**מגישים:**

**רון הכרי 312434830**

**דודי חן 300334034**

**קובץ ReadMe לתרגיל 3 בתתקשורת מחשבים**

בתרגיל זה, התבקשנו לכתוב אפליקציה שמתפקדת כמיין שרת שמקבל בקשות ומחזיר תגובות לפי סוג הבקשה שקיבל.

בתרגיל קיבלנו את הבקשה שבצורה של מחרוזת, פירקנו את המחרוזת ושמרנו את פרטי המחרוזת בstruct: request\_info: שבתוכו שומר איזה סוג בקשה בצורה enum הנקרא eMethod: GET is 1, POST is 2, HEAD is 3,PUT is 4, \_DELETE is 5,TRACE is 6, OPTIONS is 7.

הstruct שומר בנוסף לכך איזה Connection כללה הבקשה: אם זה keep\_alive or close.

בנוסף לכך, הstruct שומר char\*acceptType שהוא הcontent\_type של הבקשה למשל: text/html, text/plain… ושומרת על מנת להחזיר את הcontent\_type in response.

אם אין content type in request אז התוכנית תחפש אם יש שדה בבקשה הנקרא accept וערכו של הcontent-type יקבע לפי הaccept אם אין content-type.

ואם אין שדה בבקשה הנקרא accept and content-type, (**בכל סוג של בקשה חוץ מבקשת Trace!**) השרת יחזיר status code של **415 Unsupported Media**.

למשל:

1. כאשר הערך לא תקין:

Graphical user interface, text, application

Description automatically generated



1. כאשר ה-header בשם Content-Type חסר:

Text

Description automatically generated



Request\_info נכיל בתוכו גם שני שדות הנקראים char\*resourch\_path and char\*fullpath.

Resourch\_path אמור לשמור את שם המשאב של קובץ המקור וfullpath אמור לשמור את שם המשאב של קובץ המתורגם במקרה של בקשת GET למשל אם שם המשאב המקור:

Index.html אז שם משאב המתורגם: index\_en.html באנגלית index\_fr.html בצרפתית וindex\_he.html בעברית.

יתר על כן, request\_info מכיל בתוכו char\*bodyMessage שתפקידו לשמור את ה body של הבקשה שמגיע אחרי שורת רווח אחרי header של הבקשה.

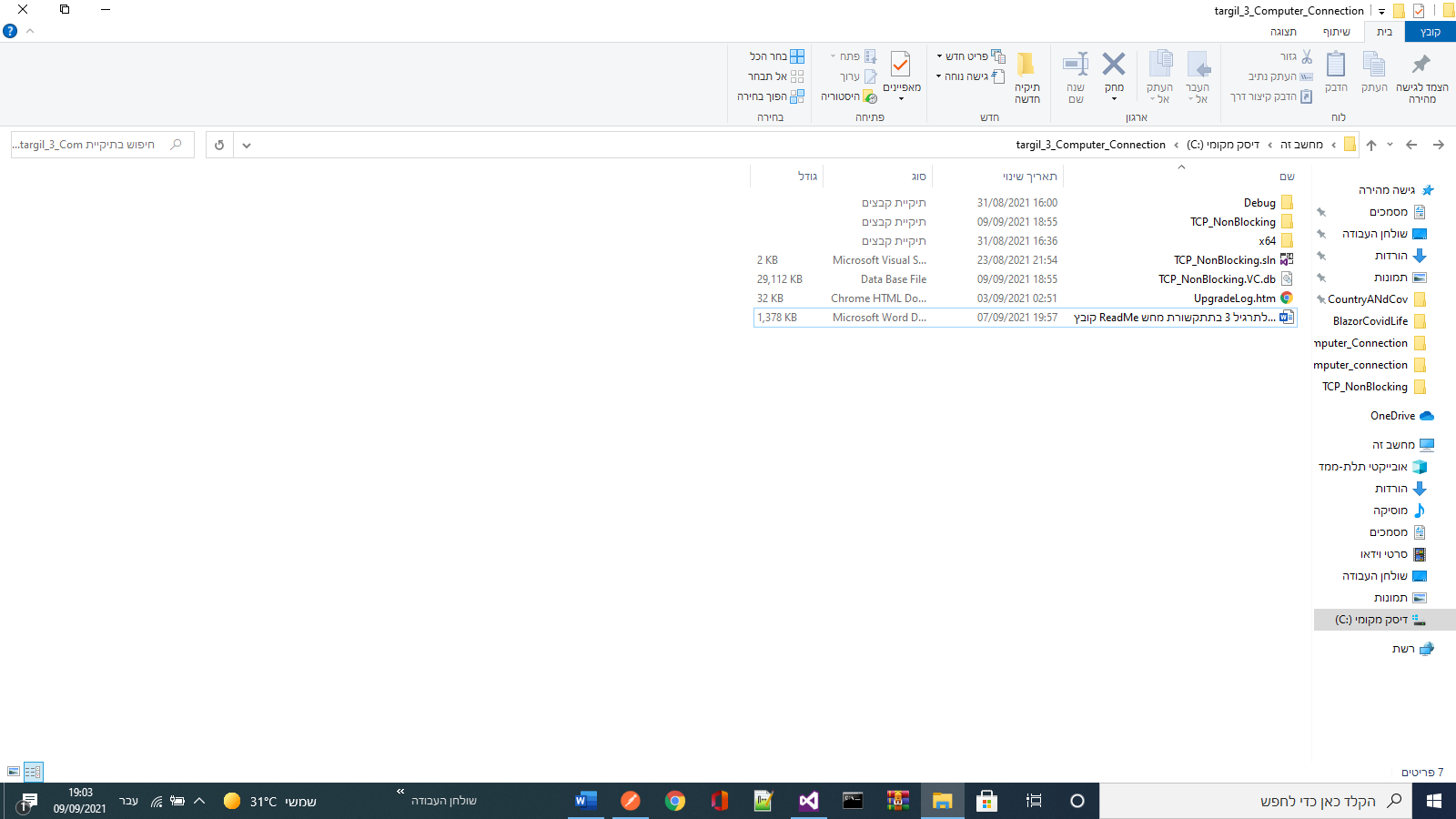
במקרה של בקשה מסוג TRACE, request\_info שומר גם שדה הנקרא: string res ששומר את הבקשה עצמה כדי לשקף אותה ומצוטט את הבקשה.

