

# דוח מסכם: פרויקט גמר בرشות תקשורת מחשבים

## ومימוש יישום צ'אט TCP/IP נושא: ניתוח העבורה בפרוטוקול

מגישות:

רומ אשכנזי ת.ז 214438186

עדן שראיין ת.ז 324929223

[קישור ל-GitHub](https://github.com/littttttttttttt/project.git)

---

### 1. חלק ראשון- אריזה ולכידת מנות:

#### א. ייצירת קובץ הנתונים

בשלב הראשון יצרנו קובץ נתונים בפורמט CSV המתאר רצף הודעות בשכבה היישום (Application Layer) המתאר רצף הודעות בscpbat היישום (app\_protocol) בפרוטוקול TCP מעל HTTP.

כל שורה בקובץ מייצגת הודעה לוגית בין ללקוח (client\_browser) לשרת (web\_server), וכוללת מזהה הודעה (message\_id), פרוטוקול היישום (app\_protocol), צד מקור ויעד (src\_app, dst\_app), תוכן הודעה (timestamp).

תהליך הייצרה בוצע באופן ידני בשילוב כלי AI (Chat GPT) כדי ליצור תבנית שיחה אמינה בין שרת ללקוח.

קובץ CSV:

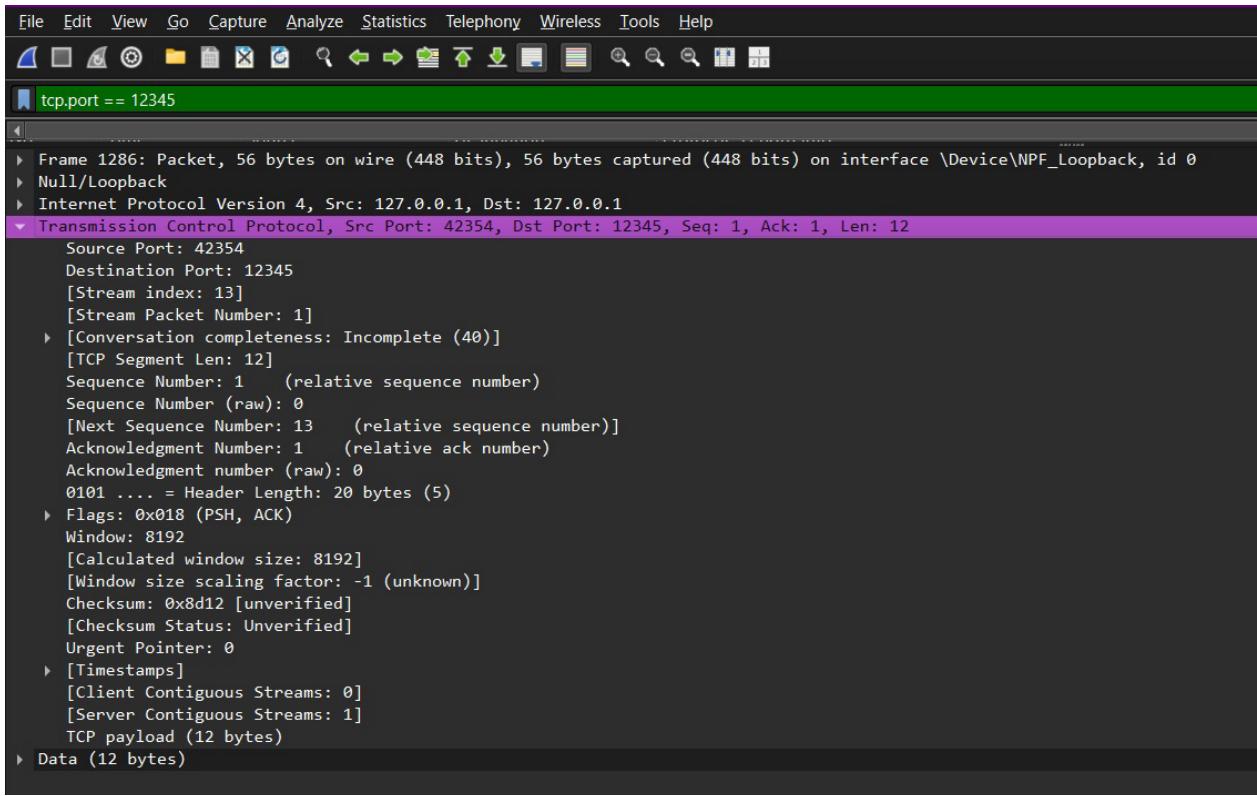
msg_id	app_protocol	src_app	dst_app	message	timestamp
1	HTTP	client_browser	web_server	GET /index.html	0.015
2	HTTP	web_server	client_browser	200 OK (image/png)	0.020
3	HTTP	client_browser	web_server	GET /styles.css	0.025
4	HTTP	web_server	client_browser	200 OK (text/css)	0.030
5	HTTP	client_browser	web_server	POST /login	0.045

