TCP vs UDP : A Comparative Analysis

Project submitted for the partial fulfillment of the requirements for the course

CSE 315L: Network Security Lab

Offered by the

Department Computer Science and Engineering

School of Engineering and Sciences

Submitted by

1)Chaitanya Nath G	- AP21110011422
2) Fayaz M	- AP21110011426
3)Harsha P	- AP21110011450
4)Ajay U	- AP21110011467
5)Jeevan J	- AP21110011480

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

March,2024

Contents:

1. Introduction

(Purpose of the project)

(Provide detailed writeup with figures if possible)

1. Background

Why it is relevant (explain with example):

What are the possible ways (if any) to solve it cite them:

(Provide detailed writeup with figures if possible)

1. Proposed Approach

Explain the approach with algorithm and flowchart

Explain the system design

(Provide detailed writeup with figures if possible)

1. Results & Discussion:

Explain the result obtained

(Provide detailed writeup with figures if possible)

1. Conclusion

Conclude your work write about what you achieved, the limitations of your approach and what is your future work

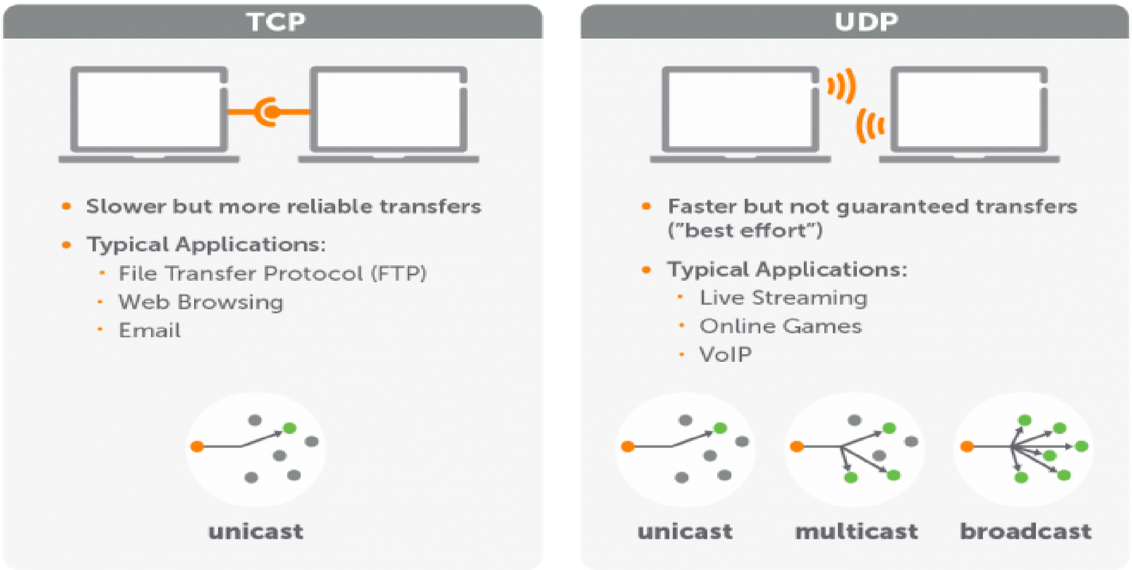
1. References

Introduction

Purpose of the project :

The purpose of this project is to develop a file transfer application that leverages the advantages of both TCP and UDP protocols while minimizing their disadvantages. By combining TCP for reliable data transfer and UDP for low-latency communication, the application aims to achieve optimal performance and efficiency. Additionally, the project includes a graphical user interface (GUI) to facilitate interaction between clients and the server.

- The problem addressed in this research is efficient and reliable file transfer between clients and servers over a network.
- File transfer is a common task in networking applications, and choosing the appropriate protocol (TCP or UDP) is crucial depending on factors such as reliability, speed, and overhead.
- TCP provides reliable, connection-oriented communication but may suffer from latency issues, while UDP offers low overhead and high-speed communication but lacks reliability.
- This application utilizes TCP for establishing a reliable connection and transferring control information (file size, metadata). UDP is used for the actual data transfer due to its faster, datagram-based approach.
- This hybrid architecture minimizes overhead while ensuring data integrity.
- A user-friendly GUI facilitates interaction for clients and the server, allowing easy selection of files for transfer and monitoring progress.



