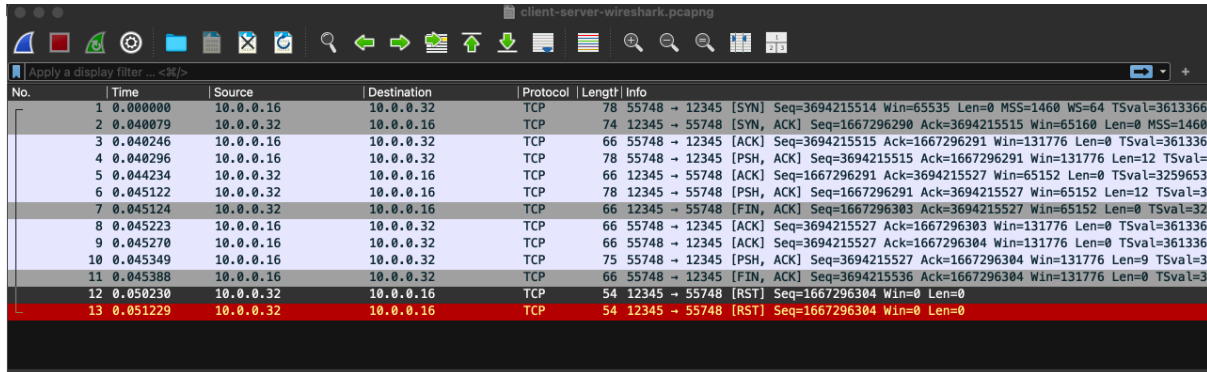


רשתות תרגיל 2 חלק 1

מצורף כאן צילום מסך של תקשורת בין שני מחשבים שונים. הלקוח (10.0.0.16) מריץ את הקובץ tcp_client.py אשר פונה לשרת (10.0.0.32) המריץ את הקובץ tcp_server.py.



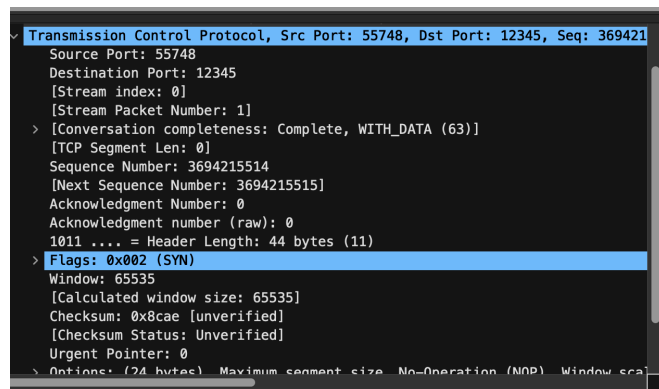
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.0.16	10.0.0.32	TCP	78	55748 → 12345 [SYN] Seq=3694215514 Win=65535 Len=0 MSS=1460 WS=64 TSval=3613366
2	0.040079	10.0.0.32	10.0.0.16	TCP	74	12345 → 55748 [SYN, ACK] Seq=1667296290 Ack=3694215515 Win=65160 Len=0 MSS=1460
3	0.040246	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215515 Ack=1667296291 Win=131776 Len=0 TSval=3613366
4	0.040296	10.0.0.16	10.0.0.32	TCP	78	55748 → 12345 [PSH, ACK] Seq=3694215515 Ack=1667296291 Win=131776 Len=12 TSval=3613366
5	0.044234	10.0.0.32	10.0.0.16	TCP	66	12345 → 55748 [ACK] Seq=1667296291 Ack=3694215527 Win=65152 Len=0 TSval=3259653
6	0.045122	10.0.0.32	10.0.0.16	TCP	78	12345 → 55748 [PSH, ACK] Seq=1667296291 Ack=3694215527 Win=65152 Len=12 TSval=3259653
7	0.045124	10.0.0.32	10.0.0.16	TCP	66	12345 → 55748 [FIN, ACK] Seq=1667296303 Ack=3694215527 Win=65152 Len=0 TSval=3259653
8	0.045223	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215527 Ack=1667296303 Win=131776 Len=0 TSval=3613366
9	0.045270	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215527 Ack=1667296304 Win=131776 Len=0 TSval=3613366
10	0.045349	10.0.0.16	10.0.0.32	TCP	75	55748 → 12345 [PSH, ACK] Seq=3694215527 Ack=1667296304 Win=131776 Len=9 TSval=3613366
11	0.045388	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [FIN, ACK] Seq=3694215536 Ack=1667296304 Win=131776 Len=0 TSval=3613366
12	0.050230	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0
13	0.051229	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0

תהליך הקמת החיבור (Three-Way Handshake)

תהליך זה מתרחש בחבילות 1 עד 3. זהו תהליך לחיצת היד המשולשת.

1. שלב ראשון (Packet 1) - שליחת SYN:

- הלקוח (10.0.0.16) יוזם את החיבור ושולח הודעת SYN לשרת (10.0.0.32).
- מספרים סידוריים: המקור מגריל מספר סידורי התחלתי.
- Seq = 3694215514
- Ack = 0 (כי עדיין לא התקבל כלום מהצד השני).
- משמעות: אני רוצה לפתוח איתך שיחה, המספר הסידורי שלי מתחיל ב-"3694215514".



2. שלב שני (Packet 2) - שליחת SYN, ACK:

- תיאור: השרת מקבל את הבקשה, ומחזיר הודעה המכילה גם SYN (כדי לפתוח ערוץ משלו) וגם ACK (אישור הבקשה של הלקוח).
- מספרים סידוריים:
- Seq = 1667296290 (המספר הסידורי ההתחלתי שהשרת הגריל).
- Ack = 3694215515 (החישוב הוא: Seq של הלקוח + 1).

- משמעות: קיבלתי את ההודעה שלך והוספתי לה 1, ואני גם רוצה לפתוח איתך שיחה במספר הסידורי שלי.

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 166729
Source Port: 12345
Destination Port: 55748
[Stream index: 0]
[Stream Packet Number: 2]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 0]
Sequence Number: 1667296290
[Next Sequence Number: 1667296291]
Acknowledgment Number: 3694215515
1010 .... = Header Length: 40 bytes (10)
> Flags: 0x012 (SYN, ACK)
Window: 65160
[Calculated window size: 65160]
Checksum: 0x2eaf [unverified]
[Checksum Status: Unverified]

```

3. שלב שלישי (Packet 3) - שליחת ACK:

- הלקוח מאשר את קבלת ה-SYN של השרת ע"י שליחת ACK. החיבור כעת הוקם.
- מספרים סידוריים:

- Seq = 3694215515 (התקדם ב-1 לעומת חבילה 1, כי דגל ה-SYN צורך מספר סידורי אחד).
- Ack = 1667296291 (החישוב הוא: ה-Seq של השרת + 1).

```

> Frame 3: Packet, 66 bytes on wire (528 bits), 66 bytes captured (528 bits) o
> Ethernet II, Src: de:6e:31:1c:f5:de (de:6e:31:1c:f5:de), Dst: Intel_46:ba:9a
> Internet Protocol Version 4, Src: 10.0.0.16, Dst: 10.0.0.32
> Transmission Control Protocol, Src Port: 55748, Dst Port: 12345, Seq: 369421
Source Port: 55748
Destination Port: 12345
[Stream index: 0]
[Stream Packet Number: 3]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 0]
Sequence Number: 3694215515
[Next Sequence Number: 3694215515]
Acknowledgment Number: 1667296291
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 2059
[Calculated window size: 131776]
[Window size scaling factor: 64]
Checksum: 0x53d0 [unverified]

```

העברת נתונים וניתוח המספרים

1. שליחת מידע (Packet 4):

- תיאור: הלקוח (10.0.0.16) שולח מידע לשרת. ניתן לראות את הדגל שמבקש להעביר את המידע לאפליקציה מיד.
- גודל המידע: Len = 12 (רואים זאת בעמודת Length או Info).
- מספר סידורי: Seq = 3694215515.

```
Sequence Number: 3694215515
[Next Sequence Number: 3694215527]
Acknowledgment Number: 1667296291
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x018 (PSH, ACK)
Window: 2059
[Calculated window size: 131776]
[Window size scaling factor: 64]
Checksum: 0x4960 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]
TCP payload (12 bytes)
Data (12 bytes)
Data: 596f617620456c6861646164
```

```
0000 f4 06 69 46 ba 9a de 6e 31 1c f5 de 08 00 45 00  ..iF...n 1.....E
0010 00 40 00 00 40 00 40 06 26 89 0a 00 00 10 0a 00  @...@. &.....
0020 00 20 d9 c4 30 39 dc 31 41 5b 63 60 ec 23 80 18  ..09.1 A[c`.#..
0030 08 0b 49 60 00 00 01 01 08 0a d7 5f 97 5d c2 4a  ..I'...._..].J
0040 5e 9b 59 6f 61 76 20 45 6c 68 61 64 61 64      ^..Yoav E lhadad
```

2. אישור קבלת מידע (Packet 5):

- תיאור: השרת (10.0.0.32) מאשר את קבלת המידע.
- חישוב ה-Ack: השרת מבצע חישוב פשוט: המספר הסידורי הקודם שקיבל + גודל המידע שקיבל.
- החישוב במספרים:
 - 3694215515 (ה-Seq של חבילה 4)
 - + 12 (ה-Len של חבילה 4)
 - = 3694215527
- לכן בחבילה 5 אנו רואים Ack = 3694215527. זהו האישור שהמידע התקבל בשלמותו והשרת מצפה לבית הבא במספר זה.

```

[Stream Packet Number: 5]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 0]
Sequence Number: 1667296291
[Next Sequence Number: 1667296291]
Acknowledgment Number: 3694215527
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 509
[Calculated window size: 65152]
[Window size scaling factor: 128]
Checksum: 0x59cd [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]

```

○

3. תשובת השרת (Packet 6):

- תיאור: השרת (10.0.0.32) שולח חזרה ללקוח את המחרוזת "YOAV ELHADAD".
- ה Seq הוא 1667296291 (זהה ל-ACK הקודם, כי השרת עדיין לא שלח מידע, רק אישר).
- ה Ack הוא 3694215527 (השרת עדיין מצפה לבית ה-27, כלומר מאשר שקיבל את השם).
- האורך הוא 12 בתים

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 1, Ack: 13, Len: 12
Source Port: 12345
Destination Port: 55748
[Stream index: 0]
[Stream Packet Number: 6]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 12]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1667296291
[Next Sequence Number: 13 (relative sequence number)]
Acknowledgment Number: 13 (relative ack number)
Acknowledgment number (raw): 3694215527
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x018 (PSH, ACK)
Window: 509
[Calculated window size: 65152]
[Window size scaling factor: 128]
Checksum: 0xcffd [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]
TCP payload (12 bytes)

