Ex3- Client&Host Sockets:

In this exercise we asked for to program a TCP client&server socket in Python.

At first, we wrote our Host's socket to be open for listening to our client socket, giving it our IP and Port.

```
serverSocket = socket(AF_INET, SOCK_STREAM) # first of all, we create
our server socket using SOCK_STREAM
serverSocket.bind(("192.168.1.155", 80))    # which related to TCP.
Than we are binding between our IP address
serverSocket.listen(1)                      # and the port which the
message sent in. and finally open our serverSocket
#f or listening until something will happen.
```

Than we create our client Socket in an infinity loop.
In our client Socket we accept the incoming connection request from the server.

```
while True:
    print('Ready to serve...')
    connectionSocket, addr = serverSocket.accept() # The accept
method of Python's socket class,
                                                   # accepts an
incoming connection request from a TCP server
```

Now, according to what we asked for, we using in try&except method to see if the message we would like to send in sent or not.

Try:

In the try part we are sending the "OK" message and the message content.

```
try:
    message = connectionSocket.recv(1024)   #First of all, we are
giving a range of 1024 bits for
    filename = message.split()[1]           # the message we would
like to send.
    f = open(filename[1:])                  # we are spliting our
message and then take the exact
    outputdata = f.read()                   # we want in our
outputdata
    connectionSocket.send("\nHTTP/1.1 200 OK\n\n".encode())    #
sending the ok message if everything went OK
    for i in range(0, len(outputdata)):                        # like
OK message for a regular HTTP request
        connectionSocket.send(outputdata[i].encode())          # in
Wireshark. and finally send our message content
    connectionSocket.send("\r\n".encode())                     # and
close our client server
```

Exception:

If the try part went wrong (can be a wrong file name, connection issues etc.) we are going into the except part which send a 404 message.

```
except IOError:                                                # if
something went wrong in our try part
    err="404 NOT FOUND"                                        # we
are sending a "404 not found" message
    connectionSocket.send(err.encode())                        # as
required and than closing our client server
    connectionSocket.close()
```

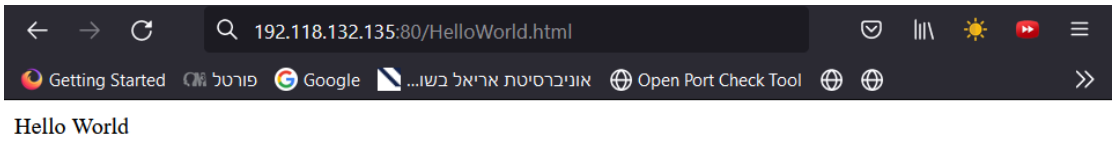
Does it actually work?

Now let us use Wireshark to see if every went OK or we might have a 404 problem (we will see both of the cases).

First of all, we create an html file which is content is "HelloWorld", it look like this:

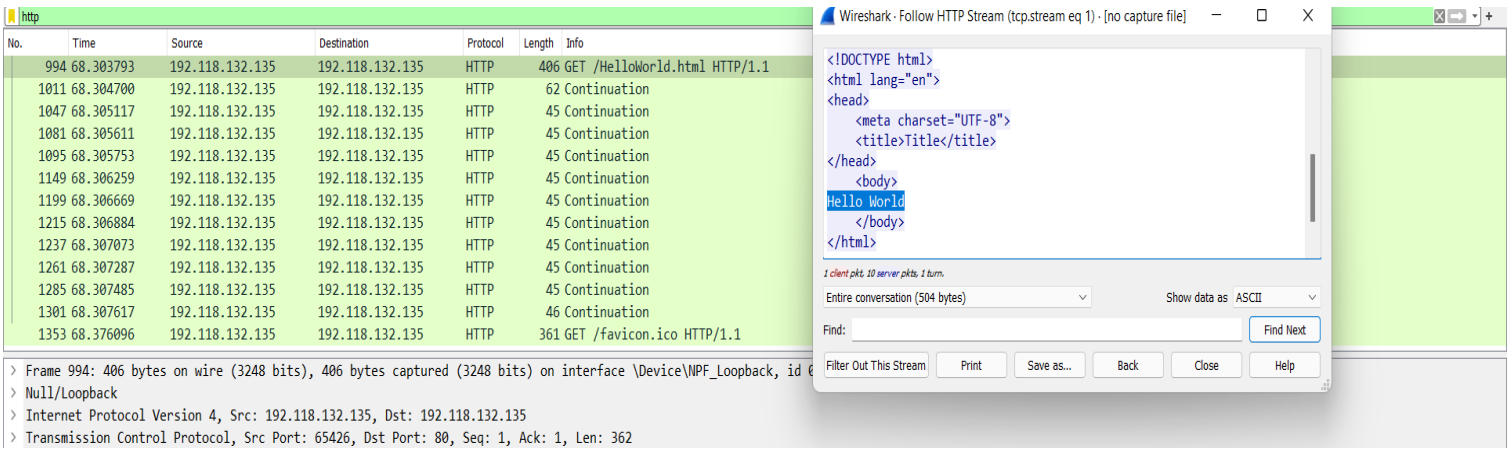
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
    <body>
Hello World
    </body>
</html>
```

Now let us see if the Wireshark captured these html file: first we run our python program, than we insert this URL into our browser : 192.118.132.135:80/HelloWorld.html



Hello World

And as we can see everything is ok, and we got our Hello World message, but does the Wireshark capture it?
Yes it did!

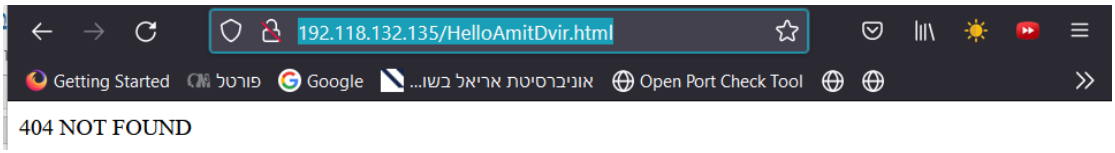


As we can see, we see the html file which has been created, we see the HTTP/1.1 message the right port that we gave etc.

Now, we will change the file name in the URL and see if we could capture a 404 problem:

Again, we ran our Python program but in this time, we put a different name in the URL into our browser:
http://192.118.132.135/HelloAmitDvir.html

And we got the "404 not found" message:



And in the Wireshark we got only the traffic in this fictive URL but not the html file in it: