Networks Final Project

TCP using UDP

CCE 2026

Submission Date: 18/5/2025

Name	ID
Rowan Ahmed Nazmi	8382
Mohamed Hesham Shabana	8349
Seif Maged Farouk	8387

Introduction:

The task is to design and implement a system that simulates TCP packets using a UDP connection by extending the user space of your application while maintaining reliability and supporting the HTTP protocol on top of UDP. UDP is a connectionless protocol that does not guarantee the delivery or order of packets. This presents several challenges when attempting to transfer data reliably.

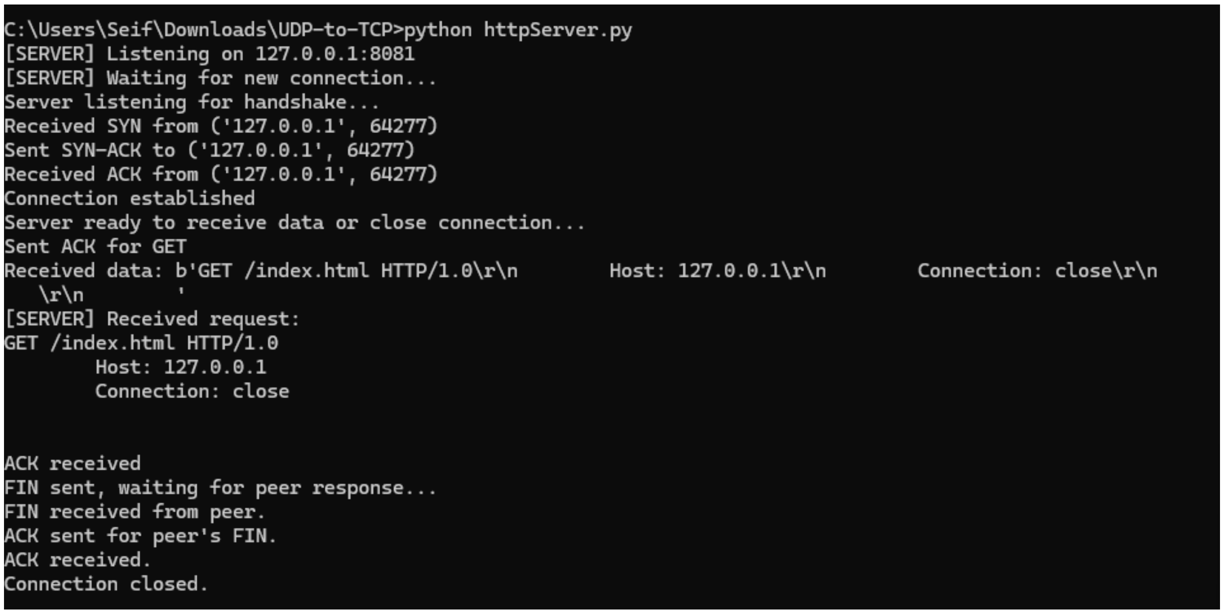
Features Implemented:

- Checksum for Error Detection
- Stop-and-Wait Protocol
- Packet Loss and Corruption Simulation
- Timeout and Retransmission of Lost Packets

Sample Runs:

Sample #1:

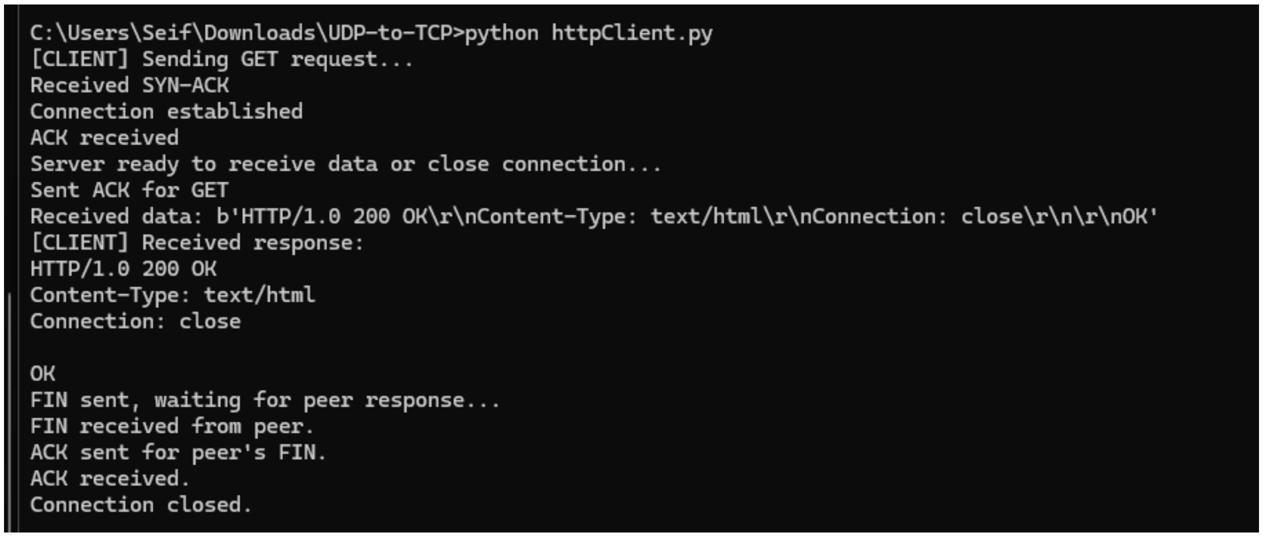
Server Side:



```
C:\Users\Seif\Downloads\UDP-to-TCP>python httpServer.py
[SERVER] Listening on 127.0.0.1:8081
[SERVER] Waiting for new connection...
Server listening for handshake...
Received SYN from ('127.0.0.1', 64277)
Sent SYN-ACK to ('127.0.0.1', 64277)
Received ACK from ('127.0.0.1', 64277)
Connection established
Server ready to receive data or close connection...
Sent ACK for GET
Received data: b'GET /index.html HTTP/1.0\r\n        Host: 127.0.0.1\r\n        Connection: close\r\n
   \r\n        '
[SERVER] Received request:
GET /index.html HTTP/1.0
        Host: 127.0.0.1
        Connection: close


ACK received
FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

Client Side:



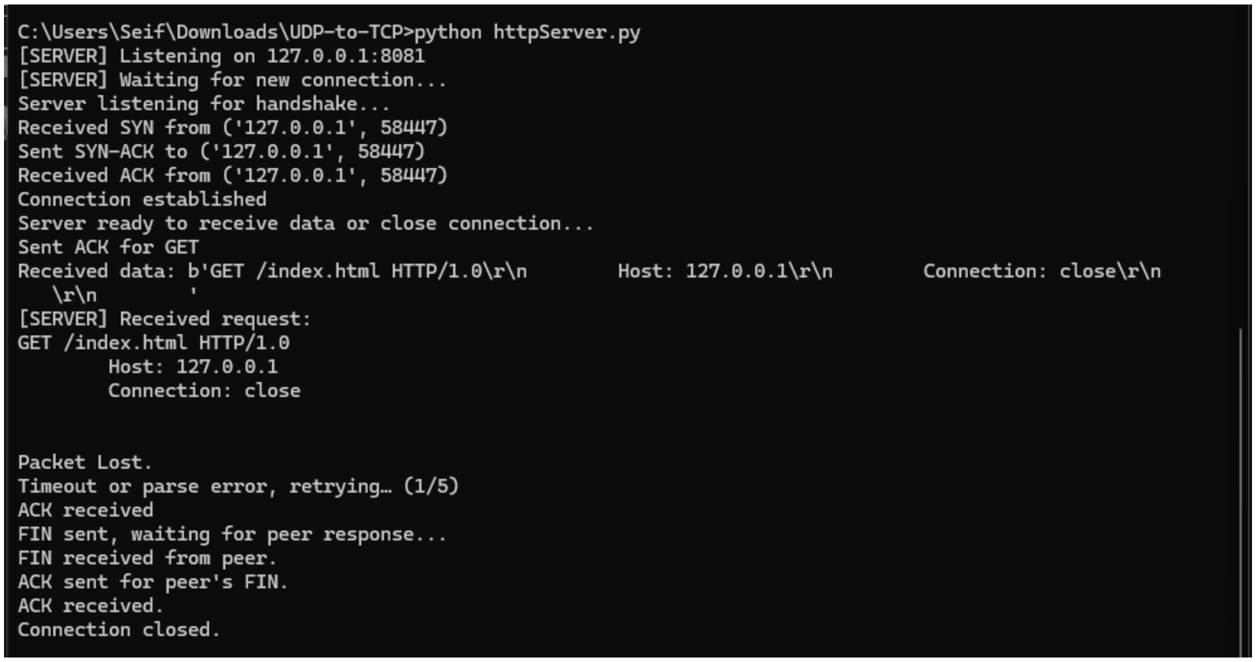
```
C:\Users\Seif\Downloads\UDP-to-TCP>python httpClient.py
[CLIENT] Sending GET request...
Received SYN-ACK
Connection established
ACK received
Server ready to receive data or close connection...
Sent ACK for GET
Received data: b'HTTP/1.0 200 OK\r\nContent-Type: text/html\r\nConnection: close\r\n\r\nOK'
[CLIENT] Received response:
HTTP/1.0 200 OK
Content-Type: text/html
Connection: close

OK
FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

In this sample run, there were no losses or corruptions.

Sample #2:

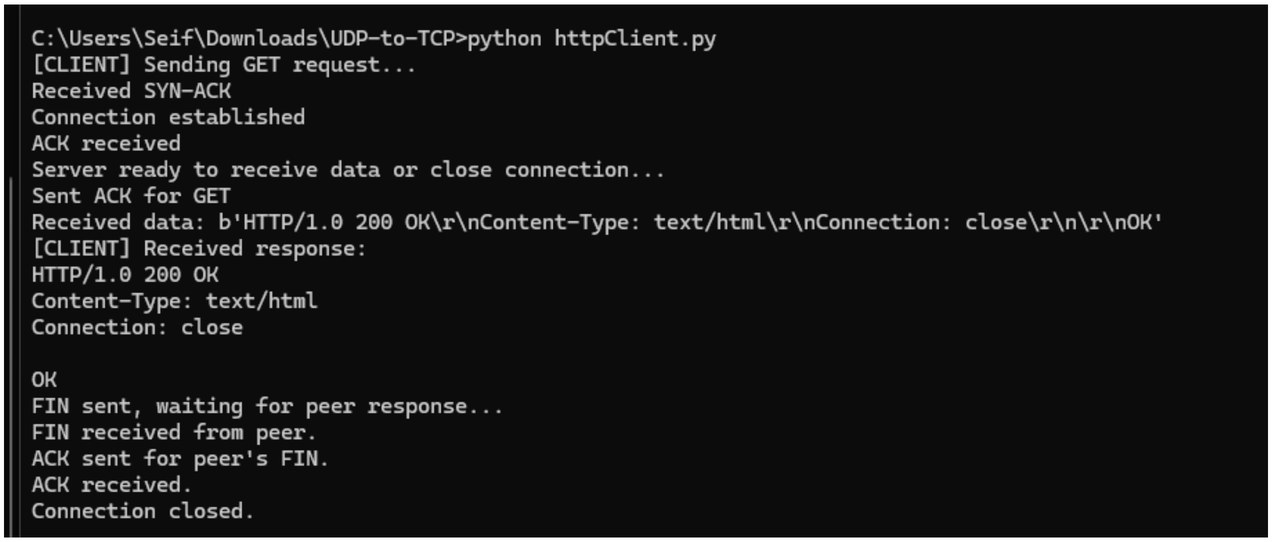
Server Side:



```
C:\Users\Seif\Downloads\UDP-to-TCP>python httpServer.py
[SERVER] Listening on 127.0.0.1:8081
[SERVER] Waiting for new connection...
Server listening for handshake...
Received SYN from ('127.0.0.1', 58447)
Sent SYN-ACK to ('127.0.0.1', 58447)
Received ACK from ('127.0.0.1', 58447)
Connection established
Server ready to receive data or close connection...
Sent ACK for GET
Received data: b'GET /index.html HTTP/1.0\r\n        Host: 127.0.0.1\r\n        Connection: close\r\n
    \r\n        '
