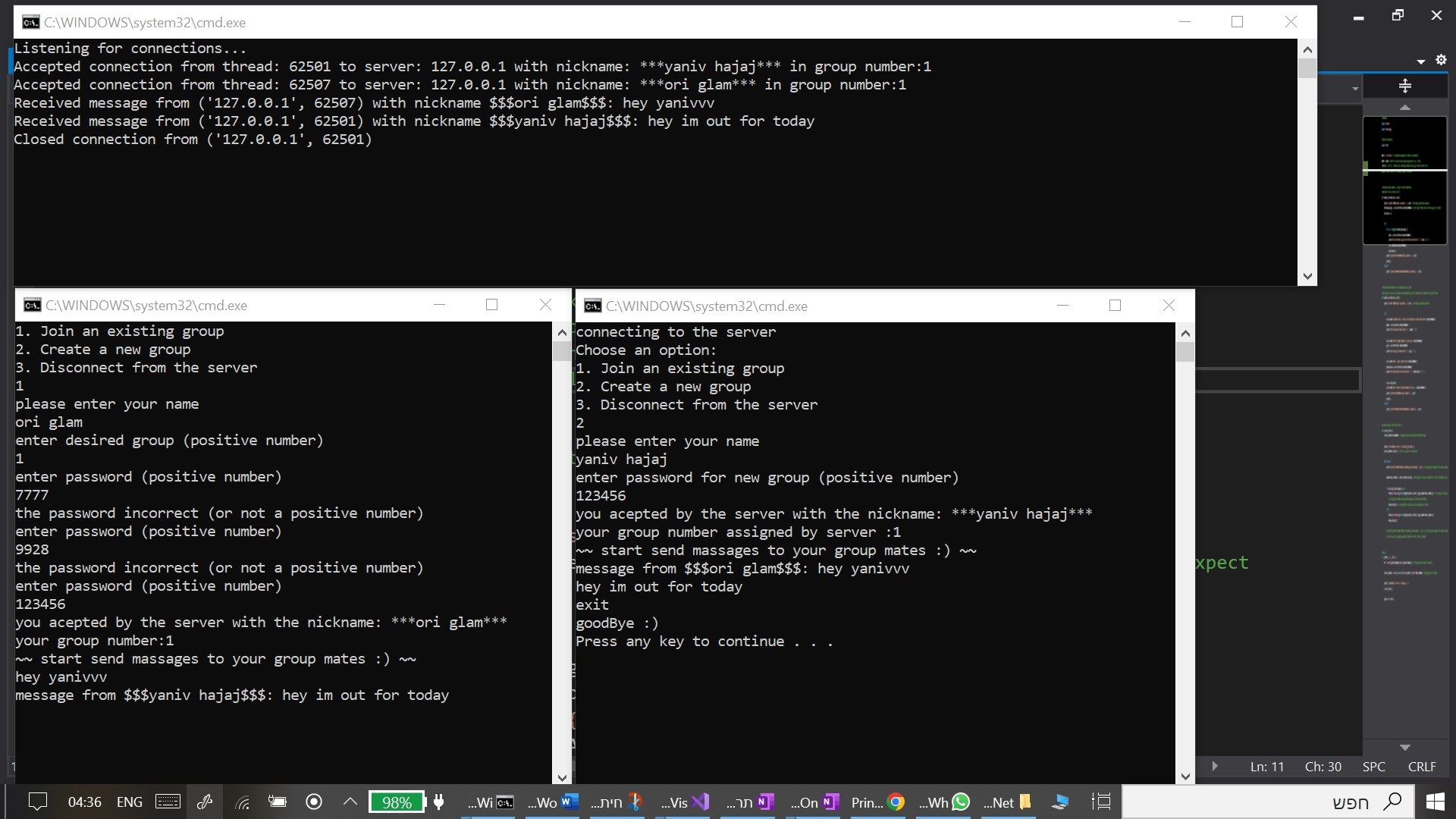
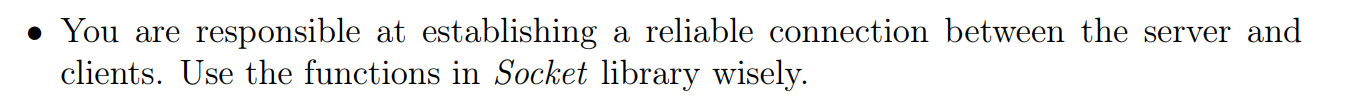
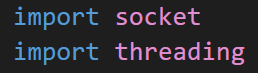
תחילה נפעיל את הסרב שיאזין ללקוחות פוטנציאלים  
 הסרבר מאזין:  
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי  
  