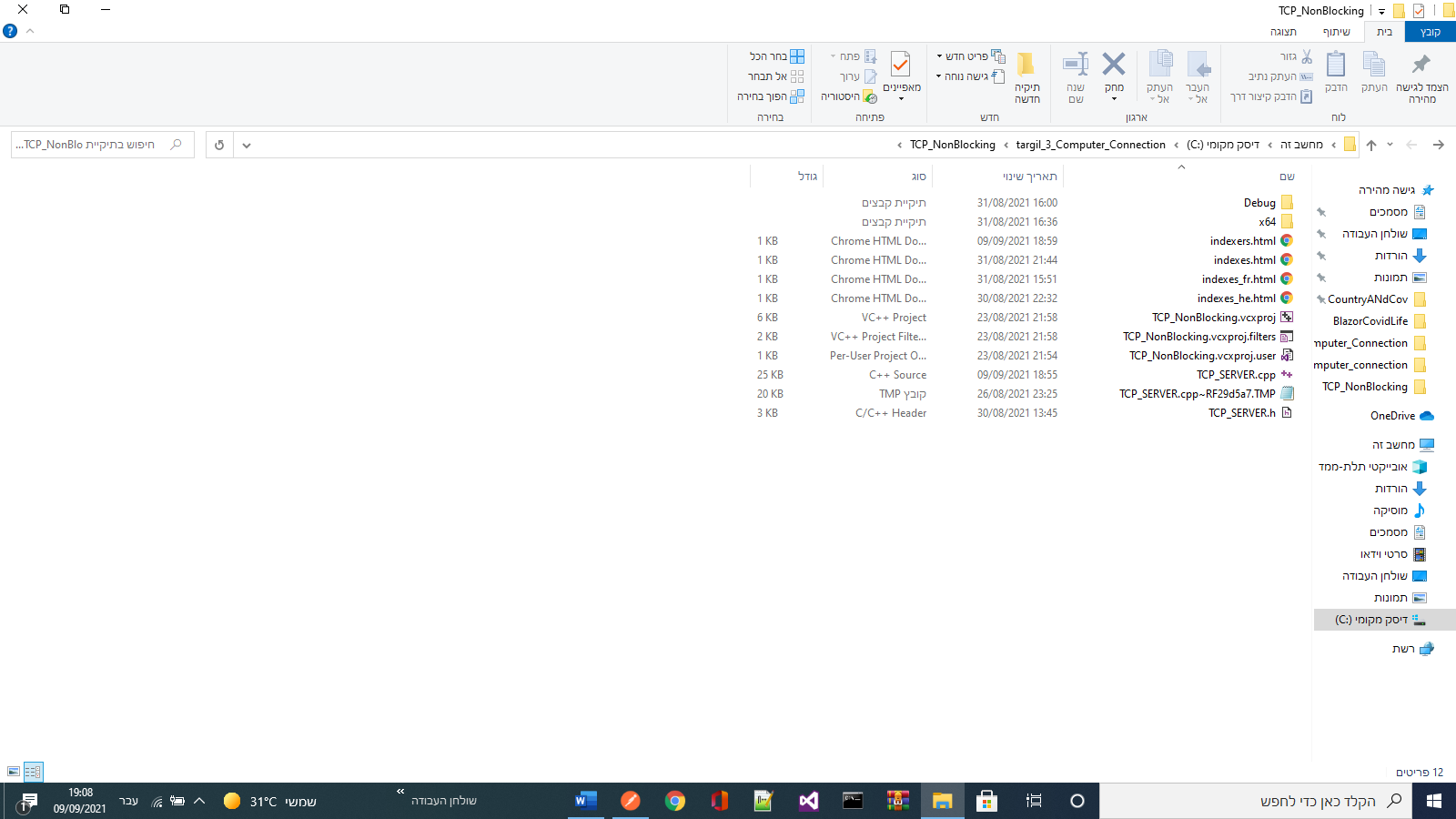
**קבצים ומבנה תיקיות**

כל קבצי :html

Indexers.html,indexes.html, indexes\_he.html, indexes\_fr.html.

צריכים להיות ממוקמים באותה תיקייה כמו TCP\_SERVER.cpp ו TCP\_SERVER.h.

למשל זו התיקייה שבו נמצא קובץ הsin, בנוסף לכך נמצא התיקייה TCP\_NONBlocking שבתוך תיקייה זו נמצאים כל קבצי html וקבצי: TCP\_SERVER.cpp ו TCP\_SERVER.h.

בתוך תיקייה TCP\_NONBlocking נמצאים:

**הresponse**

בכל הבקשות מה שמשותף זהו הheader of response והוא מורכב:

שורה ראשונה- HTTP 1.1+status\_code.

שורה שנייה- Connection.

שורה 3- תאריך.

שורה 4- content\_length.

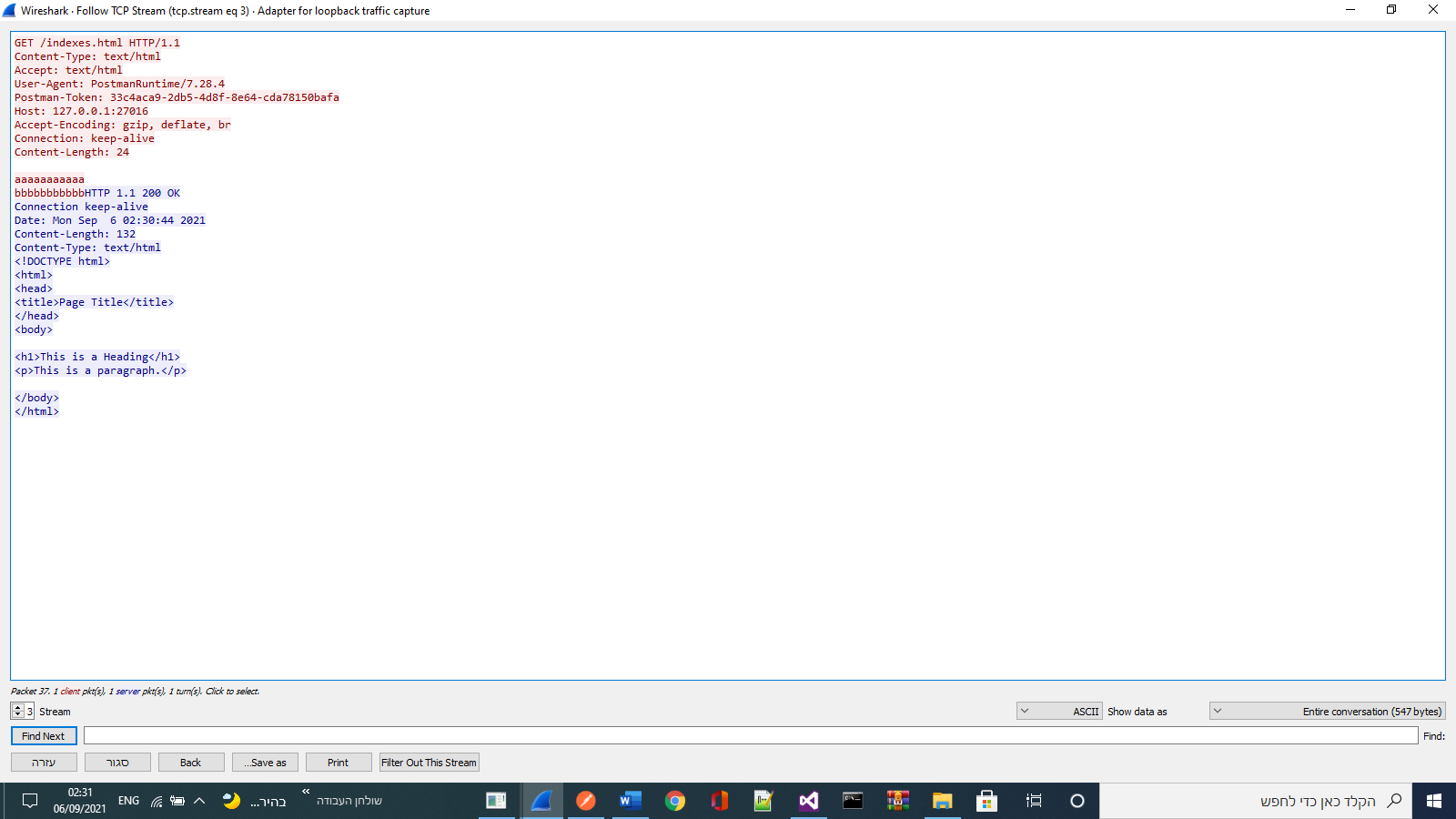
שורה 5- content\_type.