```

4. שליחת תעודת הזהות (Packet 8)

- תיאור: הלקוח (10.0.0.16) שולח את ההודעה השנייה המכילה את תעודת הזהות: "315853793"
- החישוב במספרים:
 - Seq=3694215527, המספר הסידורי הקודם של הלקוח בחבילה 4 היה 3694215515, שלחנו 12 בתים ולכן $3694215515 + 12 = 3694215527$.
 - Ack=1667296303, הלקוח קיבל מהשרת את תשובת ה Echo בחבילה 6 שהייתה באורך 12 בתים והתחילה ב 1667296291 לכן הלקוח מאשר קבלה ומצפה ל: $1667296291 + 12 = 1667296303$.

■ האורך הוא 9 בתים

```

Transmission Control Protocol, Src Port: 55748, Dst Port: 12345, Seq: 13, Ack: 13, Len: 0
  Source Port: 55748
  Destination Port: 12345
  [Stream index: 0]
  [Stream Packet Number: 8]
  [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 13 (relative sequence number)
  Sequence Number (raw): 3694215527
  [Next Sequence Number: 13 (relative sequence number)]
  Acknowledgment Number: 13 (relative ack number)
  Acknowledgment number (raw): 1667296303
  1000 .... = Header Length: 32 bytes (8)
  [Flags: 0x010 (ACK)]
  Window: 2059
  [Calculated window size: 131776]
  [Window size scaling factor: 64]
  Checksum: 0x53af [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps]
  [Timestamps]
  [SEQ/ACK analysis]
  [Client Contiguous Streams: 1]
  [Server Contiguous Streams: 1]

```

חלק 3: סיום החיבור

1. ניסיון סיום (Packet 7):
 - השרת (10.0.0.32) שולח ACK, FIN.
 - משמעות: השרת מודיע "סיימתי לשלוח מידע, אני רוצה לסגור את החיבור מצדי".
2. סגירה מיידיה / תקלה (Packet 12-13):
 - רואים הודעות של RST Reset.
 - הסבר: במקום תהליך סגירה מסודר (שבו הצד השני שולח FIN משלו ו-ACK), נשלחה פקודת "Reset". זה קורה בדרך כלל כאשר צד אחד סוגר את התוכנה בכוח, או שהפורט כבר לא מאזין, והוא "שובר" את החיבור מיד במקום לסגור אותו בעדינות.

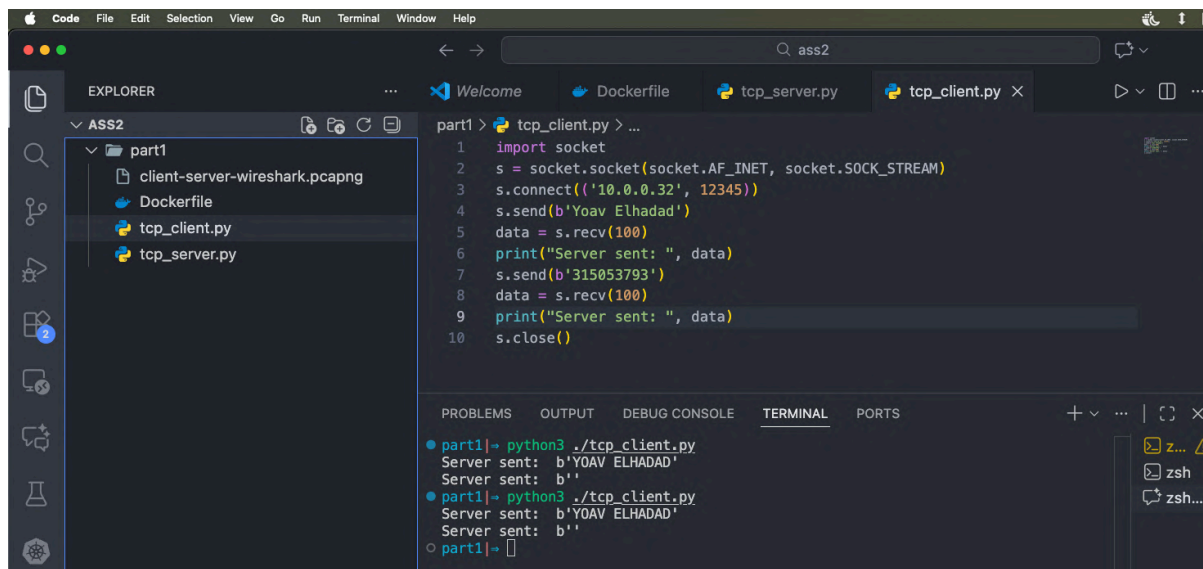
No.	Time	Source	Destination	Protocol	Length	Info
11	0.045388	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [FIN, ACK] Seq=3694215536 Ack=1667296304 Win=131776 Len=0 TSval=3
12	0.050230	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0
13	0.051229	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0


```

Frame 12: Packet, 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
  Ethernet II, Src: Intel_46:ba:9a (f4:06:09:46:ba:9a), Dst: de:6e:31:1c:f5:de
  Internet Protocol Version 4, Src: 10.0.0.32, Dst: 10.0.0.16
  Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 1667296304
    Source Port: 12345
    Destination Port: 55748
    [Stream index: 0]
    [Stream Packet Number: 12]
    [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 1667296304
    [Next Sequence Number: 1667296304]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
    [Flags: 0x004 (RST)]
    Window: 0
    [Calculated window size: 0]
    [Window size scaling factor: 128]