  
לקוח מתחבר ומקבל את המסך הבא:  
תמונה שמכילה טקסט, צילום מסך, אלקטרוניקה, מחשב

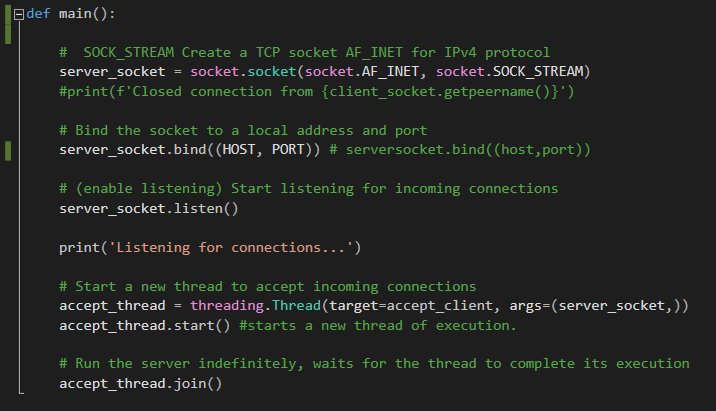
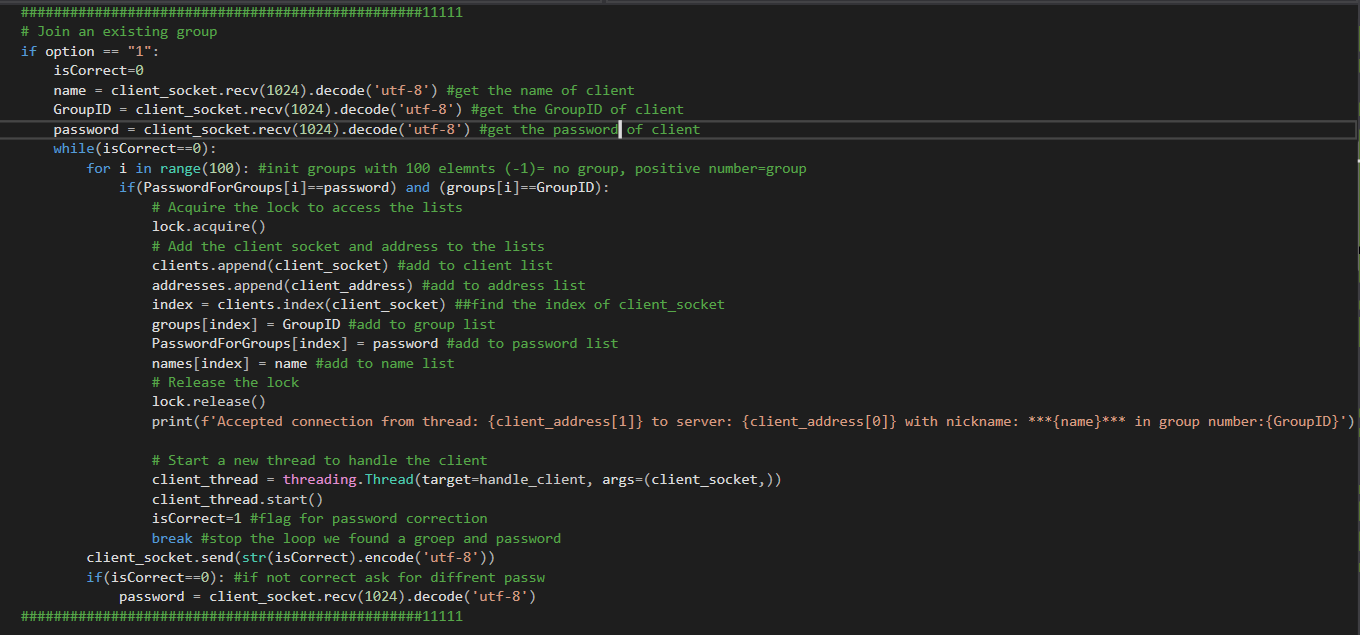
התיאור נוצר באופן אוטומטי  
אם לחץ 3 התנתק מהשרת ישירות  
  