## **ב. תהליך האריזה (Encapsulation Process)**

השתמשנו במחברת Jupyter ובקוד Python כדי לדמות את תהליך האריזה במערכת הפעלה:

1. **שכבה היישום:** התוכנה קראת ההודעה מה-CSV.
2. **שכבה התעבורה (Transport):** הקוד הוסיף כותרת TCP (כולל מספרי פורטים ודגלים).
3. **שכבה הרשת (Network):** ההודעה נשלחת דרך כותבת ה-IP (127.0.0.1) loopback (127.0.0.1), וניתן לצפות במנוף ה-IP שנוצרות ב-Wireshark.

בפלט המחברת ניתן לראות את המידע הבינארי (Hexdump) שמרת איך ההודעה "עטופה" בכותרות השונות לפני השיליחה, ה-IP מוגים כיצד ההודעה משכבה היישום נעתפה בהדרגה בכותרות השונות לפני השיליחה.



```
Frame 1286: Packet, 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface \Device\NPF_Loopback, id 0
  Null/Loopback
  Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
  Transmission Control Protocol, Src Port: 42354, Dst Port: 12345, Seq: 1, Ack: 1, Len: 12
    Source Port: 42354
    Destination Port: 12345
    [Stream index: 13]
    [Stream Packet Number: 1]
    [Conversation completeness: Incomplete (40)]
    [TCP Segment Len: 12]
    Sequence Number: 1 (relative sequence number)
    Sequence Number (raw): 0
    [Next Sequence Number: 13 (relative sequence number)]
    Acknowledgment Number: 1 (relative ack number)
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
    Flags: 0x018 (PSH, ACK)
    Window: 8192
    [Calculated window size: 8192]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x8d12 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    [Timestamps]
    [Client Contiguous Streams: 0]
    [Server Contiguous Streams: 1]
    TCP payload (12 bytes)
    Data (12 bytes)
```

## ג. תהליך הלכידה וניתוח Wireshark

לצורך ניתוח התעבורה הופעלת תוכנת Wireshark על ממושך ה-loopback, בזמן שליחת הודעות בין הלקוח לשרת.

בלכידה ניתן לראות תעבורת TCP כאשר כתובת המקור והיעד הן 127.0.0.1 והתקשורת מתבצעת מול פורט השרת 12345.

בנוסף, ניתן להבחין כי הדגל PSH מציין כי הנתונים שב-payload נשלחים מידית לישום בצד מקבל.

## ד. תיאור והסביר תעבורה שנלכדה Wireshark

במהלך הריצת המערכת נלכדה תעבורת רשות באמצעות Wireshark על ממושך ה-loopback (127.0.0.1). התעבורה שניצפתה מייצגת זרימת תקשורת TCP בין ללקוח לשרת מקומיים, כאשר החיבור מתבצע דרך פורט היעד 12345.

בתמונה מוצגת זרימת תעבורת TCP הכוללת רצף של חבילות שנלכדו ב-Wireshark, המועלות דרך ממושך ה-loopback (127.0.0.1) בין ללקוח לשרת מקומיים בפורט 12345, כאשר חלק מהחבילות מכילות payload המציג נתונים שנשלחו משבב היפוסם, רצף החבילות והמאפיינים שלן מוגנים את אופן העברת המידע בין שכבות המערכת ואת תהליך התקשורת בין הלקוח לשרת.

Capturing from Adapter for loopback traffic capture						
No.	Time	Source	Destination	Protocol	Length	Info
372170	3449.125757	127.0.0.1	127.0.0.1	TCP	56	41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=12
372171	3449.126017	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372286	3449.231183	127.0.0.1	127.0.0.1	TCP	69	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=16
372287	3449.231200	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372384	3449.335913	127.0.0.1	127.0.0.1	TCP	63	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=19
372385	3449.336008	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372408	3449.443277	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=12
372409	3449.443387	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372423	3449.547832	127.0.0.1	127.0.0.1	TCP	62	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=18
372433	3449.547941	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372448	3449.655142	127.0.0.1	127.0.0.1	TCP	63	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=19
372449	3449.655266	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372472	3449.768493	127.0.0.1	127.0.0.1	TCP	71	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=27
372473	3449.768681	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372492	3449.866300	127.0.0.1	127.0.0.1	TCP	62	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=18
372493	3449.866415	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372512	3449.971714	127.0.0.1	127.0.0.1	TCP	47	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=3
372513	3449.971849	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
372522	3450.076140	127.0.0.1	127.0.0.1	TCP	47	[TCP Retransmission] 41265 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=3
372523	3450.076237	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST] Seq=1 Win=0 Len=0
375394	3461.835998	127.0.0.1	127.0.0.1	TCP	58	[TCP Port numbers reused] 41265 → 12345 [SYN] Seq=0 Win=8192 Len=14
375399	3461.836121	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
375508	3462.341248	127.0.0.1	127.0.0.1	TCP	58	[TCP Port numbers reused] 41265 → 12345 [SYN] Seq=0 Win=8192 Len=14
375501	3462.341360	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
375600	3462.845466	127.0.0.1	127.0.0.1	TCP	58	[TCP Port numbers reused] 41265 → 12345 [SYN] Seq=0 Win=8192 Len=14
375601	3462.845586	127.0.0.1	127.0.0.1	TCP	44	12345 → 41265 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
402555	3633.649589	127.0.0.1	127.0.0.1	TCP	56	31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=12
402558	3633.649889	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402573	3633.757459	127.0.0.1	127.0.0.1	TCP	63	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=19
402574	3633.757583	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402589	3633.879882	127.0.0.1	127.0.0.1	TCP	55	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=11
402590	3633.871048	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402603	3633.977072	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=12
402604	3633.977216	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402619	3634.088268	127.0.0.1	127.0.0.1	TCP	58	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=14
402620	3634.088483	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402633	3634.193615	127.0.0.1	127.0.0.1	TCP	57	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=13
402634	3634.193744	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Win=0 Len=0
402643	3634.304547	127.0.0.1	127.0.0.1	TCP	55	[TCP Retransmission] 31938 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=11
402644	3634.304600	127.0.0.1	127.0.0.1	TCP	44	12345 → 31938 [RST] Seq=1 Ack=1 Win=0 Len=0

## חלק 2: יצירה יישום וניהוטה תעבורת

### א. הסבר כללי על מערכת ומבנה הקוד

המערכת שנבנתה מדמה תקשורת ברשת באמצעות פרוטוקול TCP והוא מורכבת משרת ולקוחות. השרת אחראי על קבלת חיבורים מלוקוחות, תיווך בין הודעות וניהול רשיימת הלקוחות הפעילים. הלקוחות מתחברים לשרת, שולחים אליו הודעות ומקבלים הודעות שנשלחו על ידי לקוחות אחרים דרך השרת. המטרה של המערכת היא לאפשר העברת הודעות בין לקוחות, כאשר כל הודעה עוברת דרך השרת שמשמש כמתווך. כל לקוח מזוהה לפי שם ייחודי שנשלח בתחילת ההתחברות, מה שמאפשר לשרת לעקוב אחרי מי שלח כל הודעה ולאן להעביר אותה.

השימוש נעשה באמצעות ספריית Socket של פיתון לצורך תקשורת ובאמצעות ספרייה Threading לשרת שתומך במספר חיבורים במקביל (5 לקוחות בו זמינה).

