Alex Hakimzadeh
4310 Programming Assignment
16 June 2024

File Transfer using Socket Programming Report

Introduction

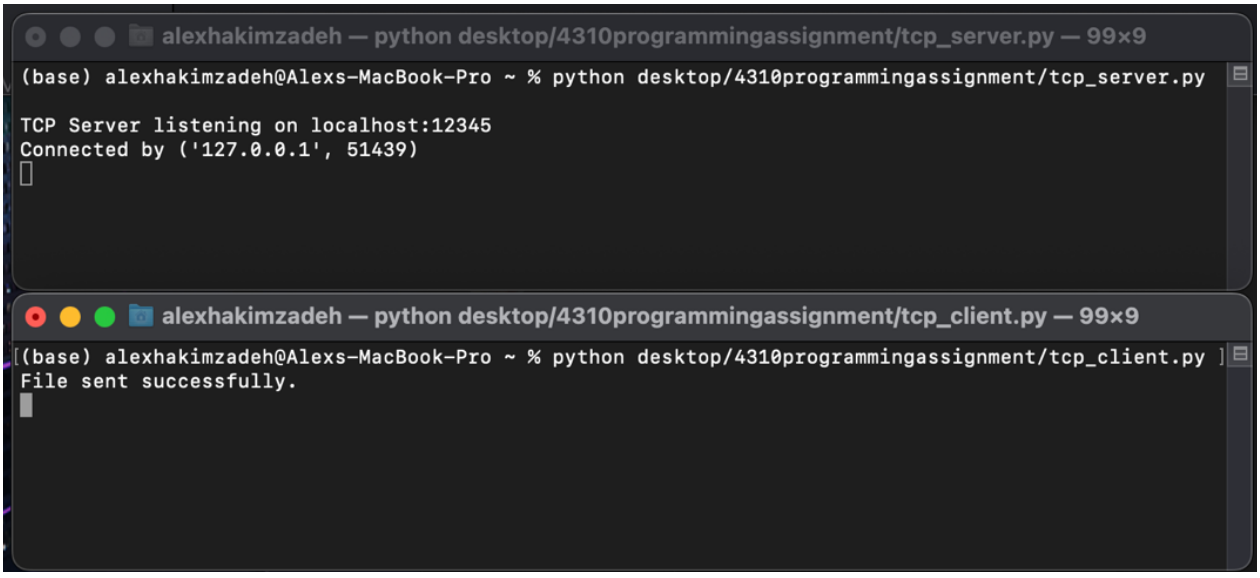
The objective of this assignment is to implement a text file transfer between a client and a server using both TCP and UDP sockets. This involves creating programs for both the client and the server to handle the file transfer and ensure that the client receives a confirmation message from the server after a successful transfer.

TCP Implementation:

TCP Server (tcp_server.py)
The TCP server listens for incoming connections on a specified port. When a client connects, the server receives a text file sent by the client, saves it to the local file system, and sends a confirmation message back to the client.

TCP Client (tcp_client.py)
The TCP client connects to the server, reads a text file from the local file system, sends it to the server, and waits for the confirmation message.

Screenshot of TCP Implementation:



Alex Hakimzadeh
4310 Programming Assignment
16 June 2024

UDP Implementation:

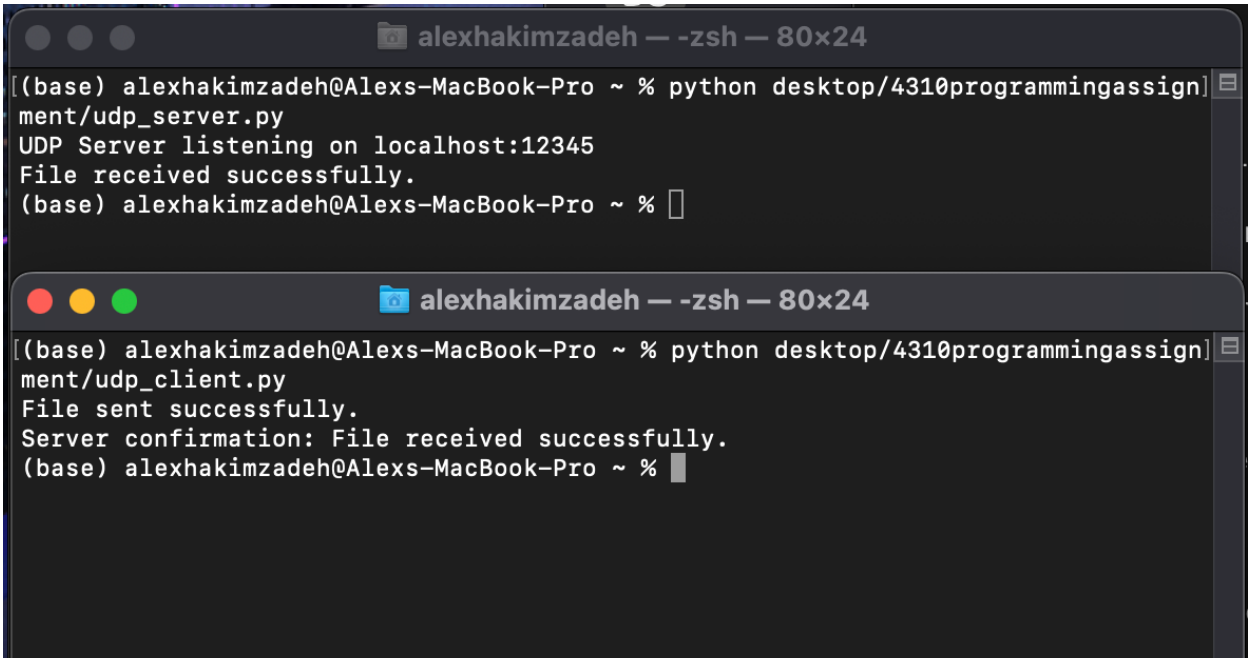
UDP Server (udp_server.py)

The UDP server listens for incoming packets on a specified port. When packets are received, it reassembles them to form the original text file, saves the file to the local file system, and sends a confirmation message back to the client.

UDP Client (udp_client.py)

The UDP client sends the contents of a text file to the server in multiple packets. After sending all the packets, it waits for a confirmation message from the server.

Screenshot of UDP Implementation:



Error Handling

TCP: The server handles exceptions related to socket operations and ensures the connection is properly closed. The client handles network errors and confirms file reception.

UDP: The server reassembles the packets and handles exceptions related to file operations and network errors. The client sends the file in chunks and ensures that all packets are sent and received correctly.

Conclusion

This assignment successfully demonstrated the implementation of file transfer using both TCP and UDP sockets in Python. The programs are designed to handle basic error conditions, ensuring robust communication between the client and server.