```

נספחים:
הקוד של הלקוח:



The screenshot shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows a project named 'ASS2' with a subfolder 'part1' containing files: 'client-server-wireshark.pcapng', 'Dockerfile', 'tcp_client.py', and 'tcp_server.py'. The main editor area displays the code for 'tcp_client.py' in 'part1'. The code is as follows:

```
1 import socket
2 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 s.connect(('10.0.0.32', 12345))
4 s.send(b'Yoav Elhadad')
5 data = s.recv(100)
6 print("Server sent: ", data)
7 s.send(b'315053793')
8 data = s.recv(100)
9 print("Server sent: ", data)
10 s.close()
```

Below the code editor, the 'TERMINAL' tab is active, showing the execution of the script. The output is as follows:

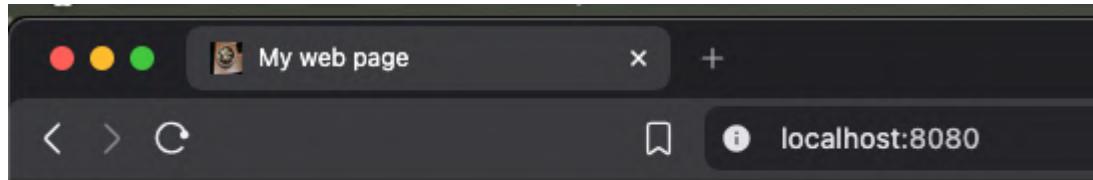
```
part1| python3 ./tcp_client.py
Server sent: b'YOAV ELHADAD'
Server sent: b''
part1| python3 ./tcp_client.py
Server sent: b'YOAV ELHADAD'
Server sent: b''
part1|
```

On the right side of the terminal, there are icons for 'zsh' and 'zsh...'.

רשתות תרגיל 2 חלק 2

דוגמא של עמוד הבית:

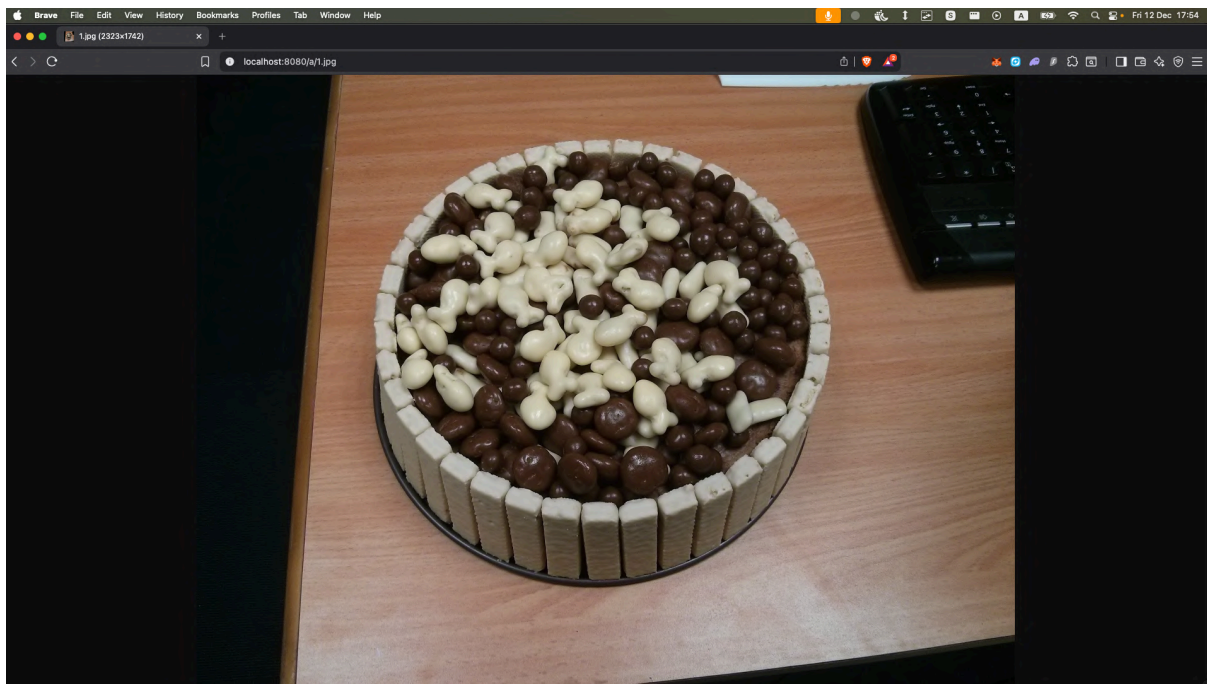
גישה ל / שמובילה לקבלת הקובץ index.html.



hello

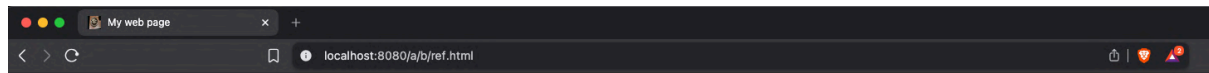
דוגמא של גישה לתמונה באופן ישיר:

גישה לכתובת a/1.jpg שמובילה לקבלת התמונה 1 בתיקיה a שבתוך התיקיה files.