2.Background

File transfer over networks is a common task in various applications, including cloud storage, peer-to-peer sharing, and remote access. Traditionally, TCP is used for reliable file transfer, ensuring data integrity and order. However, TCP may suffer from high latency and overhead, especially for large files or over congested networks. On the other hand, UDP offers low overhead and high-speed communication but lacks reliability and error handling mechanisms.

To address these challenges, our project proposes a hybrid approach that utilizes both TCP and UDP protocols. By combining the reliability of TCP with the speed of UDP, we aim to optimize file transfer performance while maintaining data integrity. This approach allows us to overcome the limitations of individual protocols and achieve a balance between reliability and efficiency.

Relevance:

Fast and reliable file transfer is crucial in various scenarios, including:

- Sharing large files (e.g., videos, documents)
- Backing up data
- Collaborating on projects

Limitations of Single Protocols:

- **TCP:** Offers reliable, in-order delivery but can be slower due to overhead.
- **UDP:** Faster but lacks error correction and guaranteed delivery, leading to potential data loss.

Possible Solutions:

- **Proprietary Protocols:** Developing custom protocols can be complex and resource-intensive.
- **Third-Party Applications:** Existing solutions might not offer the desired level of control or flexibility.

Our Approach:

This project proposes a hybrid approach leveraging the strengths of both TCP and UDP within a single application.

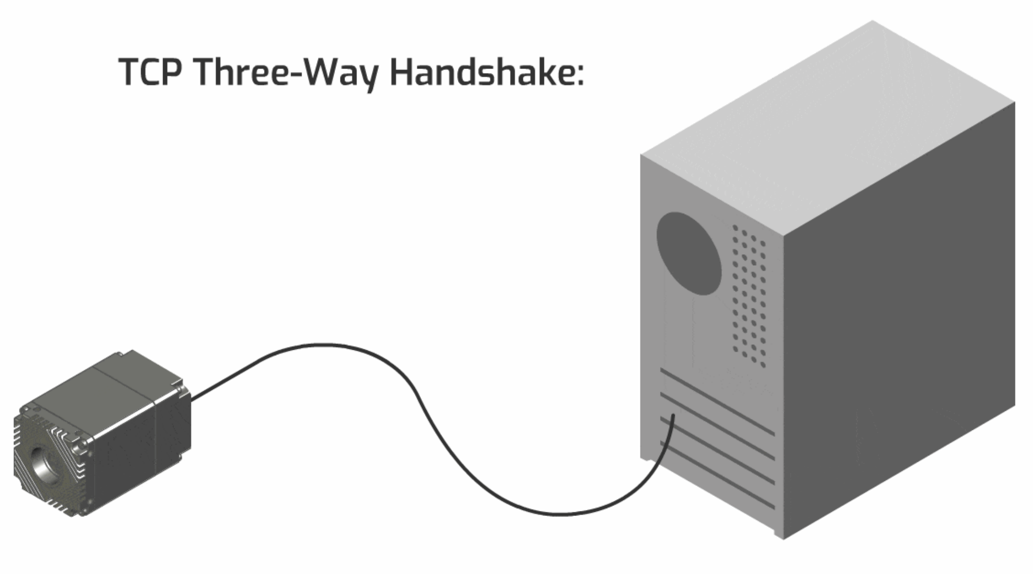
3.Proposed Approach

Working of TCP :

TCP (Transmission Control Protocol) is a connection-oriented protocol used for reliable data transmission over a network. In the context of file transfer, TCP ensures that files are transferred accurately and in the correct order between a client and a server.

1.Connection Establishment:

- TCP follows a three-way handshake process to establish a connection between the client and server.
- The client initiates the connection by sending a SYN (synchronize) packet to the server.
- The server responds with a SYN-ACK (synchronize-acknowledge) packet to acknowledge the client's request and indicate readiness to establish the connection.
- Finally, the client sends an ACK (acknowledge) packet to confirm the connection establishment.