  
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי  
עבור הדגמת אפשרויות אלה ניתן לראות שיחה בין כמה משתמשים:  
בחלון העליון נמצא השרת שמאכסן את פרטיהם של המשתמשים הקיימים ובמקביל מאזין לעוד לקוחות במידה ויתחברו.  
בחלון ימין יש את Yaniv שיצר חדר עם סיסמא 1234 והשרת הקצה לו את חדר מס 1.  
לאחר מכן shlomo התחבר גם הוא לסרבר ויצר חדר חדש עם סיסמא 1111 והשרת הקצה לו את חדר מס 2.  
לאחר מכן ori התחבר ובחר באופציה 1 הזין את חדר מס 1 והסיסמא 1234 והתחבר לחדר של Yaniv.  
  


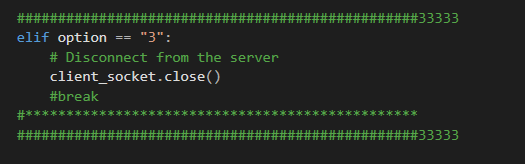
ניתן לראות כי ori+Yaniv מדברים בינהם וההודעות מגיעות אחד לשני, ולעומת זאת הודעות מ shlom לא מגיעות לחדרים של ori+Yaniv שהרי שלומו התחבר לחדר מספר 2 לעומת 1 של שניהם.  
הכל הזמן הזה כל הודעה שעוברת בשרת מאוכסנת על ידו ומודפסת כlog של כל הפעולות (התחברות, הודעה מחדר התנתקות מהשרת וכו)תמונה שמכילה טקסט

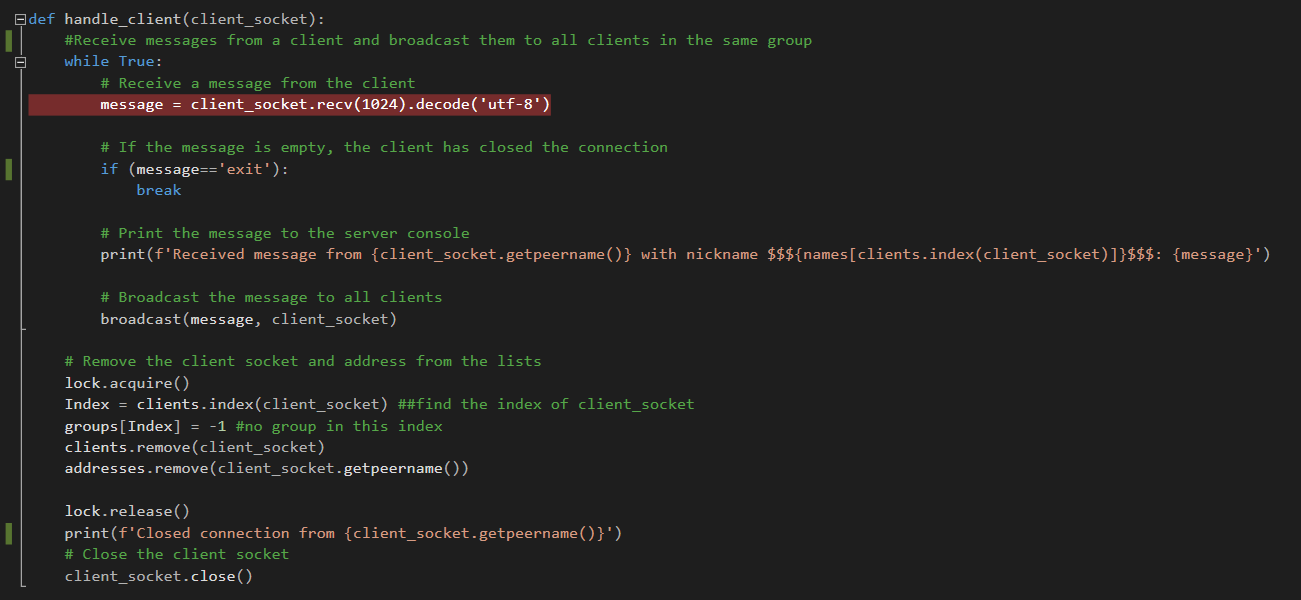
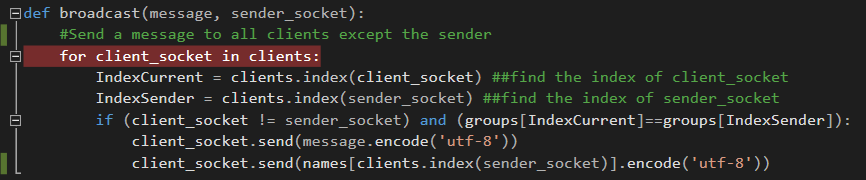
התיאור נוצר באופן אוטומטי  
  