מבנה הקוד מחלק לשני רכיבים עיקריים:

- ▢ שרת (Server) -amazon לבקשת התחברים, יוצר תהליך (Thread) נפרד עבור כל לקוח, מקבל ממנו הודעה ו מעביר אותו ללקוחות אחרים.
  - ▢ לקוח (Client) - מהחבר לשרת, שולח את שמו, שולח הודעה וamazon להודעות שמגיעות מהשרת.
- השרת שומר מיליון של לקוחות מחוברים, שבו כל לקוח מזוהה לפי השם שהוא שלח ולכל שם הקשור החיבור הפעיל. כאשר הודעה מתתקבלת מלוקה, השרת מנתב אותה ללקוחות הרלוונטיים לפי לוגיקה פנימית.

### ב. הוראות התקנה והרצתה

#### הרצת השרת -

- ▢ יש להריץ את הקובץ `server.py`.
- ▢ השרתamazon לחיבורים נוכנים על פורט ברירת מחדל (לדוגמה 12345)
- ▢ השרת תומך בחיבור של 5 לקוחות במקביל.

#### הרצת הלקוח -

- ▢ כל לקוח מופעל על ידי הרצת הקובץ `client.py`.
- ▢ עם ההרצה, המשתמש מתבקש להזין שם ייחודי.
- ▢ ניתן לפתח מספר מופעים של לקוחות (בחולנות נפרדים) לצורך בדיקת תקשורת בין מספר משתמשים.

## ג. דוגמאות קלט ופלט:

בתמונה זו מוצגת הרצת הקובץ server.py, השרת נכנס למרכז האזנה וממתין לחיוברים מלוקחות:

```
Anaconda Prompt - python s  ×  +  ▾

(base) C:\Users\Eden>conda activate networks

(networks) C:\Users\Eden>cd Desktop

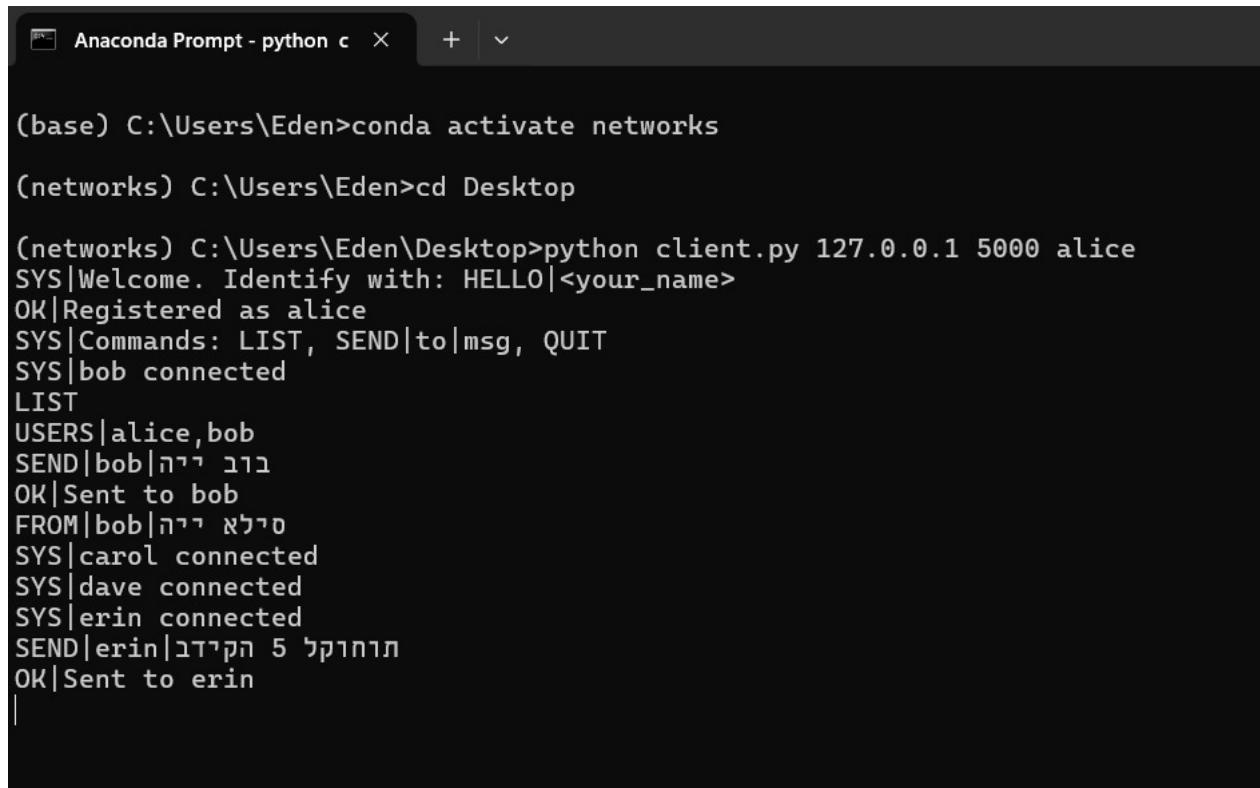
(networks) C:\Users\Eden\Desktop>dir
 Volume in drive C has no label.
 Volume Serial Number is 8207-228D

Directory of C:\Users\Eden\Desktop

18/12/2025  12:59    <DIR>      .
14/12/2025  11:49    <DIR>      ..
18/12/2025  12:53    <DIR>      .ipynb_checkpoints
18/12/2025  12:59            1,538 client.py
18/11/2024  07:22            2,371 Excel.lnk
14/12/2025  13:07            411 group03_tcpip_project1_input.csv
14/12/2025  12:57            428 group03_tcpip_project_input.csv
14/12/2025  10:56            437 group05_tcpip_project_input.csv
18/11/2024  09:48            2,392 Person 1 - Chrome.lnk
18/11/2024  07:22            2,392 PowerPoint.lnk
14/12/2025  13:09            18,828 raw_tcp_ip_notebook_fallback_annotated-v1.ipynb
18/12/2025  12:57            4,770 server.py
24/07/2025  11:22    <DIR>      study
18/12/2025  12:53            72 Untitled.ipynb
18/11/2024  07:22            2,409 Word.lnk
14/12/2025  13:41            665,565 סכום חוץ.docx
02/07/2025  10:22            432 יפני תואחסן ב.htm
02/07/2025  10:22            509,305 הדיד תואחסן ב.pdf
12/06/2025  10:59            8,069,594 הדיד תורדגן 2.pdf
24/04/2025  21:46            381,233 יפניאב (יקלח) טישו סוכיס (1).pdf
                           16 File(s)   9,662,177 bytes
                           4 Dir(s)  906,110,529,536 bytes free

(networks) C:\Users\Eden\Desktop>python server.py
|
```

בתמונה זו מוצגת הרצת ל Koho והתחברות- הלקוחה אליס מתחברת לשרת, לאחר ההתחברות מופיעה הודעה ברוכים הבאים ולאחר מכן מתחילה להופיע הודעות מערכת על התחברויות חדשות ושליחת הודעות בין משתמשים.