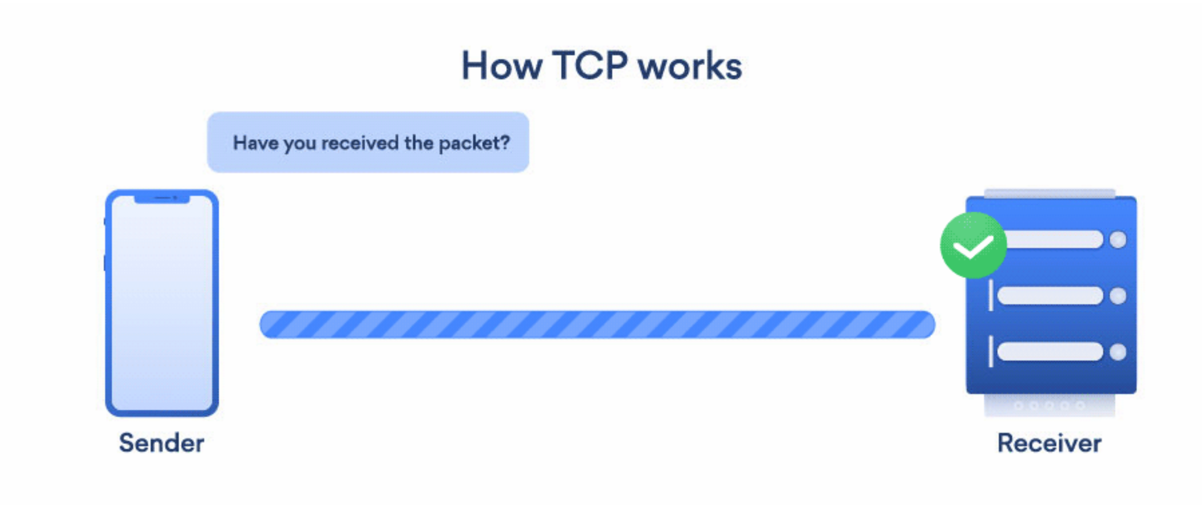
2.Data Transfer:

- Once the connection is established, data transfer begins.
- Files are divided into smaller segments or packets for transmission.
- TCP ensures that each packet is delivered reliably and in the correct order by employing sequence numbers and acknowledgment mechanisms.

- The sender sends data packets to the receiver, and the receiver acknowledges each received packet.
- If a packet is lost or corrupted during transmission, the sender retransmits the packet until it is successfully received and acknowledged by the receiver.
- This process continues until all data packets are successfully transmitted and acknowledged.

3. Flow Control and Congestion Avoidance:

- TCP includes mechanisms for flow control and congestion avoidance to optimize data transmission and prevent network congestion.



- Flow control mechanisms regulate the rate of data transmission based on the receiver's buffer capacity, ensuring that data is not overwhelmed.

- Congestion avoidance mechanisms monitor network congestion and adjust the transmission rate accordingly to prevent packet loss and maintain network stability.

4.Connection Termination:

- Once the file transfer is complete, the TCP connection is terminated to release network resources.
- TCP follows a four-way handshake process to gracefully terminate the connection.
- The client sends a FIN (finish) packet to indicate the desire to close the connection.
- The server responds with an ACK packet to acknowledge the client's request.

- The server then sends its own FIN packet to signal the termination of its end of the connection.
- Finally, the client acknowledges the server's FIN packet, and the connection is fully terminated.

Working of UDP :

UDP (User Datagram Protocol) is a connectionless protocol designed for fast data transmission with minimal overhead. It operates without the reliability and flow control mechanisms of TCP, making it faster but less reliable. UDP packets are sent without establishing a connection and may be lost or arrive out of order.

1.Connectionless Communication:

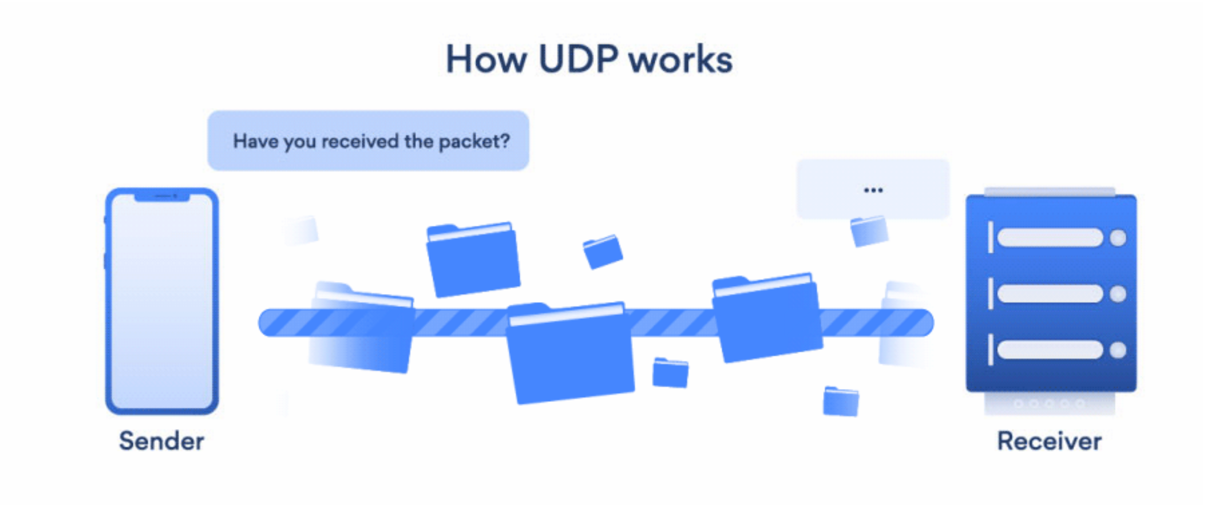
- UDP operates in a connectionless manner, meaning there is no need to establish a connection before sending data.
- Applications using UDP simply send data packets (datagrams) to the recipient without prior setup or acknowledgment.

2.Unreliable Transmission:

- UDP does not guarantee delivery of packets or ensure their order.
- Once a packet is sent, UDP does not track its status or provide mechanisms for retransmission in case of loss or corruption.
- This lack of reliability makes UDP suitable for applications where speed and low overhead are prioritized over data integrity, such as real-time multimedia streaming and online gaming.

3.Minimal Header Overhead:

- UDP has a minimal header overhead compared to TCP, which results in faster transmission and reduced network overhead.
- The UDP header consists of only four fields: source port, destination port, length, and checksum.



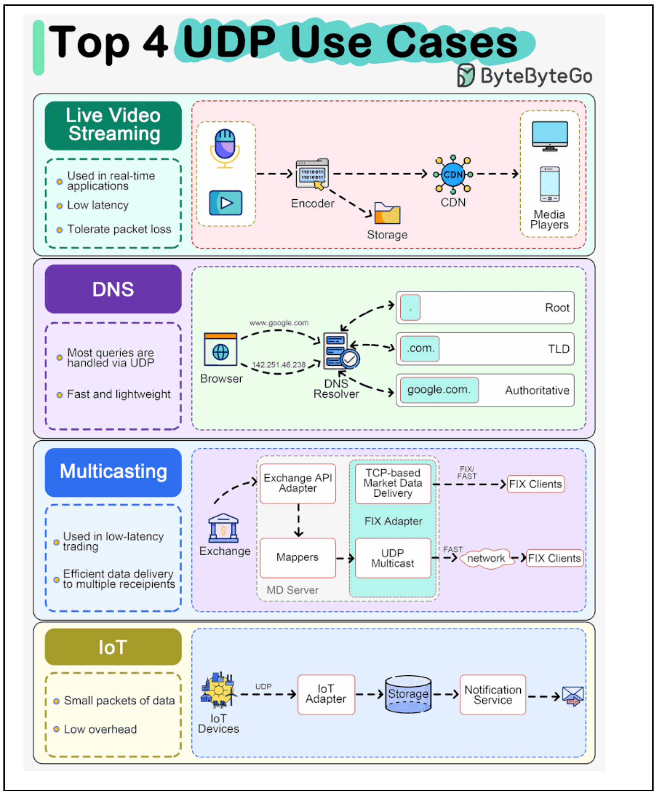
4.Fast Data Transmission:

- Due to its connectionless nature and minimal overhead, UDP offers fast and efficient data transmission.
- Applications that require real-time communication or high-speed data transfer often use UDP to minimize latency and maximize throughput.

5.No Flow Control or Congestion Avoidance:

- UDP does not include flow control or congestion avoidance mechanisms.
- This means that UDP sends data at the maximum rate possible, without adjusting the transmission rate based on network conditions.
- As a result, UDP packets may be dropped or lost in congested networks, leading to potential data loss.