**הודעת GET**

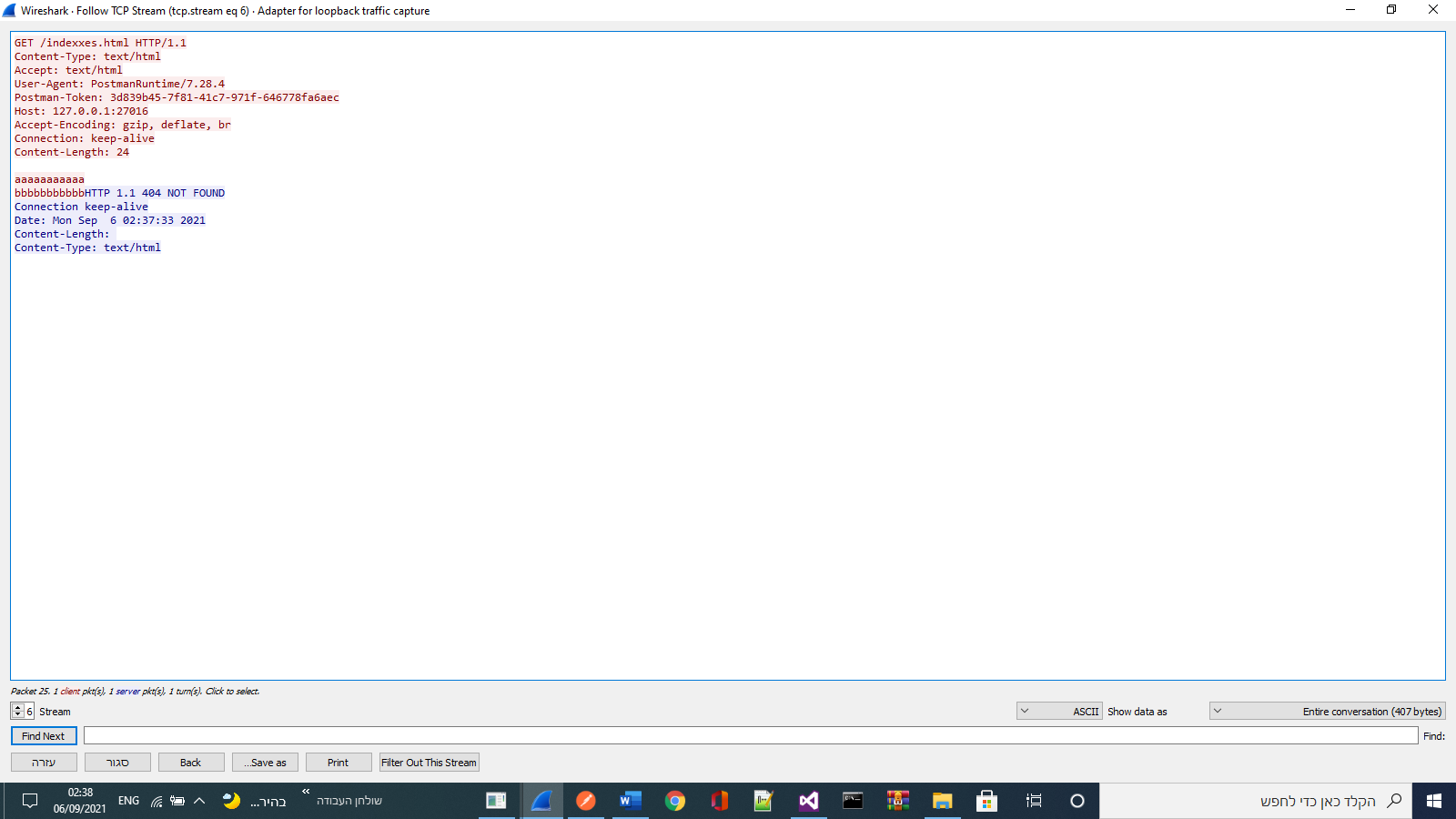
הresponse מחזיר את מה שיש בheader וגם מה שיש בתוכן הקובץ לפי המשאב שביקשו.

בהתחלה יבדקו האם יש קובץ לפי הfullpath ואם הקובץ קיים יחזיר בנוסף לheader את תוכן הקובץ המתורגם לפי השפה שביקשו ואם לא קיים , התוכנית תבדוק לפי הresourch\_file , אם הקובץ קיים, תחזיר את תוכן הקובץ המקורי אחרת תחזיר רק את הheader עם status\_code 404 NOT FOUND ואם הקובץ קיים, התוכנית תמנה את כמות האותיות בקובץ ותשמור בcontent\_length ותשמור את תוכן הקובץ בbodyMessage ותעדכן את status\_code 200 OK.

מריצים הודעת GET למשל דרך הbrowser or postman ורושמים: 127.0.0.1:27016/index.html ואם זה עם שפה: 127.0.0.1:27016/index.html?lang=fr.

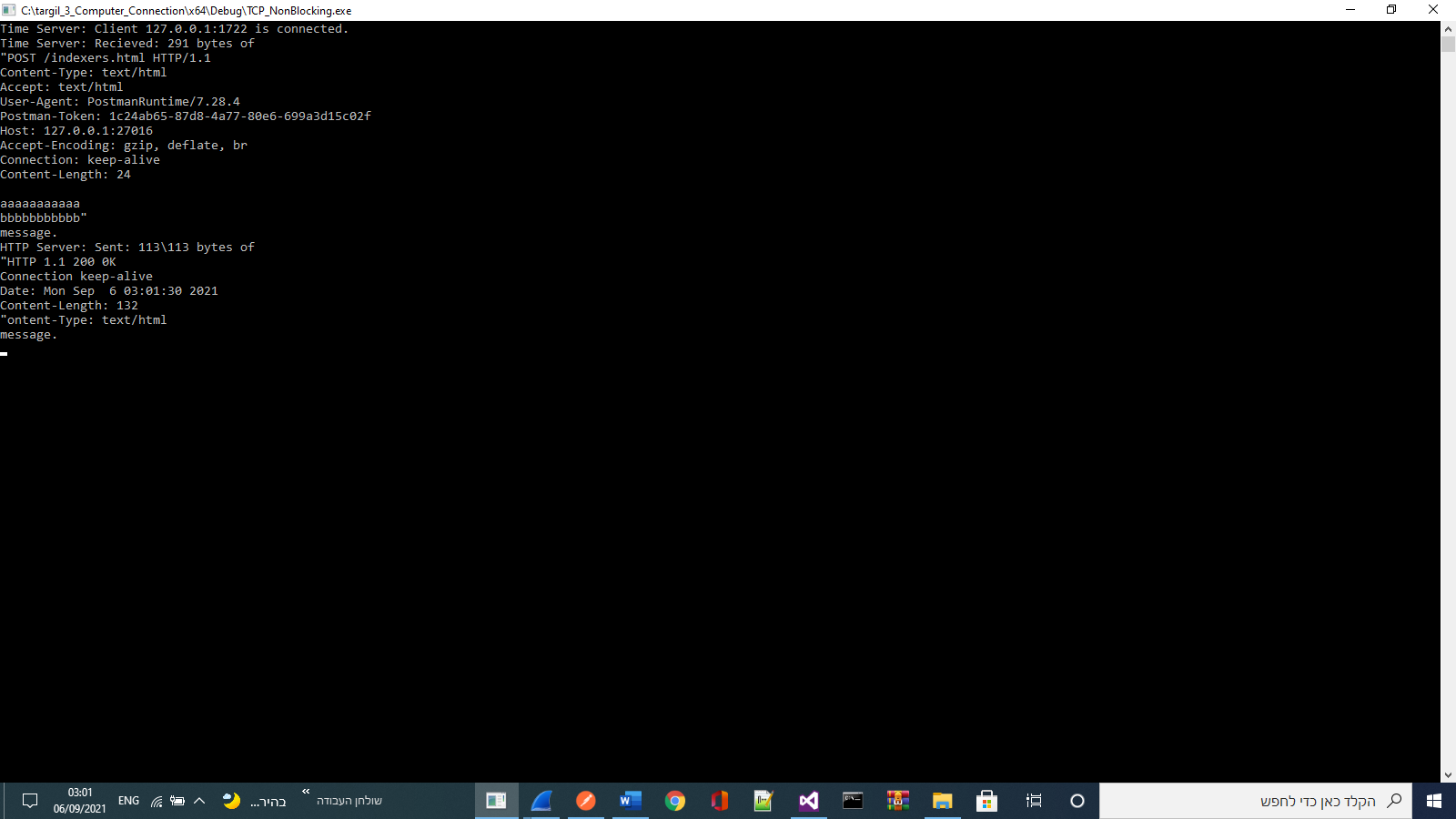
למשל 127.0.0.1:27016/indexes.html

למשל 127.0.0.1/indexer.html(קובץ שלא נמצא)