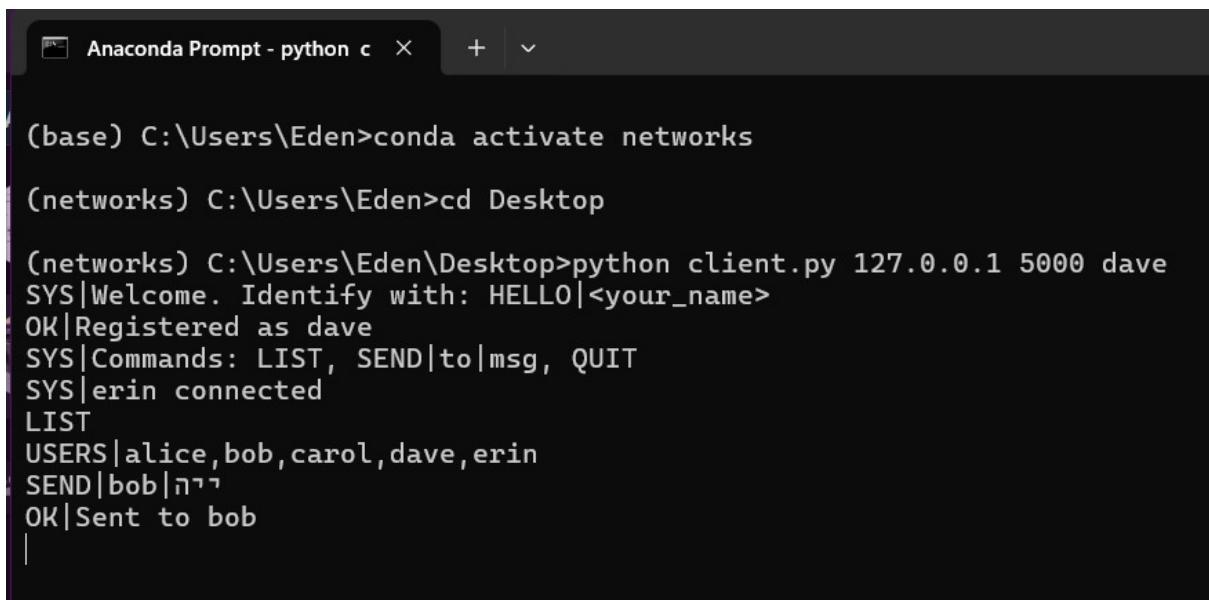
```
Anaconda Prompt - python c  X  +  ▾

(base) C:\Users\Eden>conda activate networks

(networks) C:\Users\Eden>cd Desktop

(networks) C:\Users\Eden\Desktop>python client.py 127.0.0.1 5000 alice
SYS|Welcome. Identify with: HELLO|<your_name>
OK|Registered as alice
SYS|Commands: LIST, SEND|to|msg, QUIT
SYS|bob connected
LIST
USERS|alice,bob
SEND|bob|ריה|bob
OK|Sent to bob
FROM|bob|ריה|bob
SYS|carol connected
SYS|dave connected
SYS|erin connected
SEND|erin|הקידב|5
OK|Sent to erin
|
```

הרצת Koho נוספת בשם דיב- המשמש דיב מתחבר לשרת, מקבל את רשימה המשתמשים הקיימים במערכת ושולח הודעה לבוב. ניתן לראות שהתחלה דומה להרצת Koho הראשון וכי המערכת תומכת במספר לקוחות בו זמן.



```
Anaconda Prompt - python c  X  +  ▾

(base) C:\Users\Eden>conda activate networks

(networks) C:\Users\Eden>cd Desktop

(networks) C:\Users\Eden\Desktop>python client.py 127.0.0.1 5000 dave
SYS|Welcome. Identify with: HELLO|<your_name>
OK|Registered as dave
SYS|Commands: LIST, SEND|to|msg, QUIT
SYS|erin connected
LIST
USERS|alice,bob,carol,dave,erin
SEND|bob|ריה|bob
OK|Sent to bob
|
```

#### ד. ניתוח תעבורת של היבום עד שכבה הרשת

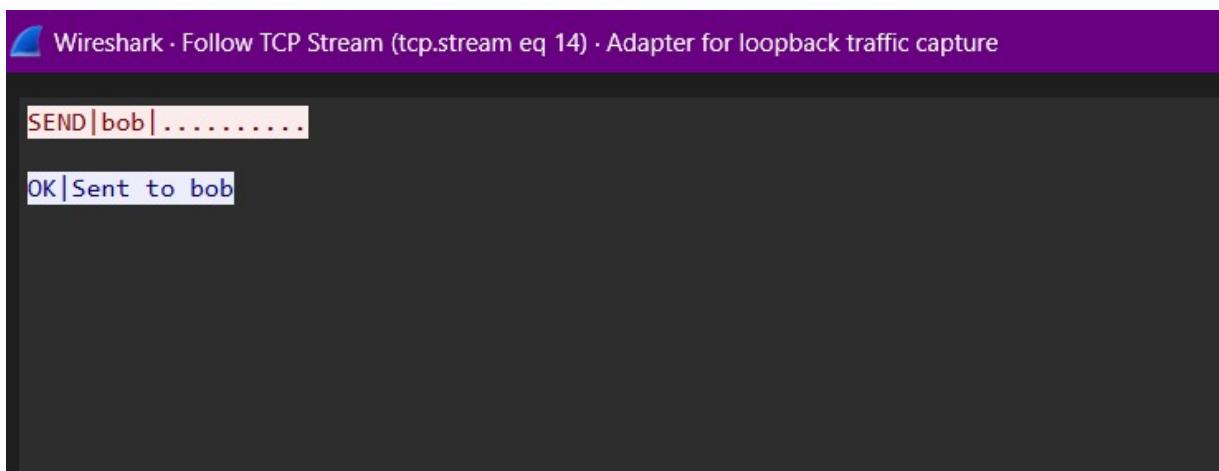
היבום עושה שימוש בפרוטוקול TCP לצורך תקשורת בין הלקוחות לשרת. כל ל Koh יוצר חיבור TCP באמצעות IP מקומית (127.0.0.1), החיבור נשמר فترة לאורך זמן התקשורת ומאפשר שליחת וקבלת הודעות טקסטואליות ברמת היבום.