6.Usage Scenarios:



- UDP is commonly used in applications where real-time communication and low latency are critical, such as VoIP (Voice over Internet Protocol), video conferencing, online gaming, and live streaming.
- It is also used for broadcasting and multicast applications, where data needs to be sent to multiple recipients simultaneously.

Differences between UDP and TCP :

The main difference between TCP and UDP is that UDP is faster than TCP. UDP has a speed advantage because the user doesn’t have to allow or acknowledge receipt of the data to be resent. This lets UDP establish connections faster and transfer data faster. However, this also causes some concerns over how safe UDP actually is.

Algorithm:

1) Initialize the server and listen for incoming connections from clients.

2)Upon connection, negotiate the protocol to be used for file transfer (TCP or UDP).

3)For TCP:

- Establish a TCP connection with the client.
- Send and receive file data in chunks, ensuring reliability and order.
- Close the connection after file transfer is complete.

4)For UDP:

Use UDP sockets for communication with the client.

Implement error detection and recovery mechanisms at the application level.

Send and receive file data in datagrams, optimizing for speed and efficiency.

5)Implement a GUI interface for users to interact with the application, including options for selecting files, initiating transfers, and monitoring progress.

System Design:

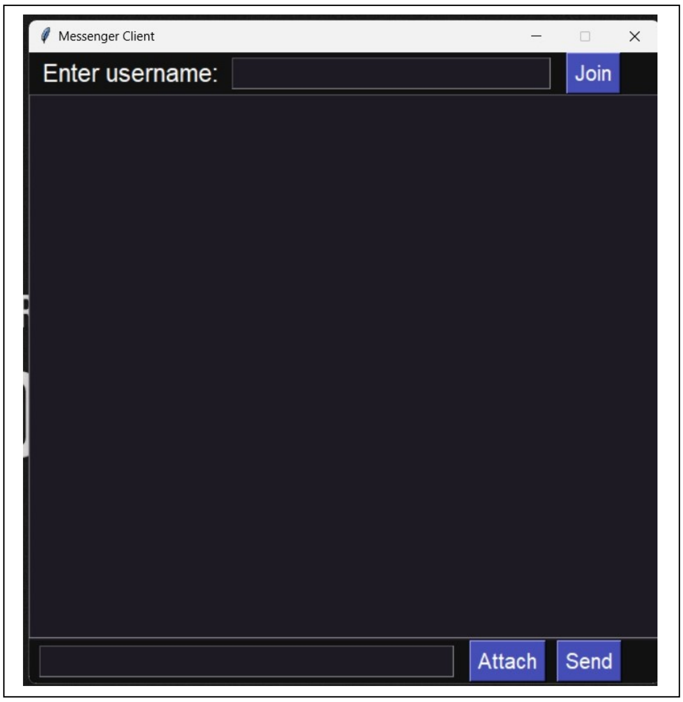
- **Client:** The client application features a user interface for file selection and transfer initiation. It communicates with the server using separate TCP and UDP sockets for control and data transfer, respectively.

- **Server:** The server application listens for incoming TCP connections and establishes communication channels. It receives file information via TCP and prepares to receive data via UDP. Upon successful transfer, the server acknowledges receipt through TCP.

- The system architecture consists of a server and multiple clients connected over a network.

- The server manages incoming connections from clients and coordinates file transfer operations.
- Clients initiate file transfers by establishing connections with the server and selecting files for transfer.

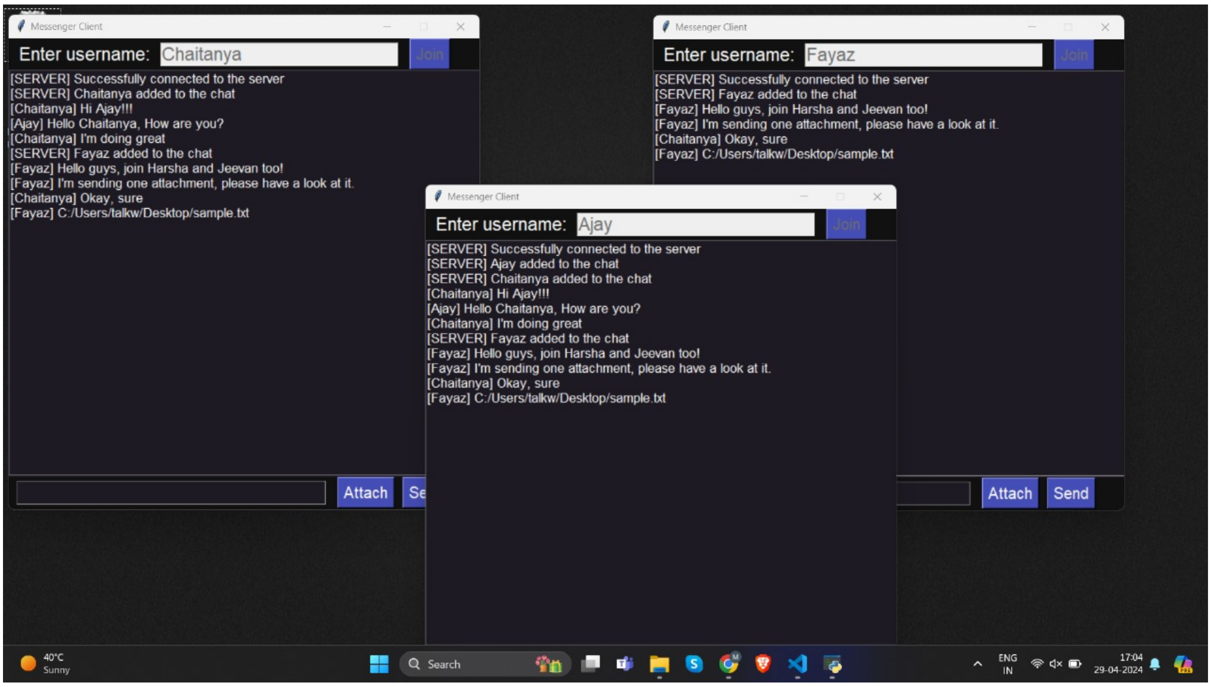
- The GUI interface provides a user-friendly way for clients to interact with the application and monitor transfer progress.
  GUI interface :



4.Results & Discussion:

- **The application successfully transfers files using both TCP and UDP protocols.**
- **We had successfully transferred the files between two clients in different end points.**
- **Results show that the hybrid approach combining TCP and UDP offers improved performance compared to using either protocol individually.**
- **The GUI interface enhances usability and provides real-time feedback to users during file transfer operations.**

- **We had created a protocols combining TCP and UDP and applied to transfer files by mitigating the disadvantages of protocols.**



Enter username: Chaitanya

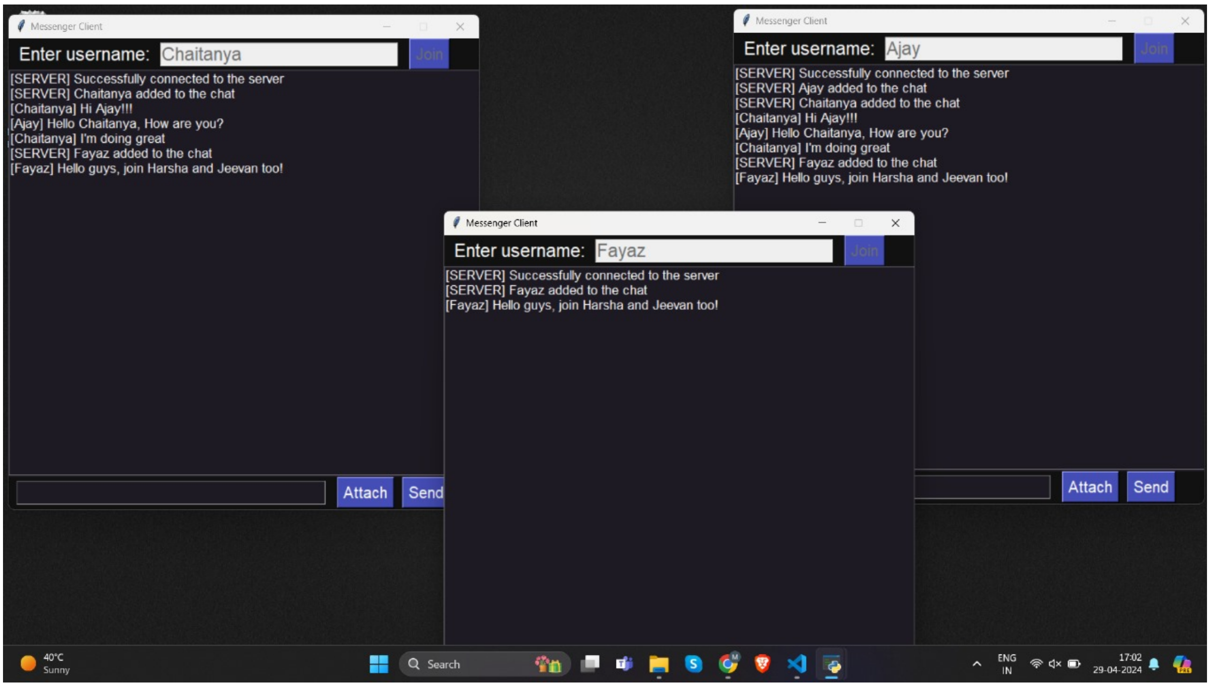
```
[SERVER] Successfully connected to the server
[SERVER] Chaitanya added to the chat
[Chaitanya] Hi Ajay!!!
[Ajay] Hello Chaitanya, How are you?
[Chaitanya] I'm doing great
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
[Fayaz] I'm sending one attachment, please have a look at it.
[Chaitanya] Okay, sure
[Fayaz] C:/Users/talkw/Desktop/sample.txt
```