[SERVER] Received request:
GET /index.html HTTP/1.0
        Host: 127.0.0.1
        Connection: close


Packet Lost.
Timeout or parse error, retrying… (1/5)
ACK received
FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

Client Side:



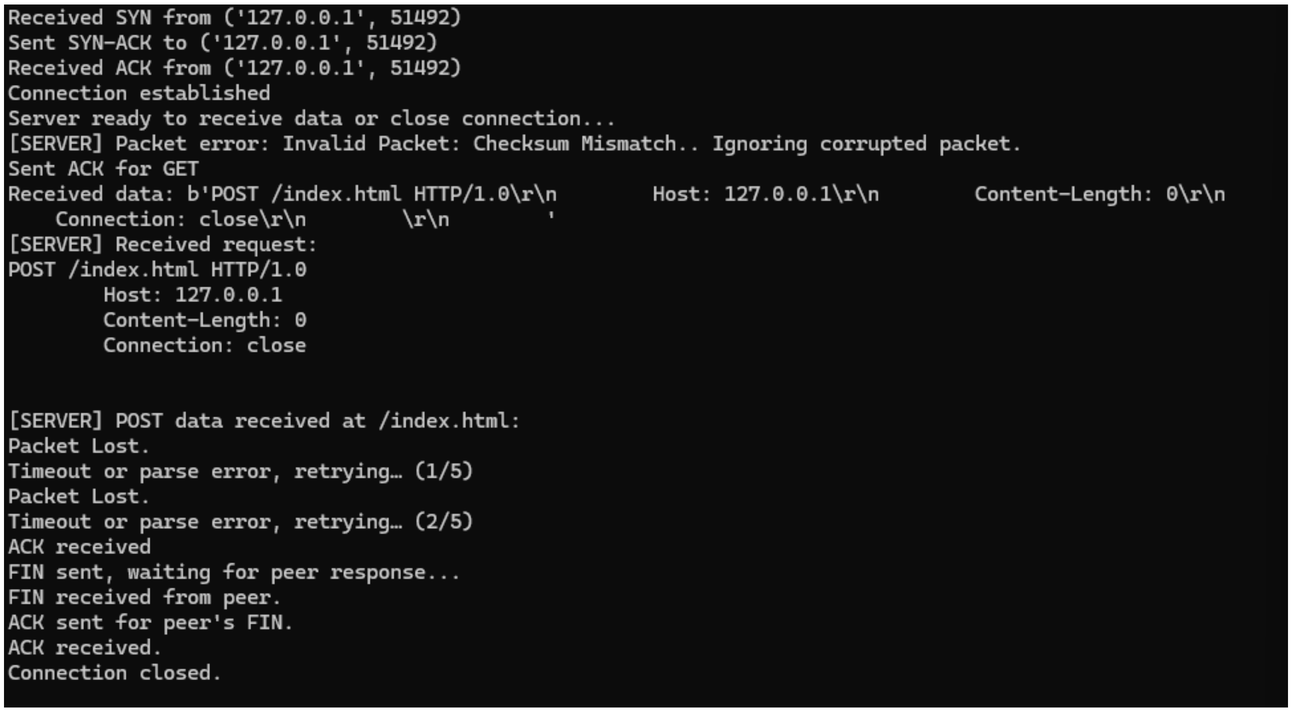
```
C:\Users\Seif\Downloads\UDP-to-TCP>python httpClient.py
[CLIENT] Sending GET request...
Received SYN-ACK
Connection established
ACK received
Server ready to receive data or close connection...
Sent ACK for GET
Received data: b'HTTP/1.0 200 OK\r\nContent-Type: text/html\r\nConnection: close\r\n\r\nOK'
[CLIENT] Received response:
HTTP/1.0 200 OK
Content-Type: text/html
Connection: close