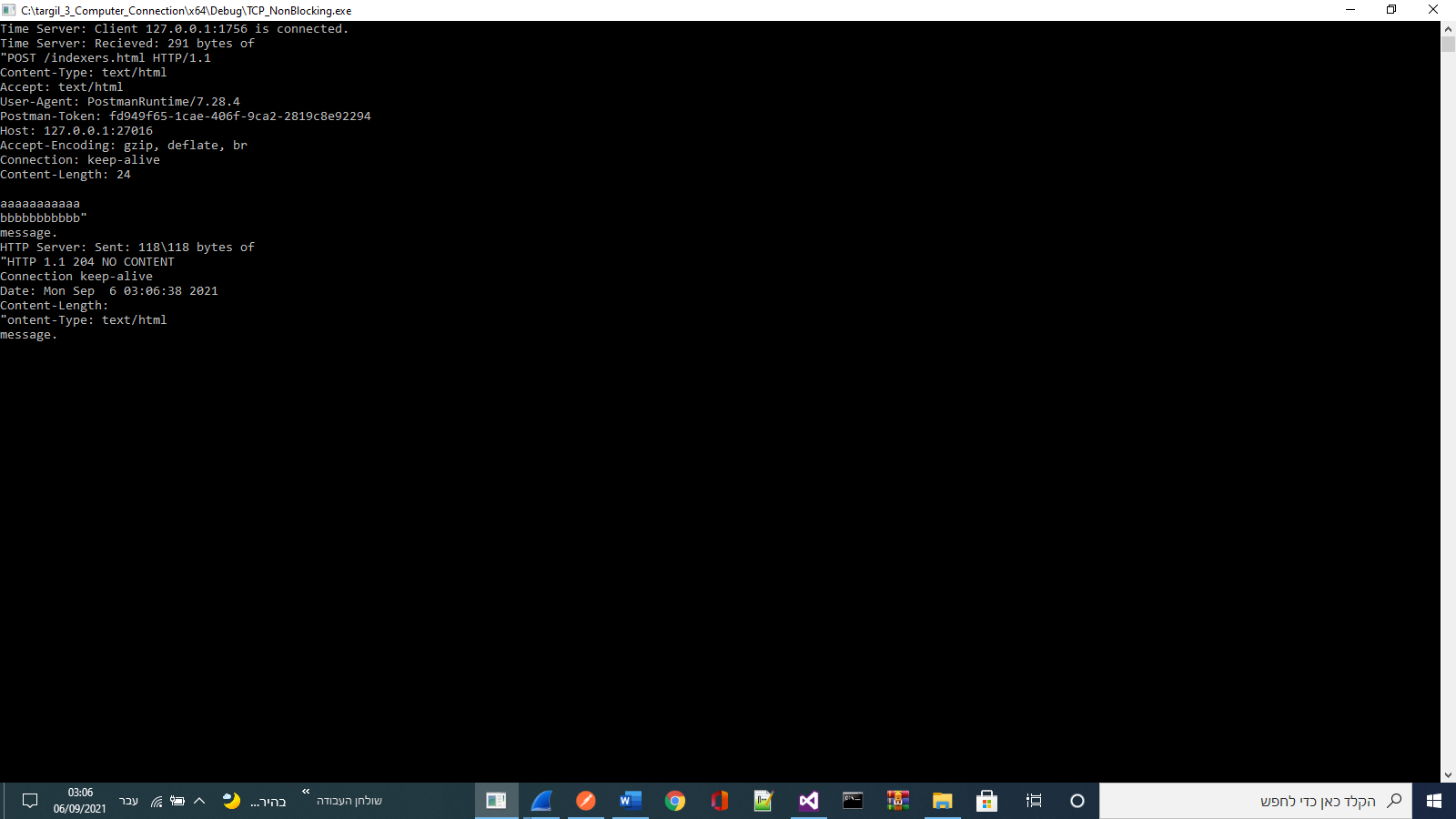


**הודעת POST**

בבקשה זו, התוכנית מציגה בקונסול את הresponse שמכיל רק את הheader ובנוסף לכך משנה את הקובץ, למשל אם רושמים 127.0.0.1:27016/index.html והקובץ קיים אזי הבקשה תשרשר את הbody בסוף הקובץ ותחזיר status\_code 200 OK ואם הקובץ ריק מתוכן, תחזיר status\_code 204 NO CONTENT ותשתול את הbody של הבקשה בקובץ ואם הקובץ לא קיים הstatus\_code 201 CREATED תיצור קובץ חדש בשם שניתן לו ותשתול את הbody של הבקשה בקובץ החדש שנוצר.

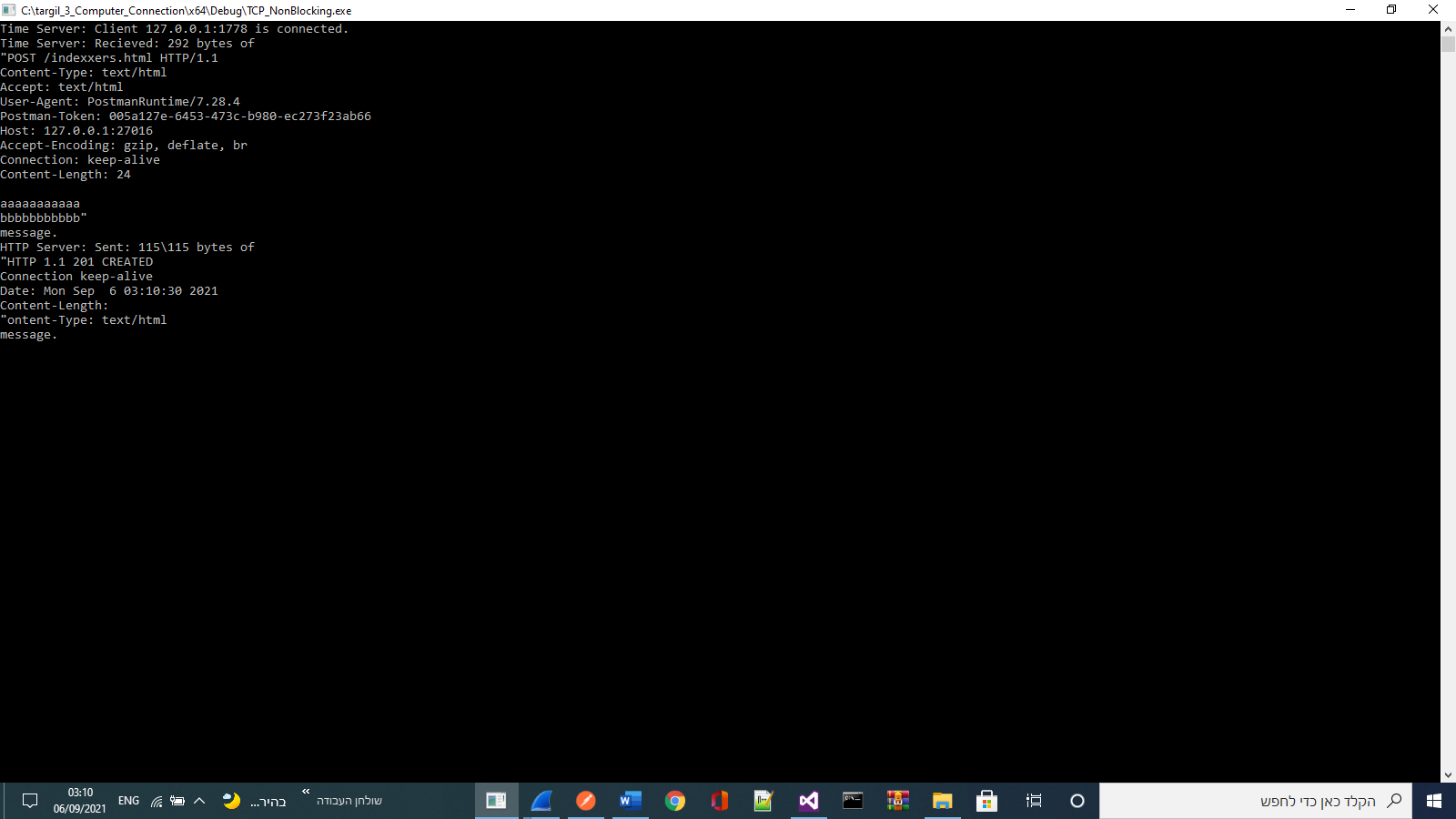
למשל: בpostman

127.0.0.1:27016/indexers.html(קובץ שקיים):

והוסיפה את הbody בסוף הקובץ.

למעלה-אם הקובץ ריק אז מחזירה status\_code 204 NO CONTENT ומוסיפה את body בקובץ.