  
כמובן שהכנסנו מקרי קצה שלא הודגשו במטלה כמו לקוח מנסה להתחבר ולא הכניס סיסמא נכונה, יקבל שגיאה ואפשרות לנסות סיסמא חדשה.  
למטה בתמונה יניב פתח חדר עם סיסמא של 1234 וניתן לראות את אורי מנסה להתחבר לחדר מספר 1 עם סיסמא שגויה (7777 או 9928) ומקבל שגיאה עד שלבסוף מכניס את הסיסמא הנכונה (1234) ומתחבר לשרת.  
כמו כן עוד סוג של תוספת\מקרי קצה זה אפשרות להתנתק מהשרת על ידי שליחת ההודעה exit בתוך הצאט  
למשל בתמונה למטה ניתן לראות התכתבות של יניב עם אורי בחדר מספר 1 וכאשר יניב רושם exit הוא מקבל הודעת התנתקות, כמו כן גם בlog של השרת יש הודעה על התנתקות של ה thread של יניב.  
  
  
  
הספריות איתם השתמשנו הן אותן 2 ספריות כמו בתרגול:  


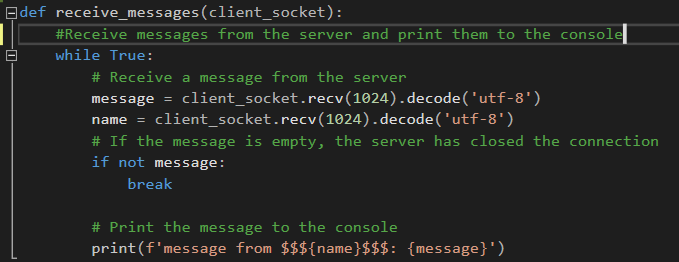


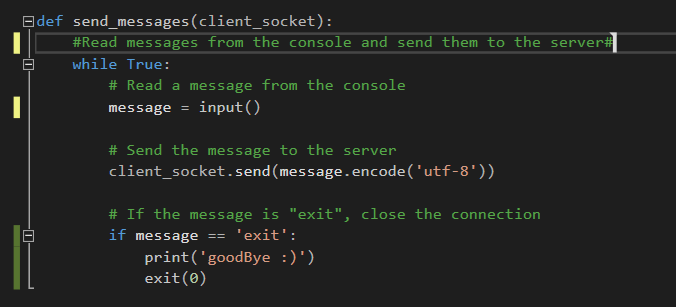
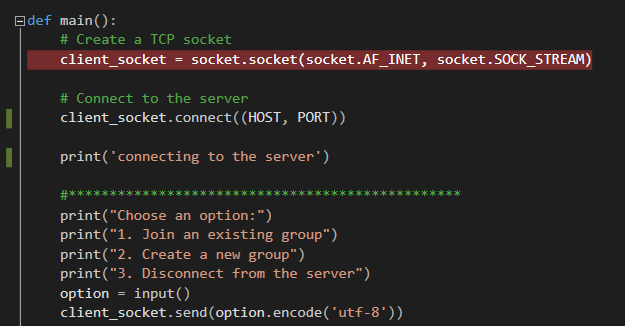
יצרנו שני קבצים תחת אותו פרוייקט, קובץ אחד נקרא server וקובץ שני נקרא client  
הרצנו את הסרבר ראשון שיאזין ואת הלקוחות אחריו.  
הserver מכיל את הפונקציות הבאות:  
קודם כל main אשר מאתחל את השרת וגורם לו להאזין.  
  
  
פונקציה נוספת היא:  
  
כמו כן בתוכה יש את כל ההתעסקות עם האופציות השונות כתלות במה לחץ הלקוח (1,2,3)  
הפוקנציה מקבלת את הלקוח לסרבר או לא אותו במקרה והוא לחץ 3.  
הלקוח מקבל ממנה פידבק בהתאם למה שהוא לוחץ, למשל אם לחץ סיסמא לא נכונה יוכל לנסות שוב. בהנחה שהכל תקין ןהמשתנה flag=1 הסרבר מחבר את הלקוח לשרת, אל החלון של הלקוח מודפס שהתחבר לקבוצה שלו.  
  
  
אופציה 1:  


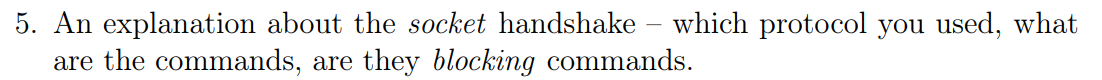
אופציה 2:  
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי  
אופציה 3:  


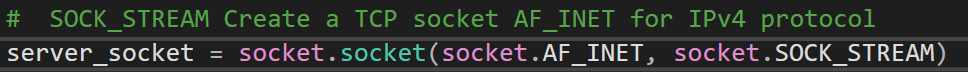
עוד פונקציה היא:  
  
אשר דואגת לטפל בהודעות בלקוחות, התפקיד שלה הוא לקבל הודעה מלקוח שמחובר אליה ולבדוק איזה אנשים נמצאים באותה קבוצה של השולח ולדאוג להפיץ את ההודעה לכל האנשים שבאותה קבוצה חוץ מהשולח. היא עושה את ההפצה באמצעות פונקציה אחרת שבנינו שנקראת broadcast  


הclient מכיל את הפונקציות הבאות:  


עוזרת לקבל הודעות שמשתמשים אחרים שלחו ולהדפיס למסך של המשתמש.  
  
עוזרת לשלוח הודעות למשתמשים אחרים, כמו כן במידה וההודעה היא exit דואגת לצאת .  
  
כמו כן יש את main של הלקוח  
  
שם יש את הטיפול באופציות 1,2,3 בדומה לSERVER רק מצד הלקוח



לחיצת היד של socket היא תהליך של יצירת חיבור בין שני מכשירים באמצעות הספרייה socket. בקוד שלנו, אנו משתמשים בפרוטוקול בקרת שידור (TCP) עבור חיבור socket.  
 TCP הוא פרוטוקול אמין, המספק שירות תקשורת מכוון חיבור, אשר נפרט עליו בהמשך בחלק התיאורתי.  
  
לחיצת היד של socket כוללת את השלבים הבאים:  
הלקוח יוצר שקע socket בקשת חיבור לשרת. זה נעשה באמצעות שיטת connect() בצד הלקוח.  
הלקוח מכוון ל (HOST, PORT) מסויימים במקרה שלנו בחרנו HOST = '127.0.0.1' PORT = 12345  
השרת יוצר socket ומקשר אותו לכתובת ויציאה ספציפיים בשיטת bind() .  
   
לאחר מכן השרת מאזין לחיבורים נכנסים באמצעות שיטת listen() .  
  
כאשר השרת מקבל בקשת חיבור מהלקוח, הוא מקבל את החיבור באמצעות שיטת accept() .   
  
שיטה זו מחזירה שקע חדש המשמש לתקשורת עם הלקוח, כמו גם את כתובת הלקוח.

בקוד שלנו, השימוש ב accept() ו- listen() חוסמות, מה שאומר שהתוכנית תיעצר בנקודות אלו עד לאישור חיבור או קבלת בקשת חיבור, בהתאמה. שיטת connect() היא גם שיטת חסימה, אך לרוב היא תחזור במהירות אלא אם יש בעיה בחיבור.  
לאחר מחקר באינטרנט גילינו כי גם השימוש ב send() ו- recv() יכולות לחסום, בהתאם לדגלים המשמשים בעת יצירת הsocket וכברירת מחדל שיטות אלה חוסמות, אך אפשר להגדיר אותם עם דגל SO\_NONBLOCK כך שהן לא חוסמות.  
  
  
השתמשנו בTCP עם פרוטוקול IPv4 כפי שהוצג בקוד במודל.  
SOCK\_STREAM עבור TCP ו AF\_INET עבור IPv4