OK
FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

In this run, the client sent a GET request and faced a simulated packet loss but eventually received the data after retransmission.

Sample #3:

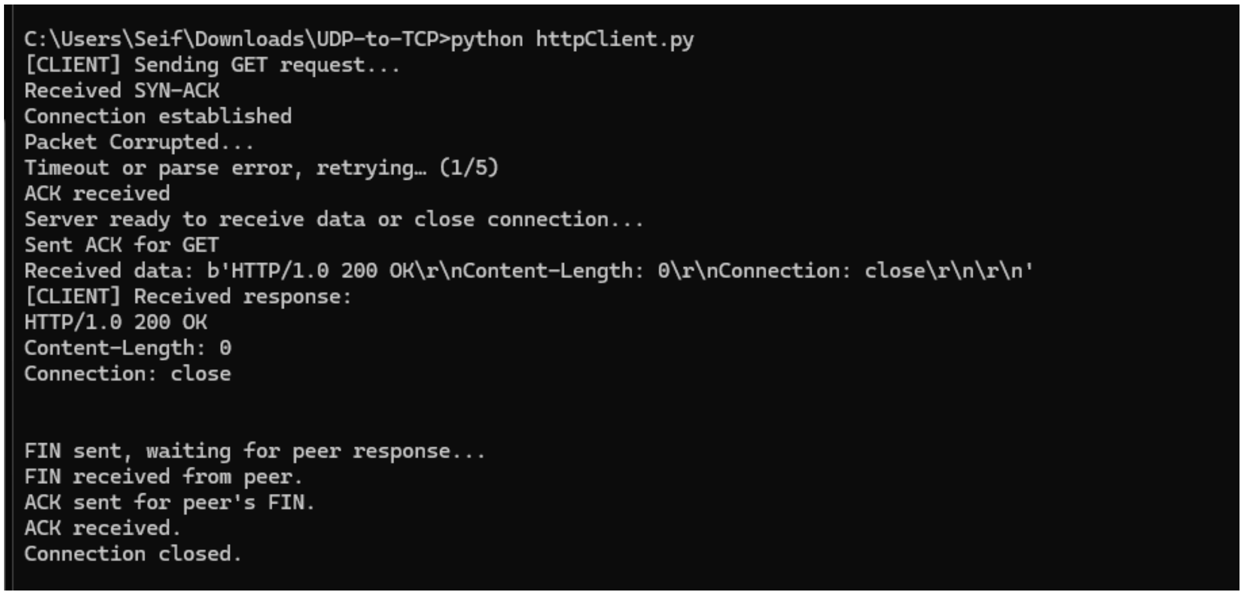
Server Side:



```
Received SYN from ('127.0.0.1', 51492)
Sent SYN-ACK to ('127.0.0.1', 51492)
Received ACK from ('127.0.0.1', 51492)
Connection established
Server ready to receive data or close connection...
[SERVER] Packet error: Invalid Packet: Checksum Mismatch.. Ignoring corrupted packet.
Sent ACK for GET
Received data: b'POST /index.html HTTP/1.0\r\n        Host: 127.0.0.1\r\n        Content-Length: 0\r\n
    Connection: close\r\n        \r\n        '
[SERVER] Received request:
POST /index.html HTTP/1.0
        Host: 127.0.0.1
        Content-Length: 0
        Connection: close


[SERVER] POST data received at /index.html:
Packet Lost.
Timeout or parse error, retrying… (1/5)
Packet Lost.
Timeout or parse error, retrying… (2/5)
ACK received
FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

Client Side:

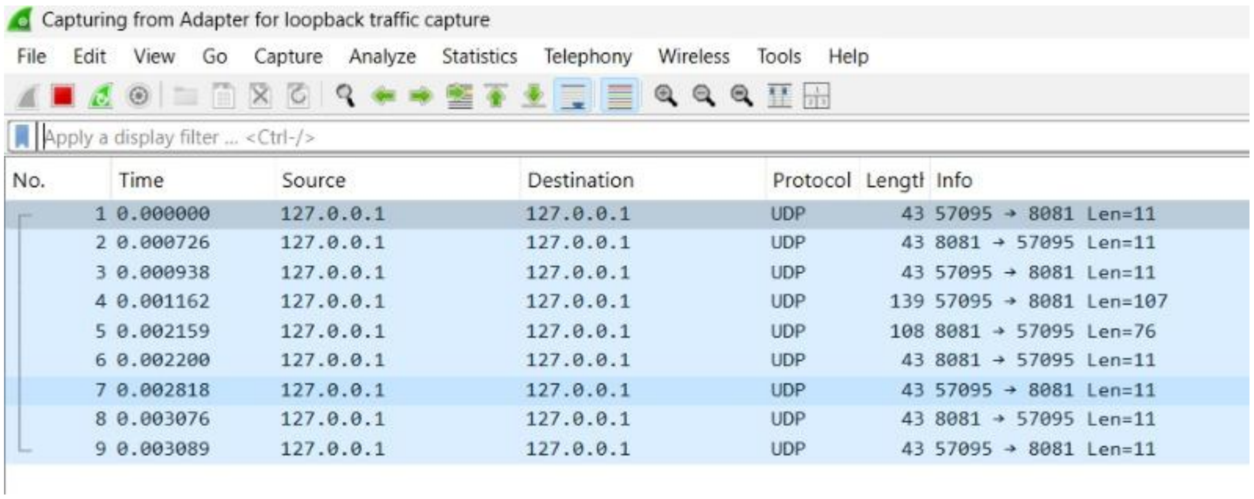


