מטלת תכנות – צ'ט קבוצתי וירטואלי

רשתות מחשבים ואינטרנט 1 – 83455-01

סמסטר חורף תשפ"ג

מגישים:

איתי אהרון גולדברג - ת"ז:

יונתן קופפר – ת"ז:

הוגש בתאריך: 1.1.23

במטלה זו נתבקשנו לכתוב שני קטעי קוד – שרת ולקוח, כך שניתן יהיה ליצור צ'ט קבוצתי וירטואלי. ביצענו את המטלה בשפת Python, גרסה 3.10.6 במערכת הפעלה Windows. קובץ הZIP מכיל את קובץ הPDF הזה וכן את שני קטעי הקוד: Client.py ו Server.py.

קובץ זה מכיל: רקע תיאורטי, הסבר על הקוד שכתבנו וצילומי מסך המתארים את פעולת האפליקציה.

רקע תיאורטי

רשתות מחשבים הן קבוצה של מחשבים מחוברים שיכולים לתקשר זה עם זה כדי לחלוק משאבים, להחליף מידע ולתקשר. ניתן לסווג רשתות אלו לסוגים שונים בהתבסס על גודלן והיקפן, כגון רשתות מקומיות (LANs), ורשתות רחבות (WANs). רשתות משתמשות בטכנולוגיות, פרוטוקולים ותקנים שונים כדי לאפשר תקשורת בין מכשירים. טכנולוגיות אלו כוללות חיבורים קוויים ואלחוטיים. פרוטוקולים מגדירים את הכללים והסטנדרטים לתקשורת בין מכשירים.

בקוד שכתבנו השתמשנו בספריית Socket ב Python כדי ליצור חיבור רשת בין המכשירים. Sockets הם ממשק תוכנה המאפשר תקשורת בין מכשירים דרך רשת. השרת והלקוח משתמשים בsockets כדי ליצור חיבור ולהחליף נתונים זה עם זה. השרת יוצר server socket, מקשר אותו לHOST וPORT ספציפיים ומאזין לחיבורים נכנסים מלקוחות. הלקוח יוצר client socket ומתחבר לserver socket באמצעות הHOST והPORT שצוינו. לאחר יצירת החיבור, השרת והלקוח יכולים לשלוח ולקבל נתונים זה מזה באמצעות socket של כל אחד מהם.

על מנת שהקוד יוכל להתמודד עם מספר לקוחות במקביל השתמשנו ב threading. threading מאפשר לתוכנית לבצע זרימה נפרדת של ביצוע התוכנית. באופן כזה, הרצת הסרבר אל מול לקוח אחד לא דורסת או מתעכבת בגלל טיפול של הסרבר בלקוח אחר.

הסבר על הקוד

Start_server:

פונקציה זו פותחת סרבר לצ'אט ה"מאזין" לחיבורים של קליינט. כאשר קליינט מתחבר לשרת, הפונקציה יוצרת תהליכון המטפל בבקשה של הקליינט. הפונקציה ממשיכה להקשיב לחיבורים נוספים של קליינטים עד שהסוקט נסגר.

Client_handle:

הפונקציה מקבלת סוקט של קליינט וכתובת ומטפלת בחיבור לשרת. קודם כל, היא מבקשת מהקליינט להזין האם הוא רוצה ליצור קבוצה חדשה או להצטרף לאחת קיימת. אם הקליינט בוחר ליצור קבוצה חדשה, היא קוראת לcreate_new_group ומעבירה את הקליינט סוקט, הכתובת ומספר הקבוצה. לאחר מכן היא יוצרת תהליכון לטפל בצ'אט הקבוצתי עבור הקליינט. אם הקליינט בוחר להצטרף לקבוצה קיימת היא קוראת לjoin_existing_group ומעבירה את הקליינט סוקט והכתובת ואז יוצרת תהליכון לטפל בצ'אט.

Join_existing_group:

פונקציה זו מקבלת קליינט סוקט וכתובת על מנת לנסות לחבר בין הקליינט ובין הקבוצה הקיימת. ראשית, היא מבקשת מהקליינט להזין את שמו ואת מזהה הקבוצה אליה הוא רוצה להצטרף. לאחר מכן, היא מבקשת מהלקוח להזין את הסיסמא עבור הקבוצה. אם הסיסמא שגויה, הלקוח מקבל ניסיונות נוספים להזין סיסמא נכונה. אם לאחר מספר הניסיונות שהוגדר, הסיסמא עדיין שגויה, הפונקציה סוגרת את הקליינט סוקט וחוזרת. אם הסיסמא נכונה, הקליינט מתווסף לקבוצה ומתחיל תהליכון לטיפול בצ'אט עבור הלקוח.

Create_new_group:

פונקציה זו מקבלת קליינט סוקט, כתובת ומספר הקבוצה. היא מבקשת מהלקוח להזין את שמו וליצור סיסמא עבור הקבוצה החדשה. לאחר מכן היא יוצרת קבוצה חדשה עם הסיסמא ועם הקליינט כחבר היחיד בקבוצה. היא מחזירה את מזהה הקבוצה ואת אובייקט הקליינט החדש.

Disconnect_from_server:

פונקציה זו אחראית לניתוק הקליינט מהשרת. היא סוגרת את הקליינט סוקט ומסיימת את התהליכון של שעליו רץ הלקוח.

Handle_group_chat:

פונקציה זו אחראית לטיפול בצ'אט הקבוצתי בין הלקוחות המחוברים לקבוצה. היא מקבלת הודעות מקליינטים ומשדרת אותן לכל שאר הלקוחות בקבוצה. בנוסף, היא מטפלת גם במקרה בו קליינט שולח הודעת 'exit' על מנת לצאת מהקבוצה. במקרה זה, היא מודיעה לכל שאר החברים בקבוצה על עזיבת הקליינט, מסירה את הקליינט מהקבוצה וסוגרת את הקליינט סוקט שלו.

Remove_client_from_group:

פונקציה זו היא פונקציית עזר לפונקציה Handle_group_chat להסרת קליינט מקבוצה כאשר הוא רוצה לעזוב. היא מסירה את הקליינט מרשימת הקליינטים של הקבוצה.

בתוך הפונקציה start_server המתודה server_socket.accept היא blocking command. התוכנית מחכה עד שמתקבל חיבור קליינט חדש על ידי השרת במקום לבדוק כל הזמן מתי יש נסיון ליצור חיבור חלש כל הזמן.

<u>פונקציות קליינט</u>

הקליינט מיוצג באמצעות קלאס Clientwindow המייצגת את הGUI שלו. בתוכו יש מספר פונקציות חוץ מ"__init__" כקונסטרקטור ו"__init_ui__" להציג את הlayout של התצוגה:

Send_message:

פונקציה זו נקראת כאשר נלחץ כפתור הsend או כאשר נלחץ enter בשורת הinput. ההודעה נשלחת דרך השרת.

Receive:

זהו תהליכון המקבל כל הזמן הודעות מהשרת ומציג אותן במסך השיחה.

Start_client:

זוהי פונקציה היוצרת קליינט סוקט ומתחילה שני תהליכונים. אחד בשביל לקבל הודעות ואחד בשביל לשלוח הודעות. השליחה והקבלה מתבצעות באמצעות הפונקציות לעיל.

בתוך הפונקציה receive, מופעלת המתודה: 'client_socket.recv' שהיא blocking command.
הפונקציה מחכה לקבל הודעת דרך הסוקט במקום לבדוק כל הזמן האם התקבל מידע חדש, דבר שהיה מעמיס מאוד על המערכת.

socket handshake

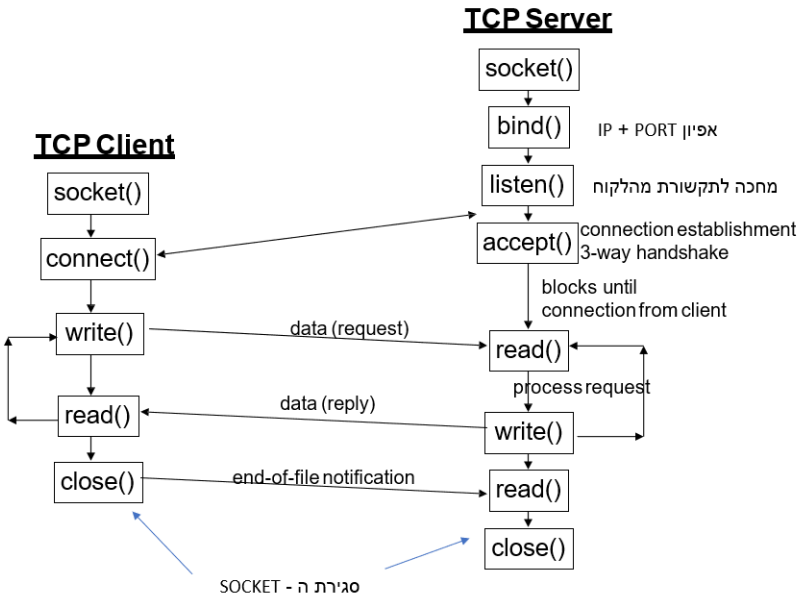
socket handshake הוא תהליך של יצירת חיבור בין שני sockets. בתוכנית שכתבנו, השרת והלקוח משתמשים ב-TCP (Transmission Control Protocol) כדי ליצור חיבור. תהליך יצירת הקשר באמצעות TCP כרוך בהחלפה של מספר הודעות בין הלקוח לשרת, המכונה לחיצת יד תלת כיוונית. לחיצת היד התלת כיוונית מורכבת מהשלבים הבאים:

1. הלקוח שולח הודעת "סנכרון" (SYN) לשרת, המעידה על כוונתו ליצור חיבור.
2. השרת מגיב בהודעת "סנכרון-אישור" (SYN-ACK), המאשר את קבלת הודעת ה-SYN וגם מציין את כוונתו ליצור חיבור.
3. לבסוף, הלקוח שולח הודעת "אישור" (ACK) לשרת, המאשר את קבלת הודעת SYN-ACK.

לאחר השלמת לחיצת היד התלת כיוונית, החיבור נוצר והלקוח והשרת יכולים להתחיל להחליף נתונים אחד עם השני.

בתוכנית שלנו, הפונקציות bind ו- listen בצד השרת והפונקציה connect בצד הלקוח משמשות להפעלת לחיצת היד התלת-כיוונית. פונקציות אלה חייבות להתקיים, כלומר, הפעלת התוכנית נעצרת עד ליצירת חיבור socket.

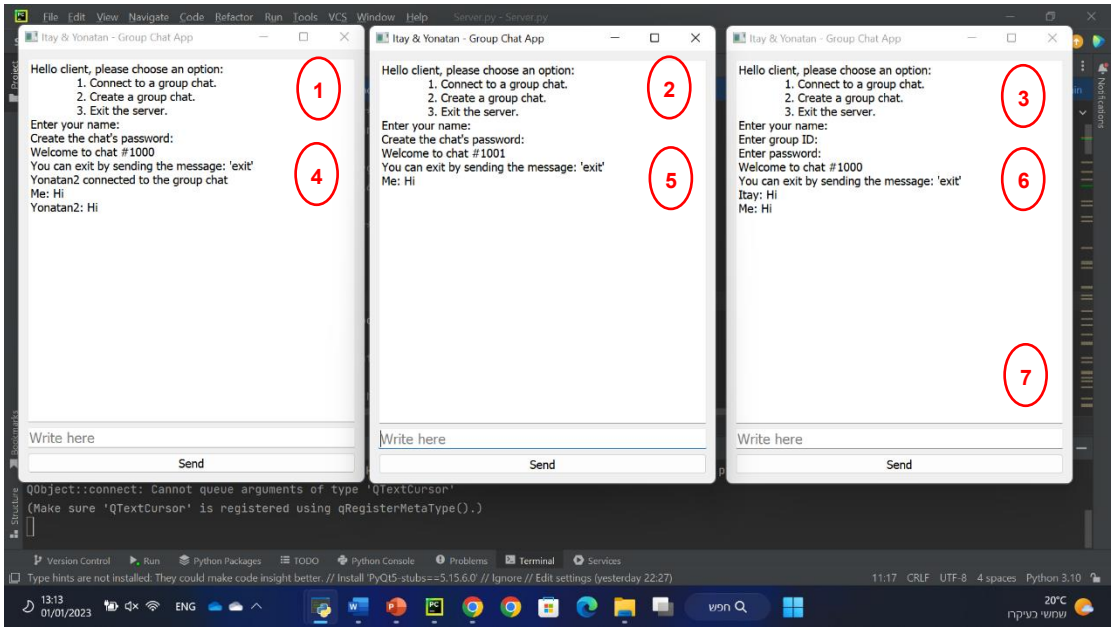
להלן תרשים הזרימה שהוצג בתרגול ועל בסיסו בנינו את התוכנית שלנו:



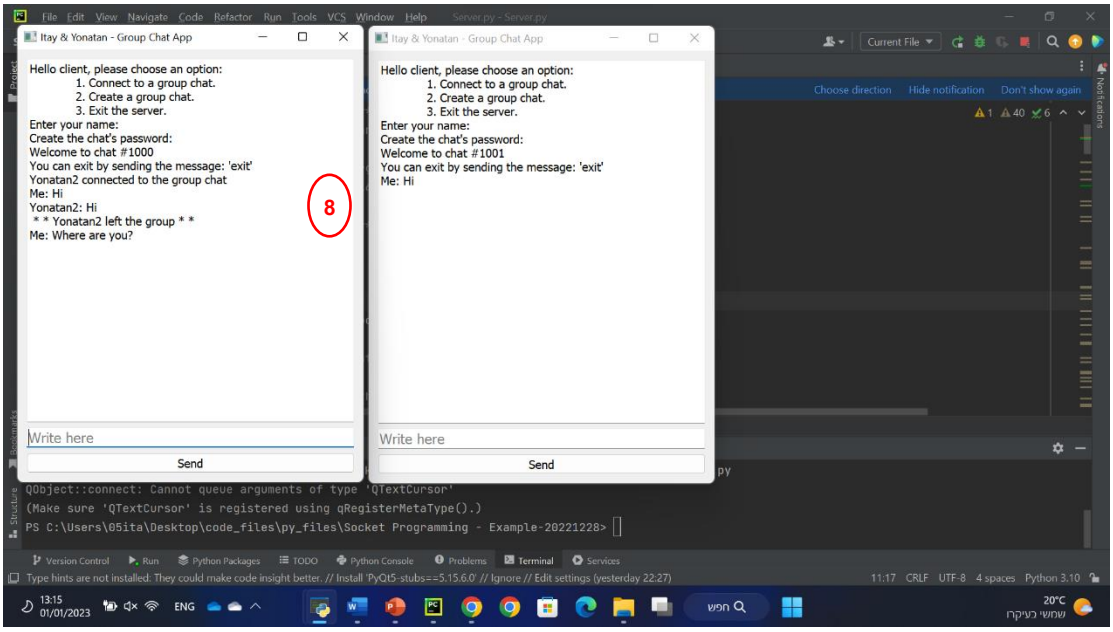
צילומי מסך

דוגמת הפעלה מס' 1:

הרצנו את הקוד של הסרבר וכן את הקוד של הלקוח 3X פעמים. סדר כתיבת הפקודות מופיע ליד כל פקודה בהתאמה:



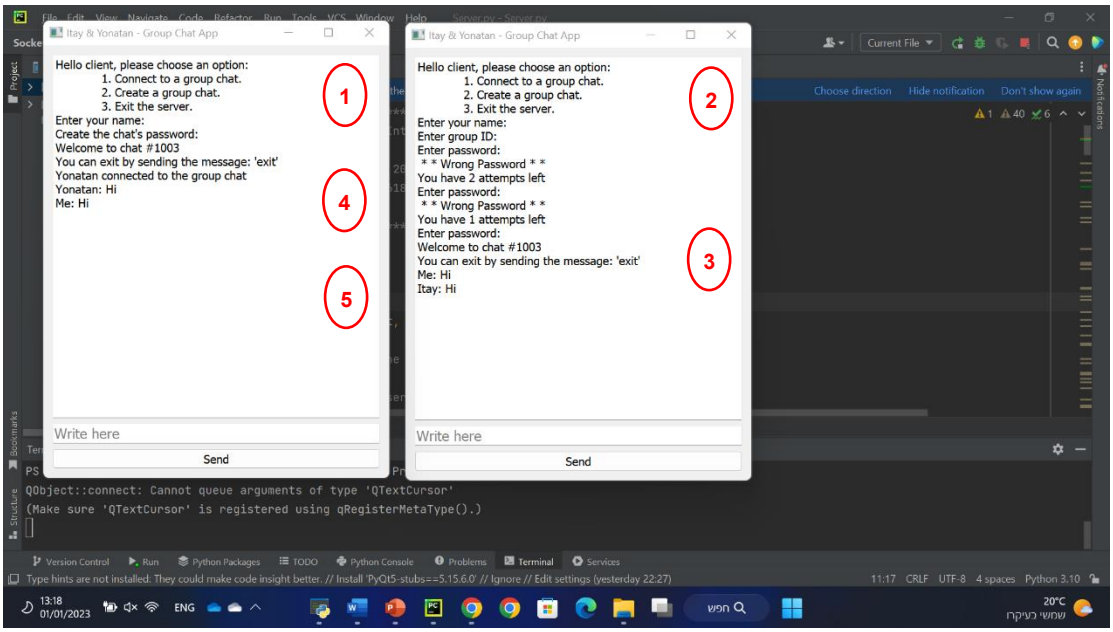
1. 2, Itay, 123
2. 2, Yonatan, 456
3. 1, Yonatan2, 1000, 123
4. Hi
5. Hi
6. Hi
7. Exit



8. Where are you?

דוגמת הפעלה מס' 2

הרצנו את הקוד של הסרבר וכן את הקוד של הלקוח 2X פעמים. סדר כתיבת הפקודות מופיע ליד כל פקודה בהתאמה:



1. 2, Itay, 123
2. 1, Yonatan, 1003, 789, 456, 123, Hi
3. Hi
4. Hi
5. exit