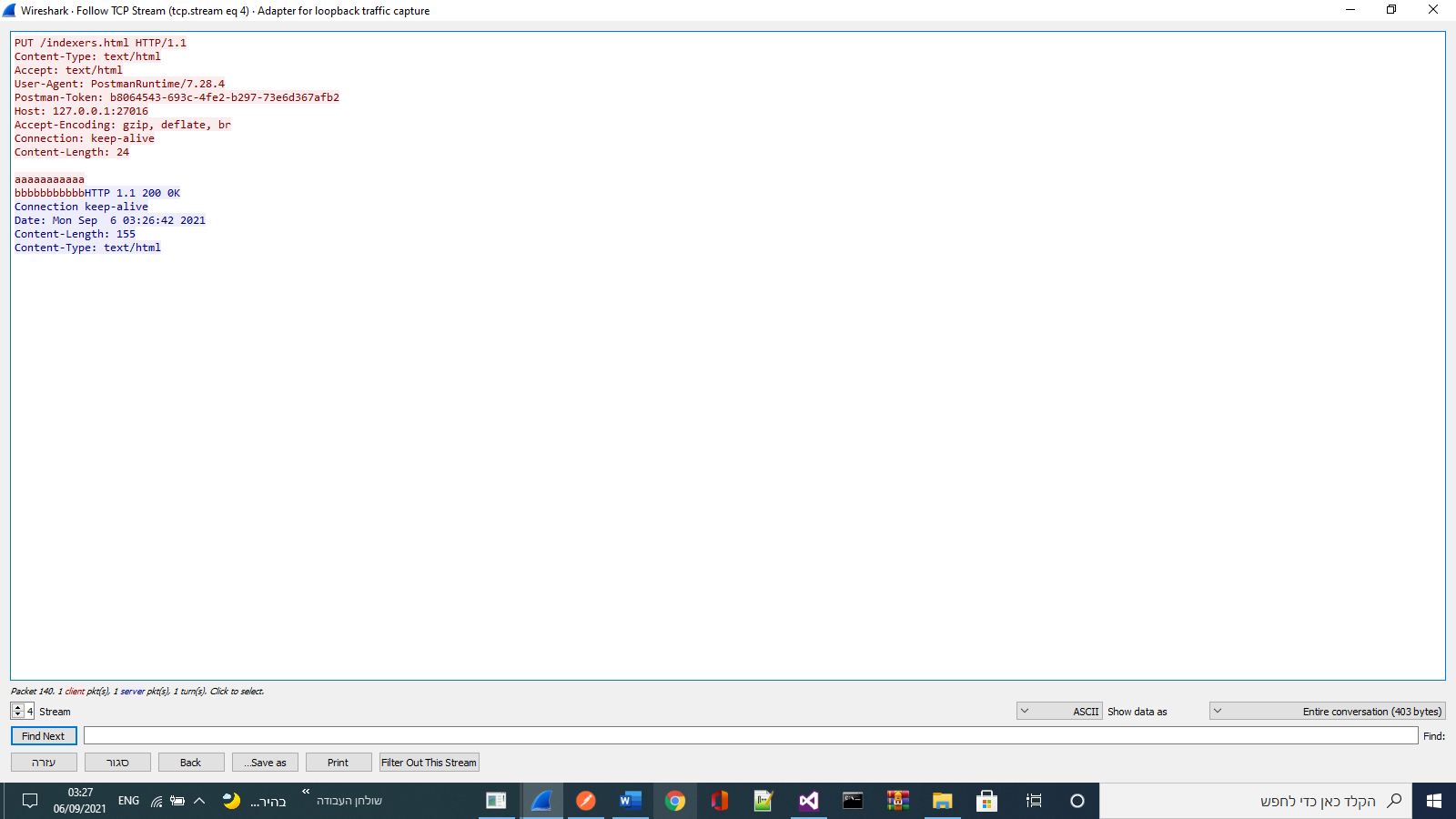
אם הקובץ לא קיים:

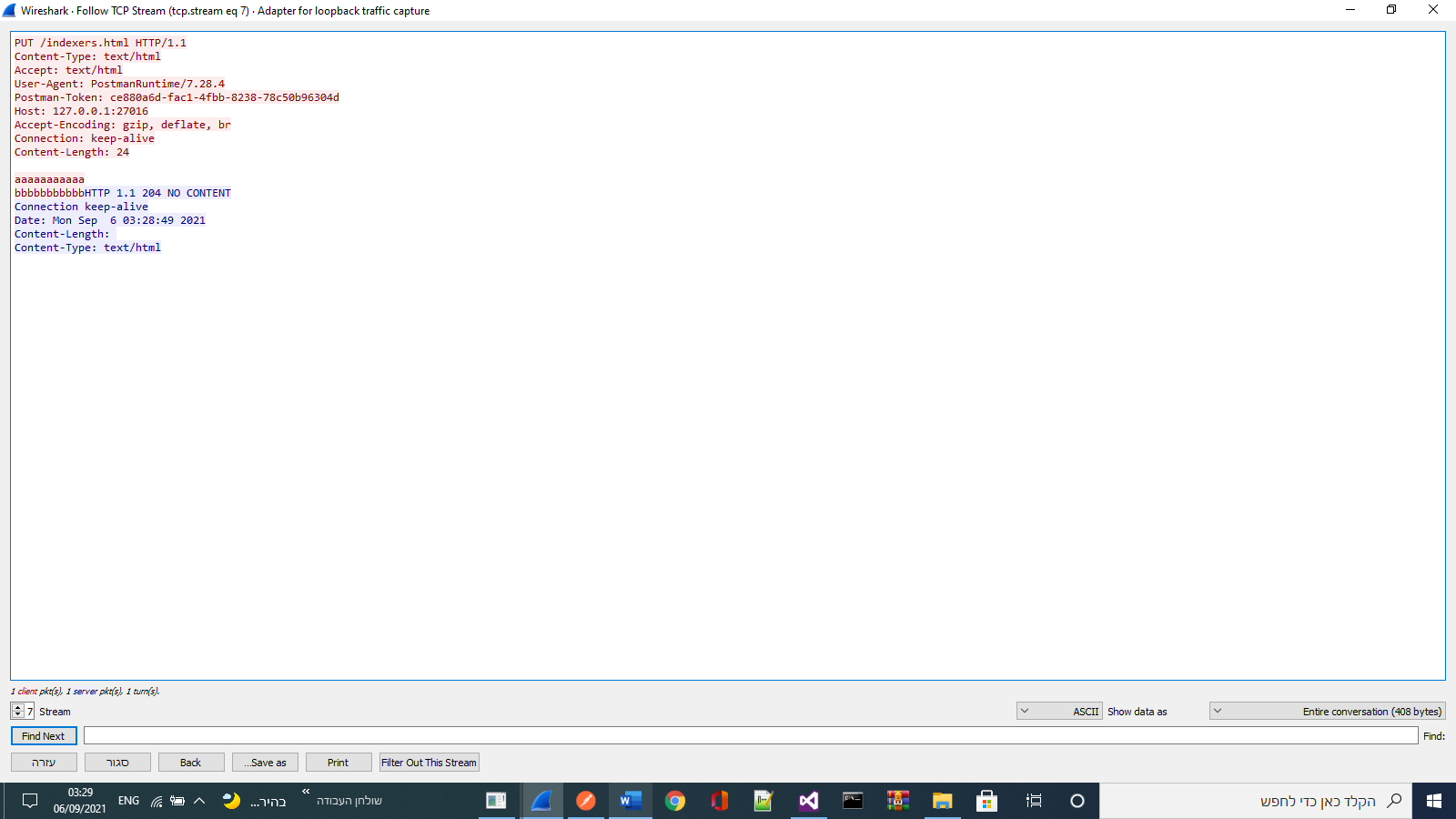


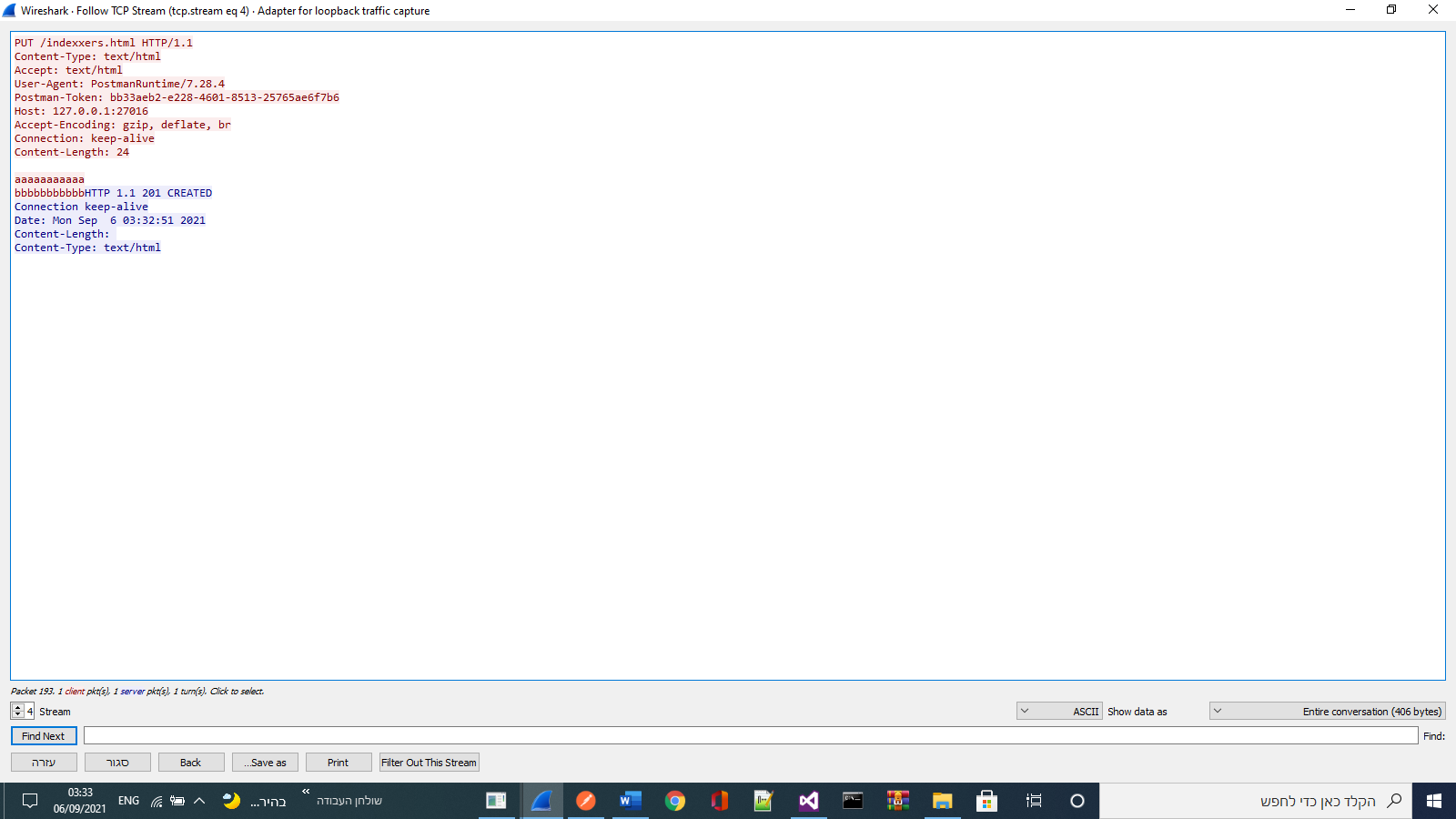
יוצרת אותו ושותלת בו את הbody של הבקשה.

**הודעת PUT**

כמו בPOST, אך במקום להציג את הresponse בקונסול, התוכנית שולחת את ה response.

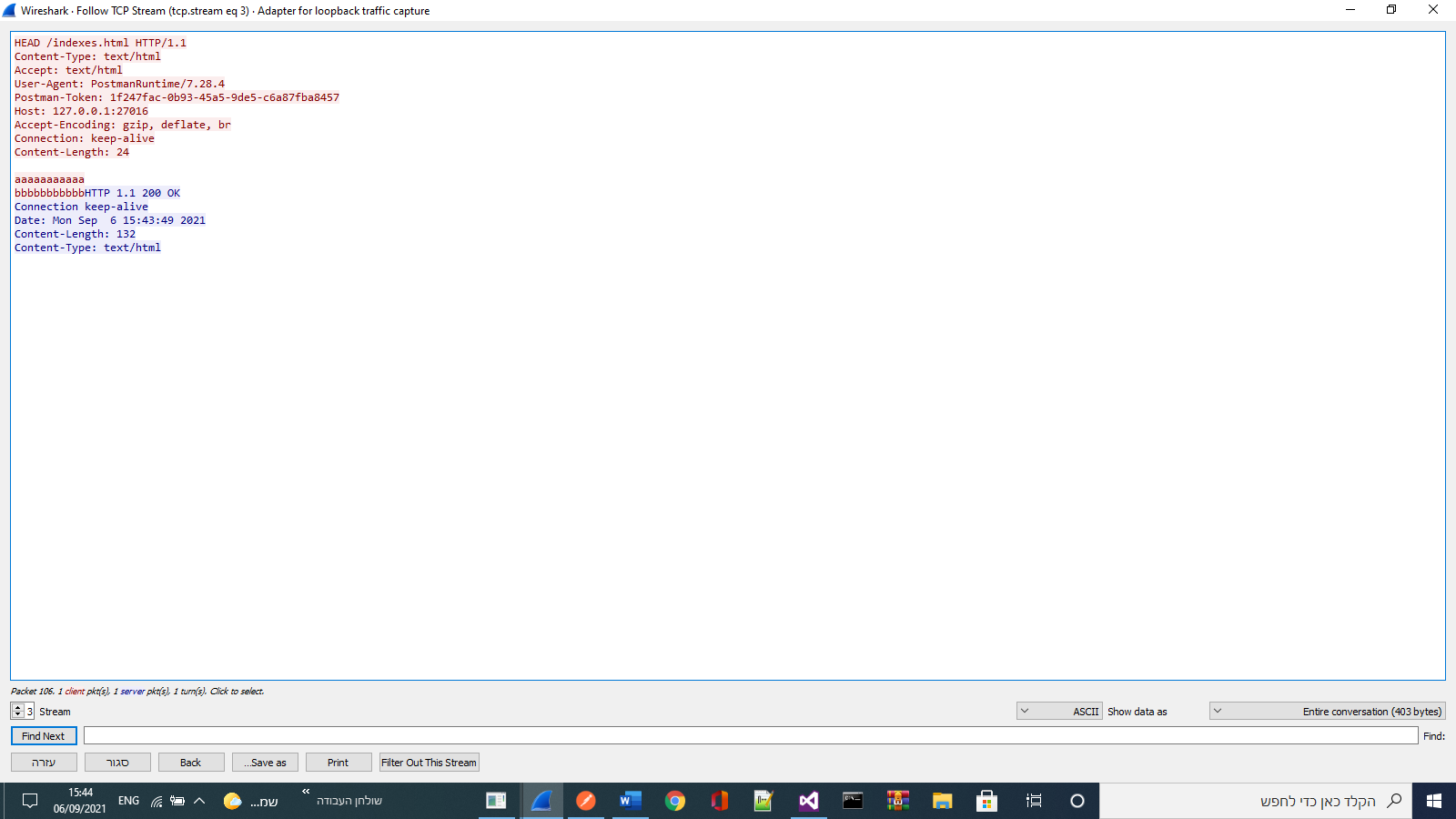
אם המשאב קיים ויש בו תוכן:

אם הקובץ קיים ואין בו תוכן:

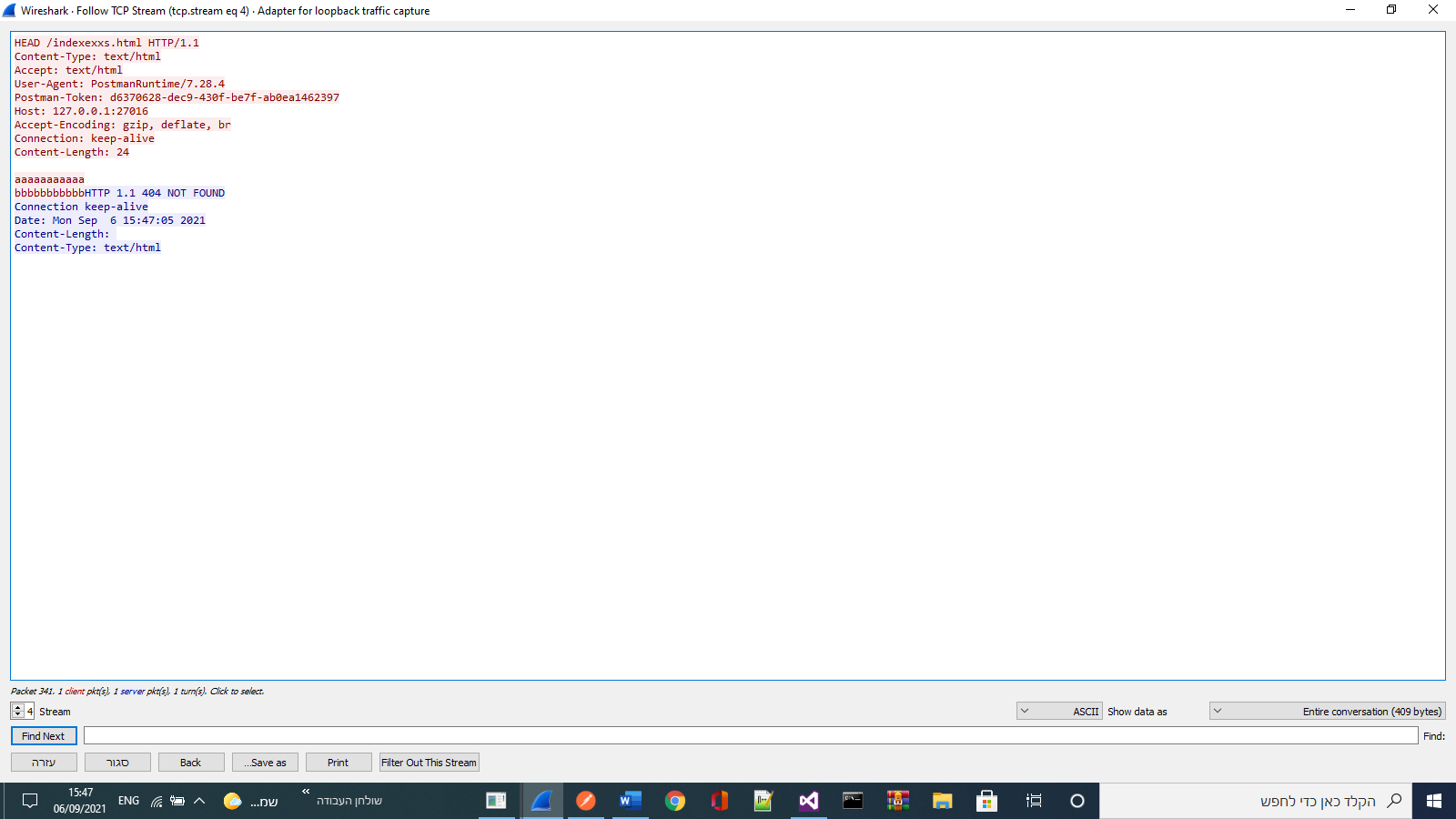
ואם הקובץ לא קיים:

**בקשת HEAD**

בבקשה זו, התוכנית מציגה רק את הheader של הresponse, אם הקובץ נמצא, הstatus code is 200 OK, אם הקובץ לא נמצא הstatus code is 404 NOT FOUND.

למשל בpostman נרשום בקשה head ונרשום: 127.0.0.1:27016/indexes.html והקובץ נמצא

אם הקובץ לא נמצא:



**בקשת DELETE**

בבקשה זו, התוכנית תבדוק אם הקובץ קיים ויש בו תוכן והמחיקה הצליחה, תחזיר status code 200 OK, אם הקובץ קיים אבל אין בו תוכן הstatus code is 204 NO CONTENT ואם הקובץ לא קיים או מחיקת הקובץ לא הצליחה, הstatus code is 202 FILE NOT SUCCESSFULLY DELETED.

למשל אם בpostman מבקשים בקשת DELETE ורושמים: 127.0.0.1:27016/indexes.html ומחיקת הקובץ הצליחה והקובץ קיים:

Text

Description automatically generated

ואם הקובץ לא קיים:

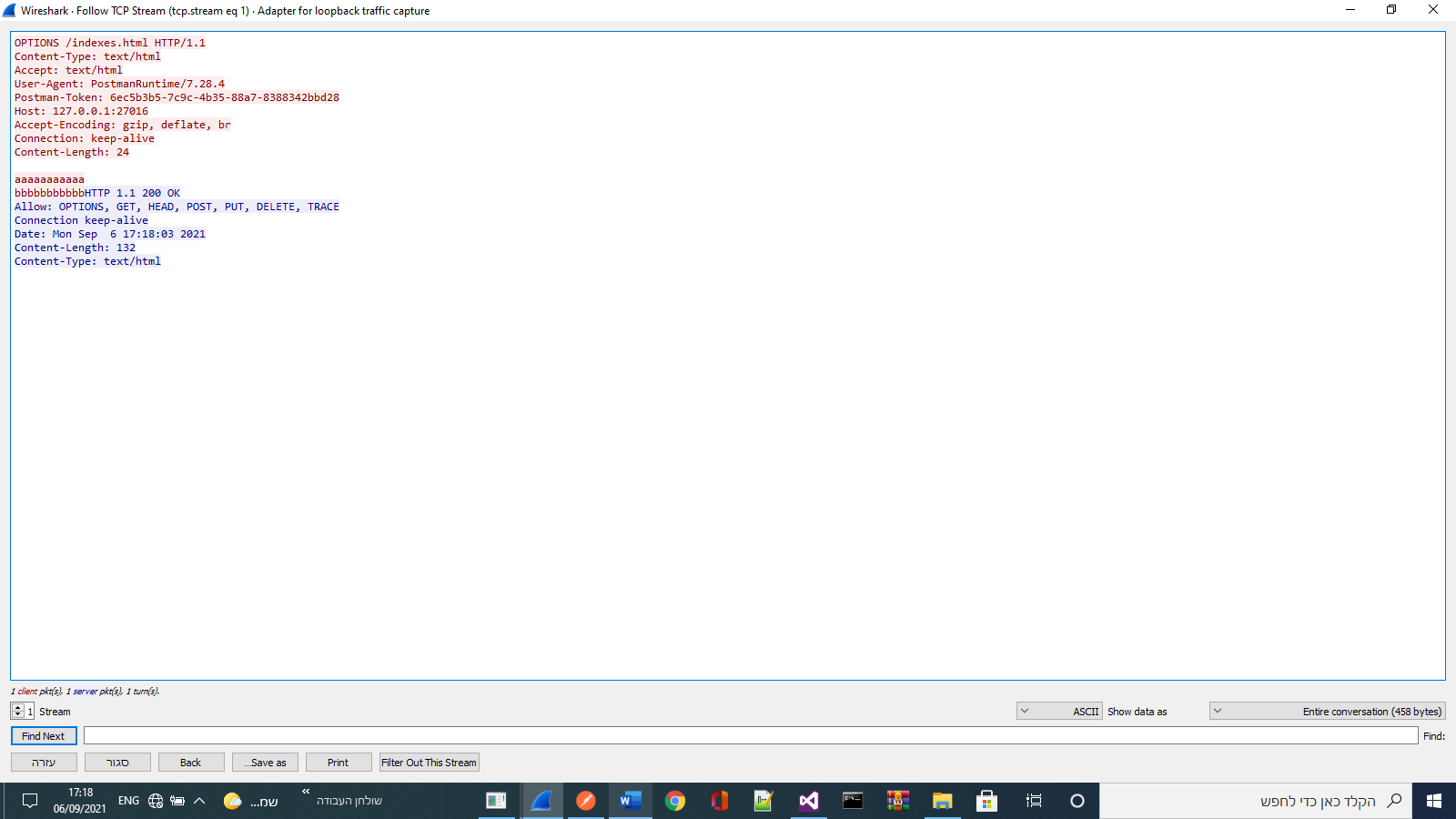
Text

Description automatically generated

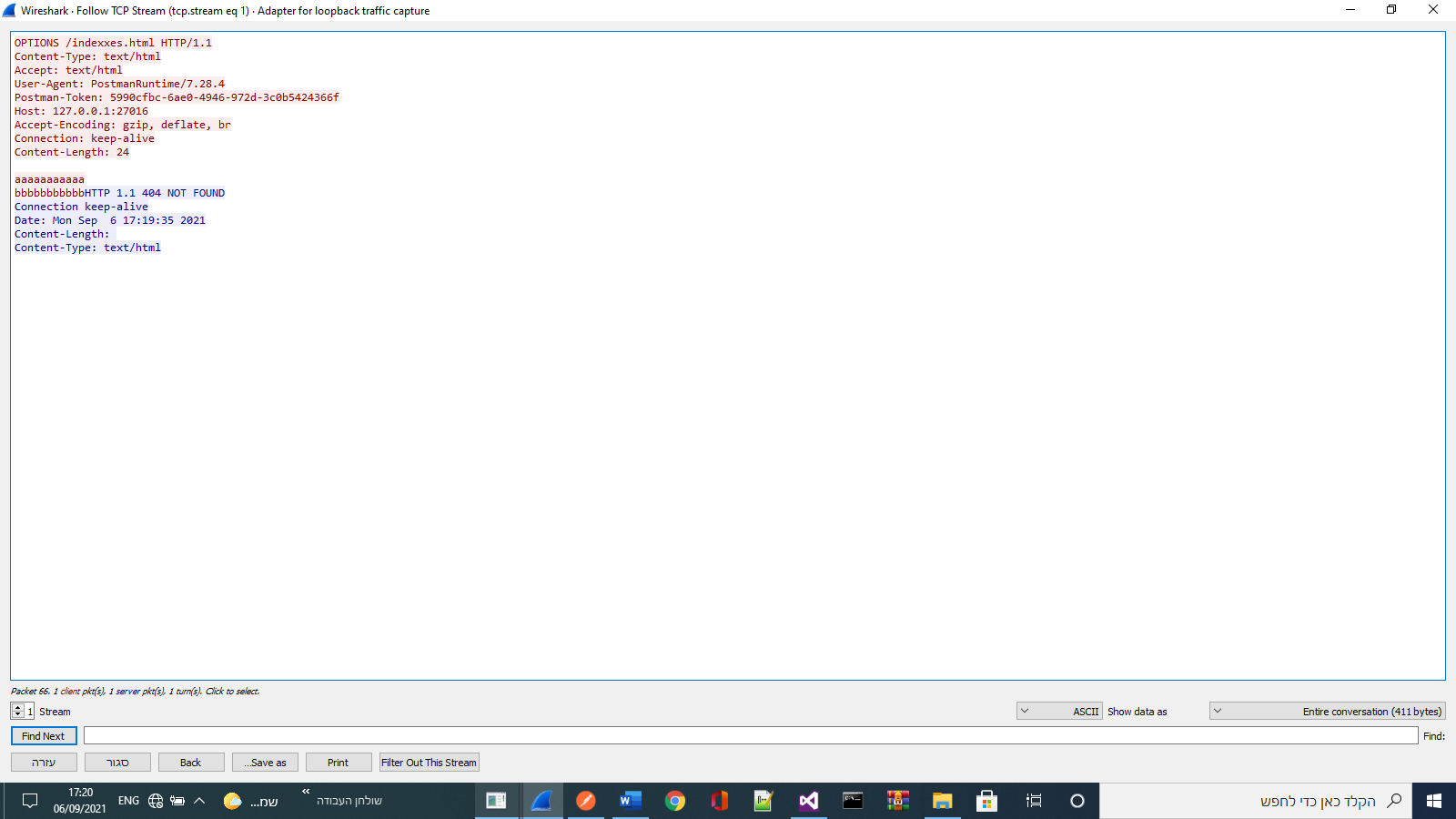
**בקשת OPTIONS**

בבקשה זו, בודקים אם הקובץ קיים, אם לא קיים הstatus code is 404 NOT FOUND, והheader of response ישאר כמו שהוא, אחרת אם הקובץ קיים, הstatus code is 200 OK, ובין השורה הראשונה בheader of response שרשום HTTP 1.1+status code לשורה השנייה תתווסף שורה שבה רשום האופציות: Allow: OPTIONS, GET,HEAD,POST,PUT,DELETE,TRACE.

בpostman, אם נעשה בקשה OPTIONS, ונרשום 127.0.0.1:27016/indexes.html:



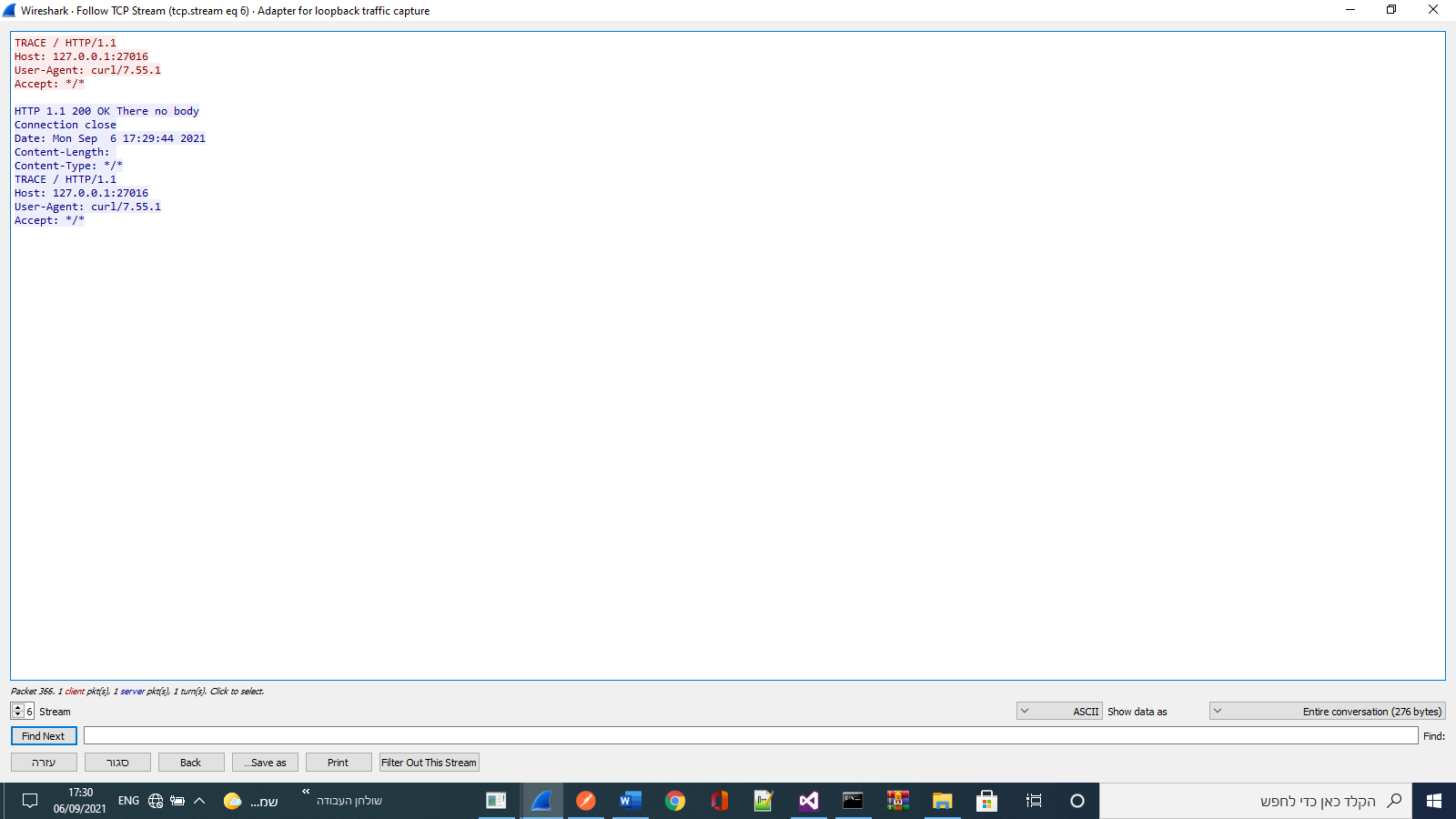
אם הקובץ לא נמצא:



**בקשת TRACE**

בבקשה זו, התוכנית תשתול ותצוטט את מה שהתקבל בrequest אחרי הheader of response.

אם יש body in resquest, הstatus code is 200 0K HAVE A BODY IN REQUEST. אם אין body in request, הstatus code is 200 OK THERE NO BODY.

בבבקשה TRACE השתמשנו בcommand ורשמנו: curl –insecure -v -X TRACE 127.0.0.1:27016 :