```
C:\Users\Seif\Downloads\UDP-to-TCP>python httpClient.py
[CLIENT] Sending GET request...
Received SYN-ACK
Connection established
Packet Corrupted...
Timeout or parse error, retrying… (1/5)
ACK received
Server ready to receive data or close connection...
Sent ACK for GET
Received data: b'HTTP/1.0 200 OK\r\nContent-Length: 0\r\nConnection: close\r\n\r\n'
[CLIENT] Received response:
HTTP/1.0 200 OK
Content-Length: 0
Connection: close


FIN sent, waiting for peer response...
FIN received from peer.
ACK sent for peer's FIN.
ACK received.
Connection closed.
```

In this run, the client sent a POST request and faced a simulated packet corruption and loss but eventually received the data after retransmission.

Wireshark:


Capturing from Adapter for loopback traffic capture
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	127.0.0.1	127.0.0.1	UDP	43	57095 → 8081 Len=11
2	0.000726	127.0.0.1	127.0.0.1	UDP	43	8081 → 57095 Len=11
3	0.000938	127.0.0.1	127.0.0.1	UDP	43	57095 → 8081 Len=11
4	0.001162	127.0.0.1	127.0.0.1	UDP	139	57095 → 8081 Len=107
5	0.002159	127.0.0.1	127.0.0.1	UDP	108	8081 → 57095 Len=76
6	0.002200	127.0.0.1	127.0.0.1	UDP	43	8081 → 57095 Len=11
7	0.002818	127.0.0.1	127.0.0.1	UDP	43	57095 → 8081 Len=11
8	0.003076	127.0.0.1	127.0.0.1	UDP	43	8081 → 57095 Len=11
9	0.003089	127.0.0.1	127.0.0.1	UDP	43	57095 → 8081 Len=11