דוגמא של גישה לדף מורכב:

גישה לכתובת `a/b/ref.html` הוא דף אשר טוען תמונות נוספות רבות ולכן מתבצעות קריאות רבות עם `connection keep alive` עד אשר הדפדפן מסיים לקבל את כל הקבצים אשר דרושים לו להצגת העמוד.

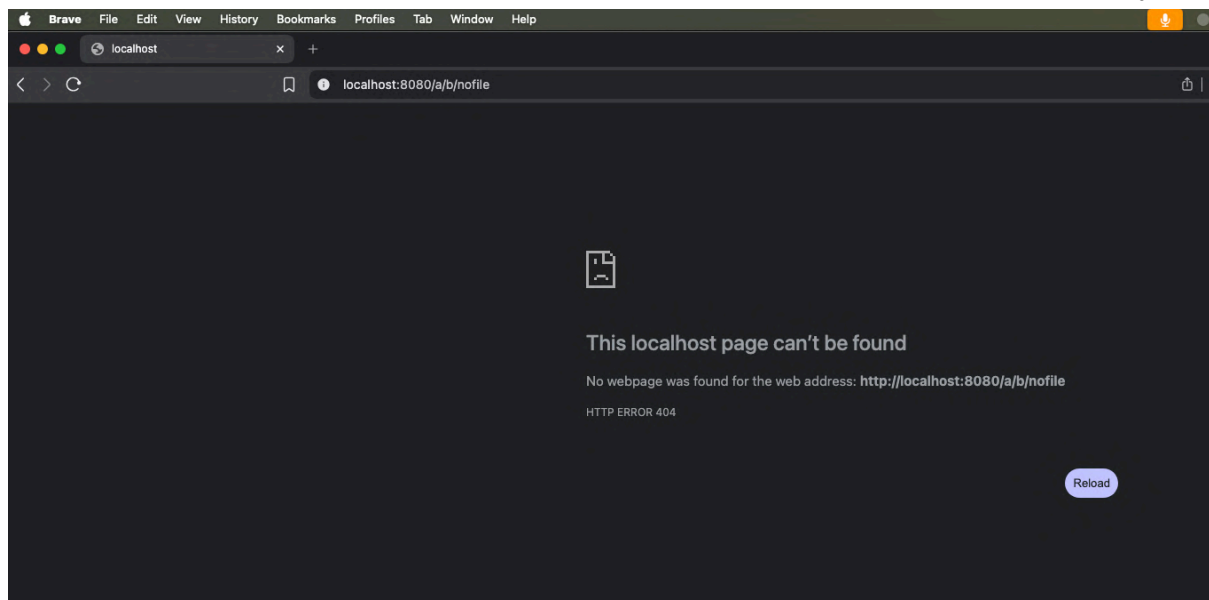


Amazing:



דוגמא של גישה לדף שלא קיים:

גישה לכתובת `a/b/nofile` גורמת לנו לקבל את התשובה `not found 404` כי הקובץ הזה לא קיים ב `.filesystem`.



No.	Time	Source	Destination	Protocol	Length	Info
11	47.192137	127.0.0.1	127.0.0.1	TCP	68	56875 → 8080 [SYN, Seq=3015514758 Win=65535 Len=0 MSS=16344 WS=64 TSval=26659
12	47.192184	127.0.0.1	127.0.0.1	TCP	68	8080 → 56875 [SYN, ACK] Seq=1528830962 Ack=3015514759 Win=65535 Len=0 MSS=163
13	47.192195	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015514759 Ack=1528830963 Win=408320 Len=0 TSval=26659
14	47.192203	127.0.0.1	127.0.0.1	TCP	56	[TCP Window Update] 8080 → 56875 [ACK] Seq=1528830963 Ack=3015514759 Win=4083
15	47.192335	127.0.0.1	127.0.0.1	HTTP	689	GET / HTTP/1.1
16	47.192351	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [ACK] Seq=1528830963 Ack=3015515392 Win=407744 Len=0 TSval=23671
17	47.192352	127.0.0.1	127.0.0.1	HTTP	279	HTTP/1.1 200 OK
18	47.193264	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515392 Ack=1528831186 Win=408128 Len=0 TSval=26659
19	47.558855	127.0.0.1	127.0.0.1	HTTP	654	GET /favicon.ico HTTP/1.1
20	47.558878	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [ACK] Seq=1528831186 Ack=3015515990 Win=407168 Len=0 TSval=23671
21	47.559292	127.0.0.1	127.0.0.1	HTTP	3247	HTTP/1.1 200 OK
22	47.559303	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515990 Ack=1528834377 Win=404992 Len=0 TSval=26659
23	48.560171	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [FIN, ACK] Seq=1528834377 Ack=3015515990 Win=407168 Len=0 TSval=
24	48.560253	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515990 Ack=1528834378 Win=404992 Len=0 TSval=26659
25	49.114447	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [FIN, ACK] Seq=3015515990 Ack=1528834378 Win=404992 Len=0 TSval=
26	49.114498	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [ACK] Seq=1528834378 Ack=3015515991 Win=407168 Len=0 TSval=23671

הסבר קצר על חלק מהפקטות

פקטה 15 (HTTP GET): זוהי בקשת **HTTP** מסוג **GET** שנשלחת מהלקוח אל השרת. בבקשה זו, הלקוח מבקש לקבל את המשאב הראשי של האתר כלומר את דף הבית. התקשורת מתבצעת בתוך המחשב המקומי זוהי תחילת השיחה ברמת האפליקציה, שבה הדפדפן או התוכנה מבקשים מידע מהשרת המאזין.