Enter username: Fayaz

```
[SERVER] Successfully connected to the server
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
[Fayaz] I'm sending one attachment, please have a look at it.
[Chaitanya] Okay, sure
[Fayaz] C:/Users/talkw/Desktop/sample.txt
```

Enter username: Ajay

```
[SERVER] Successfully connected to the server
[SERVER] Ajay added to the chat
[SERVER] Chaitanya added to the chat
[Chaitanya] Hi Ajay!!!
[Ajay] Hello Chaitanya, How are you?
[Chaitanya] I'm doing great
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
[Fayaz] I'm sending one attachment, please have a look at it.
[Chaitanya] Okay, sure
[Fayaz] C:/Users/talkw/Desktop/sample.txt
```



Enter username: Chaitanya

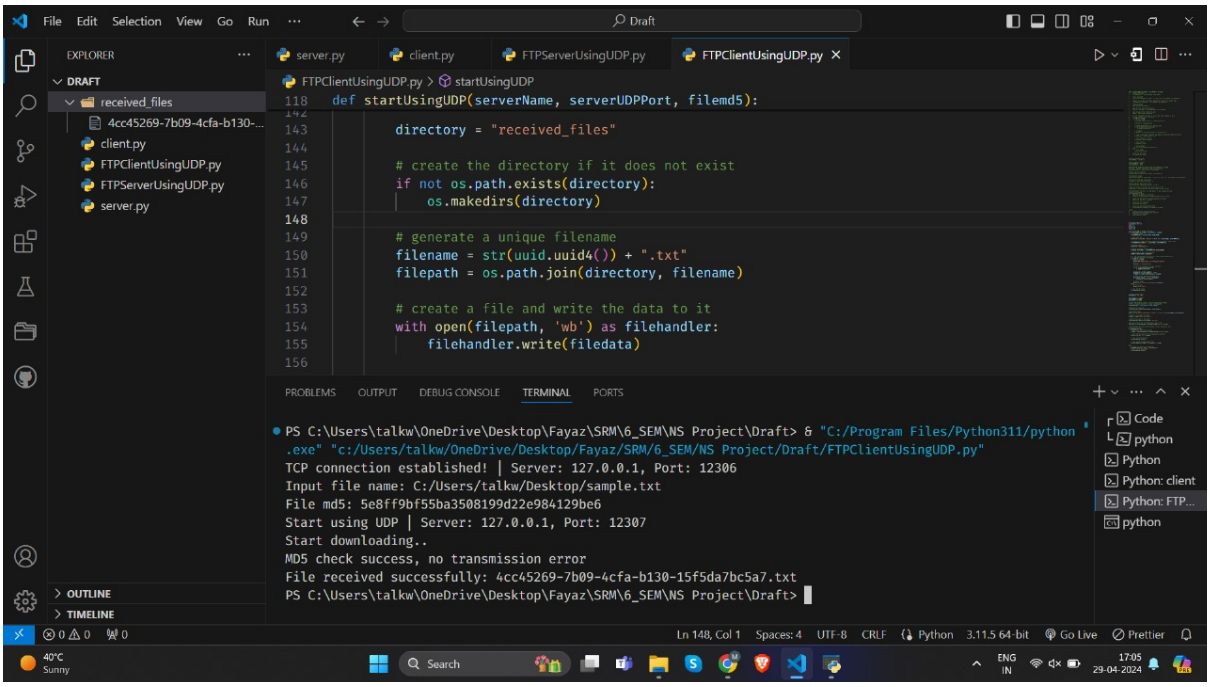
```
[SERVER] Successfully connected to the server
[SERVER] Chaitanya added to the chat
[Chaitanya] Hi Ajay!!!
[Ajay] Hello Chaitanya, How are you?
[Chaitanya] I'm doing great
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
```