ברמת שכבה היבום (Application layer) – הלקוחות שולחים פקודות בפורמט טקסטואלי כגון:

SEND|bob|<message> □  
QUIT פקודות נוספות כמו LIST ו- □

ב- Wireshark, באמצעות סינון לפי tcp.stream ניתן לראות בבירור את תוכן ה הודעות שנשלחו.

בתמונה הראשונה מוצגת ה הודעת SEND|bob|..... שנשלחה מהלקוח לשרת ולאחריה ה הודעת התגובה מהשרת OK| Sent to bob, המעידת על עיבוד מוצלח של הבקשה ברמת היבום.



בתמונה השנייה מוצגת רשימת חבילות TCP הרלוונטיות לאותו stream. ניתן להזיהות:

חבילות TCP מסווג [PSH, ACK] ה כוללות payload, כולן נתוני יישום. □  
חבילות [ACK] המאשרות קבלת מידע. □  
שימוש ב- sequence number ו- acknowledgment number של TCP. □

tcp.stream eq 14						
No.	Time	Source	Destination	Protocol	Length	Info
161	60.979775	127.0.0.1	127.0.0.1	TCP	64	55943 → 5000 [PSH, ACK] Seq=1 Ack=1 Win=255 Len=20
162	60.979862	127.0.0.1	127.0.0.1	TCP	44	5000 → 55943 [ACK] Seq=1 Ack=21 Win=255 Len=0
165	60.980406	127.0.0.1	127.0.0.1	TCP	59	5000 → 55943 [PSH, ACK] Seq=1 Ack=21 Win=255 Len=15
166	60.980427	127.0.0.1	127.0.0.1	TCP	44	55943 → 5000 [ACK] Seq=21 Ack=16 Win=255 Len=0

ניתוח זה מדגים כיצד פקודות ברמת היישום מתרגמות לתעבורה TCP/IP וכי怎 ניתן לעקוב אחר זרימת המידע משכבה היפיינומן ועד שכבה הרשות באמצעות כלי ניתוח תעבורה.

### **חלק 3: תיאור שימוש במבנה מלאכותית**

במהלך העבודה נעשה שימוש בכללי בינה מלאכותית (ChatGPT) ככלי עזר להבנה ופתרון בעיות. המטרה המרכזית הייתה לקבל הבנות בנוגע לדרישות המשימה, לבנייה הרצוי של המערכת, ולהבין טוב יותר את עקרונות העבודה עם sockets ו- TCP בפייתון.

נעשה שימוש בכללי לקבלת עזרה בזיהוי תקלות בקוד ובפתרון בעיות טכניות שעלו במהלך הפיתוח.

דוגמאות פרומטיטים:

□ "תוכל בקשה להסביר לי על השלב שבו צריך לנצל חיבורים בין לköחות, מה השרת אמר לעשות?"

□ "איך אני יכולה לראות ב Wireshark את הודעה שנשלחה מהלכה לשרת?"

□ "יש לי קוד שמתחילה לעבוד ואז נתקע, איך אני יכולה זהות איפה הבעיה?"