פקטה 16 (TCP ACK): זוהי הודעת אישור (**ACK**) ברמת פרוטוקול התעבורה **TCP**, הנשלחת מהשרת (פורט 8080) חזרה ללקוח. השרת מאותת ללקוח שהוא קיבל בהצלחה את בקשת ה-**GET** שנשלחה בפקטה הקודמת (מספר 15). נשים לב שגודל המידע (Len) הוא 0, מכיוון שזוהי הודעת בקרה טכנית בלבד ללא תוכן של דף אינטרנט. הודעה זו הכרחית כדי לשמור על אמינות החיבור ולוודא ששום מידע לא אבד בדרך לפני שהשרת מתחיל לעבד את התשובה.

פקטה 17 (HTTP 200 OK): זוהי התשובה של השרת לבקשה המקורית, הכוללת את קוד הסטטוס הסטנדרטי **200 OK**. המשמעות היא שהשרת מצא את הדף המבוקש, עיבד את הבקשה בהצלחה, וכעת הוא שולח את הנתונים חזרה ללקוח. בתוך הפקטה הזו נמצא ה-**Payload** (המטען), כלומר התוכן הממשי של דף ה-**HTML** או המידע שהלקוח ביקש. כאן מסתיים מחזור הבקשה-תגובה (Request-Response) המוצלח של פרוטוקול ה-**HTTP** עבור משאב זה.

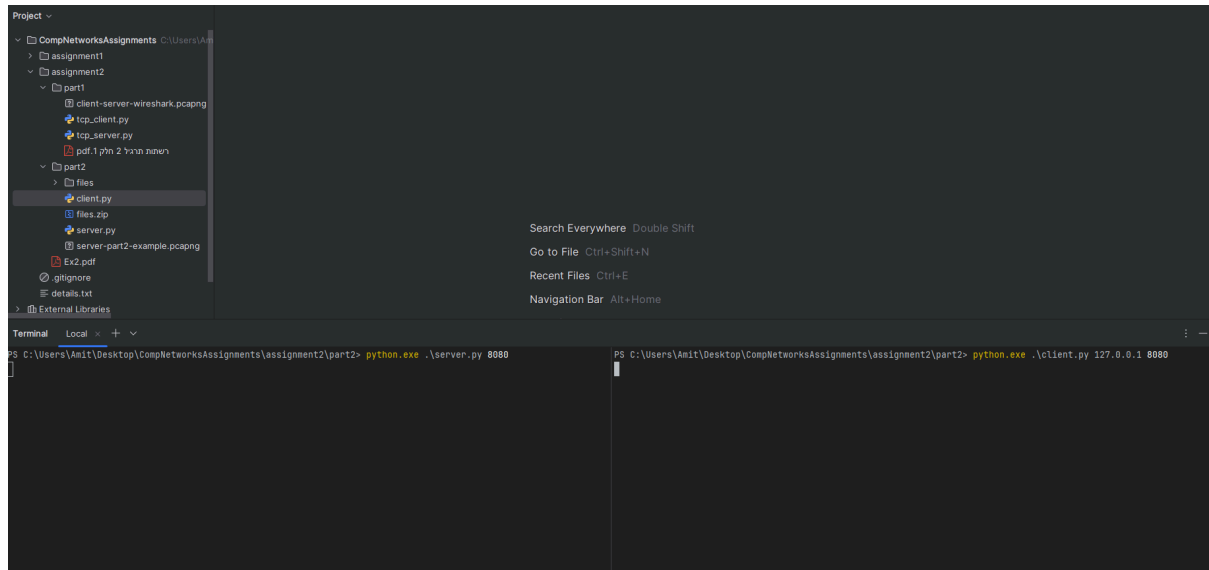
17	47.192352	127.0.0.1	127.0.0.1	HTTP	279	HTTP/1.1 200 OK
18	47.193264	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515392 Ack=1528831186 Win=408128 Len=0 TSval=26659
19	47.558855	127.0.0.1	127.0.0.1	HTTP	654	GET /favicon.ico HTTP/1.1
20	47.558878	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [ACK] Seq=1528831186 Ack=3015515990 Win=407168 Len=0 TSval=23671
21	47.559292	127.0.0.1	127.0.0.1	HTTP	3247	HTTP/1.1 200 OK
22	47.559303	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515990 Ack=1528834377 Win=404992 Len=0 TSval=26659
23	48.560171	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [FIN, ACK] Seq=1528834377 Ack=3015515990 Win=407168 Len=0 TSval=
24	48.560253	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [ACK] Seq=3015515990 Ack=1528834378 Win=404992 Len=0 TSval=26659
25	49.114447	127.0.0.1	127.0.0.1	TCP	56	56875 → 8080 [FIN, ACK] Seq=3015515990 Ack=1528834378 Win=404992 Len=0 TSval=
26	49.114498	127.0.0.1	127.0.0.1	TCP	56	8080 → 56875 [ACK] Seq=1528834378 Ack=3015515991 Win=407168 Len=0 TSval=23671