Enter username: Ajay

```
[SERVER] Successfully connected to the server
[SERVER] Ajay added to the chat
[SERVER] Chaitanya added to the chat
[Chaitanya] Hi Ajay!!!
[Ajay] Hello Chaitanya, How are you?
[Chaitanya] I'm doing great
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
```

Enter username: Fayaz

```
[SERVER] Successfully connected to the server
[SERVER] Fayaz added to the chat
[Fayaz] Hello guys, join Harsha and Jeevan too!
```



```python
def startUsingUDP(serverName, serverUDPPort, filemd5):
        directory = "received_files"

        # create the directory if it does not exist
        if not os.path.exists(directory):
            os.makedirs(directory)

        # generate a unique filename
        filename = str(uuid.uuid4()) + ".txt"
        filepath = os.path.join(directory, filename)

        # create a file and write the data to it
        with open(filepath, 'wb') as filehandler:
            filehandler.write(filedata)
```

```
PS C:\Users\talkw\OneDrive\Desktop\Fayaz\SRM\6_SEM\NS Project\Draft> & "C:/Program Files/Python311/python
.exe" "c:/Users/talkw/OneDrive/Desktop/Fayaz/SRM/6_SEM/NS Project/Draft/FTPClientUsingUDP.py"
TCP connection established! | Server: 127.0.0.1, Port: 12306
Input file name: C:/Users/talkw/Desktop/sample.txt
File md5: 5e8ff9bf55ba3508199d22e984129be6
Start using UDP | Server: 127.0.0.1, Port: 12307
Start downloading..
MD5 check success, no transmission error
File received successfully: 4cc45269-7b09-4cfa-b130-15f5da7bc5a7.txt
PS C:\Users\talkw\OneDrive\Desktop\Fayaz\SRM\6_SEM\NS Project\Draft>
```

6.Conclusion

In conclusion, the project successfully develops a file transfer application that leverages the advantages of both TCP and UDP protocols. By combining reliability and efficiency, the application achieves optimal performance for file transfer operations over networks. However, there are limitations to the approach, such as increased complexity and potential overhead associated with managing multiple protocols. Future work includes further optimization, scalability testing, and integration with additional features such as encryption and authentication to enhance security.

7.References

1.G. Xylomenos and G. C. Polyzos, "TCP and UDP performance over a wireless LAN," IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320), New York, NY, USA, 1999, pp. 439-446 vol.2, doi: 10.1109/INFCOM.1999.751376. keywords: {Wireless LAN;Testing;Throughput;Wireless communication;Hardware;Protocols;Internet;Traffic control;Laboratories;Computer science},

2. D. Madhuri and P. C. Reddy, "Performance comparison of TCP, UDP and SCTP in a wired network," 2016 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2016, pp. 1-6, doi: 10.1109/CESYS.2016.7889934. keywords: {Delays;Throughput;Jitter;Packet loss;Transport protocols;Data models;TCP;UDP;SCTP;Performance;Comparison},

3. https://images.app.goo.gl/kFLfYTeH7nb2e6B29

4. https://www.linkedin.com/posts/alexxubyte_systemdesign-coding-interviewtips-activity-7150169092048486401-WUXA/