> Frame 17: Packet, 279 bytes on wire (2232 bits), 279 bytes captured (2232 bi > Null/Loopback > Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 > Transmission Control Protocol, Src Port: 8080, Dst Port: 56875, Seq: 1528830 Source Port: 8080 Destination Port: 56875 [Stream index: 3] [Stream Packet Number: 7] [Conversation completeness: Complete, WITH_DATA (31)] [TCP Segment Len: 223] Sequence Number: 1528830963 [Next Sequence Number: 1528831186] Acknowledgment Number: 3015515392 1000 = Header Length: 32 bytes (8) Flags: 0x018 (PSH, ACK) Window: 6371 [Calculated window size: 407744] [Window size scaling factor: 64] Checksum: 0xff07 [unverified]		0000 02 00 00 00 45 00 00 01 13 00 00 40 00 40 06 00 00E...@... 0010 7f 00 00 01 7f 00 00 01 1f 90 de 2b 50 20 1b f3+{.. 0020 b3 bd 1d 00 80 18 18 e3 ff 07 00 00 01 01 08 0a000001080a 0030 8d 18 7d a2 9e e6 79 09 48 54 50 2f 31 2e 31Y..HTTP/1.1 0040 20 32 30 30 20 4f 4b 0d 0a 43 6f 6e 6e 65 63 74 200 OK: Connect 0050 69 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d ion: keep-alive 0060 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a .Content-Length: 0070 20 31 35 39 0d 0a 0d 0a 3c 68 74 6d 6c 3e 0a 09 159...<html>.. 0080 3c 68 65 61 64 3e 0a 09 09 3c 74 69 74 6c 65 3e <head>...<title>.. 0090 4d 79 20 77 65 62 20 70 61 67 65 3c 2f 74 69 74 My web page/tit 00a0 6c 65 3e 0a 09 09 3c 6c 69 6e 6b 20 72 65 6c 3d lex...<link rel= 00b0 22 69 63 6f 6e 22 20 74 79 70 65 3d 22 69 6d 61 "icon" t type="ima 00c0 67 65 2f 78 2d 69 63 6f 6e 22 20 68 72 65 66 3d ge/x-ico n" href= 00d0 22 66 61 76 69 63 6f 6e 2e 69 63 6f 22 3e 0a 09 "favicon.ico">.. 00e0 3c 2f 68 65 61 64 3e 0a 09 3c 62 6f 64 79 3e 0a </head>...<body>.. 00f0 09 09 3c 62 3e 3c 75 3e 68 65 6c 6c 6f 3c 2f 75 <<cu> hello/u 0100 3e 3c 2f 62 3e 0a 09 3c 2f 62 6f 64 79 3e 0a 3c >< /body>.< 0110 2f 68 74 6d 6c 3e 0a /html>.<
--	--	--

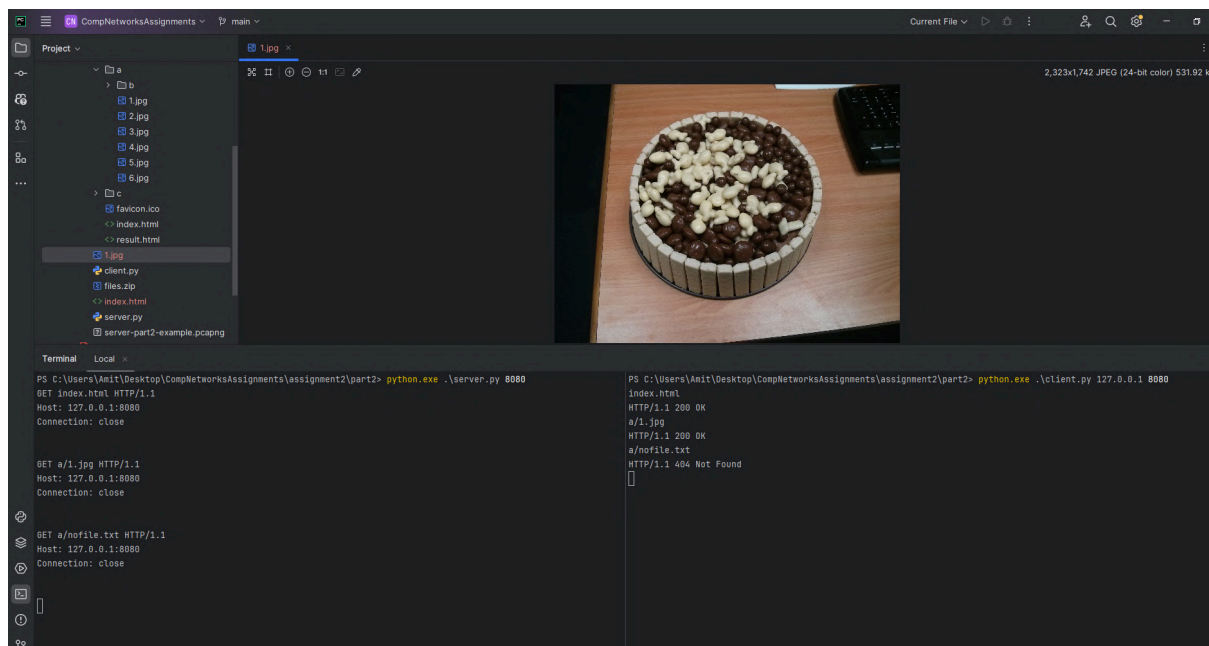
דוגמא של הרצה של הלקוח:

ניתן לראות פה הרצה של הלקוח ושל השרת מהתרגיל, הלקוח מבקש קבצים מהשרת, מקבל אותם בתור תשובה ושומר אותם מחדש על המחשב.

בתמונה הראשונה רואים שאין עדיין את הקבצים index.html ואת התמונה 1.



לאחר מכן אחרי שהלקוח מבקש את הקבצים האלו הם נשמרים וניתן לראות אותם בתור תמונה גם.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	45	60978 → 61001 [ACK] Seq=1 Ack=1 Win=10231 Len=1
2	0.000017	127.0.0.1	127.0.0.1	TCP	56	61001 → 60978 [ACK] Seq=1 Ack=2 Win=10232 Len=0 SLE=1 SRE=2
3	1.393950	127.0.0.1	127.0.0.1	TCP	45	60976 → 61002 [ACK] Seq=1 Ack=1 Win=10228 Len=1
4	1.393971	127.0.0.1	127.0.0.1	TCP	56	61002 → 60976 [ACK] Seq=1 Ack=2 Win=10085 Len=0 SLE=1 SRE=2
5	7.593636	127.0.0.1	127.0.0.1	TCP	56	49470 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
6	7.593686	127.0.0.1	127.0.0.1	TCP	56	8080 → 49470 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
7	7.593746	127.0.0.1	127.0.0.1	TCP	44	49470 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
8	7.593750	127.0.0.1	127.0.0.1	HTTP	112	GET index.html HTTP/1.1
9	7.593777	127.0.0.1	127.0.0.1	TCP	44	8080 → 49470 [ACK] Seq=1 Ack=69 Win=2619648 Len=0
10	7.594682	127.0.0.1	127.0.0.1	HTTP	271	HTTP/1.1 200 OK
11	7.594699	127.0.0.1	127.0.0.1	TCP	44	49470 → 8080 [ACK] Seq=69 Ack=228 Win=2619392 Len=0
12	7.594717	127.0.0.1	127.0.0.1	TCP	44	8080 → 49470 [FIN, ACK] Seq=228 Ack=69 Win=2619648 Len=0
13	7.594725	127.0.0.1	127.0.0.1	TCP	44	49470 → 8080 [ACK] Seq=69 Ack=229 Win=2619392 Len=0
14	7.594756	127.0.0.1	127.0.0.1	TCP	44	49470 → 8080 [FIN, ACK] Seq=69 Ack=229 Win=2619392 Len=0
15	7.594777	127.0.0.1	127.0.0.1	TCP	44	8080 → 49470 [ACK] Seq=229 Ack=70 Win=2619648 Len=0
16	14.849438	127.0.0.1	127.0.0.1	TCP	45	55883 → 61067 [ACK] Seq=1 Ack=1 Win=10206 Len=1
17	14.849480	127.0.0.1	127.0.0.1	TCP	56	61067 → 55883 [ACK] Seq=1 Ack=2 Win=10002 Len=0 SLE=1 SRE=2

הסבר קצר על חלק מהפקטות

פקטות 5-7 (TCP 3-Way Handshake): זה הפקטות שמבצעות את תהליך להקמת החיבור.

- **פקטה 5 (SYN):** הלקוח (פורט אקראי 49470) שולח בקשת סנכרון לשרת (פורט 8080) כדי ליזום חיבור.
- **פקטה 6 (SYN, ACK):** השרת מאשר את הבקשה (ACK) ושולח בקשת סנכרון משלו (SYN).
- **פקטה 7 (ACK):** הלקוח מאשר את סנכרון השרת.

פקטה 8 (HTTP GET Request): זאתי בקשה של האפליקציה. הלקוח שולח הודעת GET index.html לשרת. בפקטה זו הלקוח מבקש את הקובץ index.html שנמצא בתיקייה בשרת.

פקטה 9 (TCP ACK): הודעת אישור ברמת ה-TCP הנשלחת מהשרת ללקוח. השרת מאותת ללקוח שהוא קיבל את בקשת ה-GET שלו בהצלחה.

פקטה 10 (HTTP 200 OK Response): זאתי התשובה של השרת. השרת איתר את הקובץ index.html ושולח את תוכן הקובץ.

פקטה 11 (TCP ACK): הלקוח מאשר לשרת שהוא קיבל את התשובה (את פקטה 10).

פקטה 12 (TCP FIN, ACK): השרת יוזם את סגירת החיבור. הדגל FIN מסמן שהשרת סיים לשלוח מידע והוא מעוניין לנתק את ההתקשרות.

פקטות 13-15 (Connection Teardown): תהליך סיום החיבור משני הצדדים:

- **פקטה 13:** הלקוח מאשר (ACK) את בקשת הסגירה של השרת.
- **פקטה 14:** הלקוח שולח FIN משלו כדי לסגור את הצד שלו בחיבור.
- **פקטה 15:** השרת מאשר (ACK) את הסגירה של הלקוח.