Implementing File Transfer using

Socket Programming and HTTP GET/POST requests

CSE 3111: Computer Networking Lab

Batch: 29 / 3rd Year 1st Semester 2024

Date

23 May, 2025

Submitted by

Aditto Raihan (Roll-09)
Jubair Ahammad Akter (Roll-59)

Department of Computer Science and Engineering
University of Dhaka

1 Introduction

In computer networking, file transfer enables data exchange between systems. Socket programming establishes direct client-server communication, providing low-level control. HTTP protocol offers standardized, reliable file transfer integrated with web technologies.

This lab explores both socket programming and HTTP GET/POST requests. Socket programming demonstrates network mechanics, while HTTP simplifies data transfer. Both methods have advantages for different scenarios.

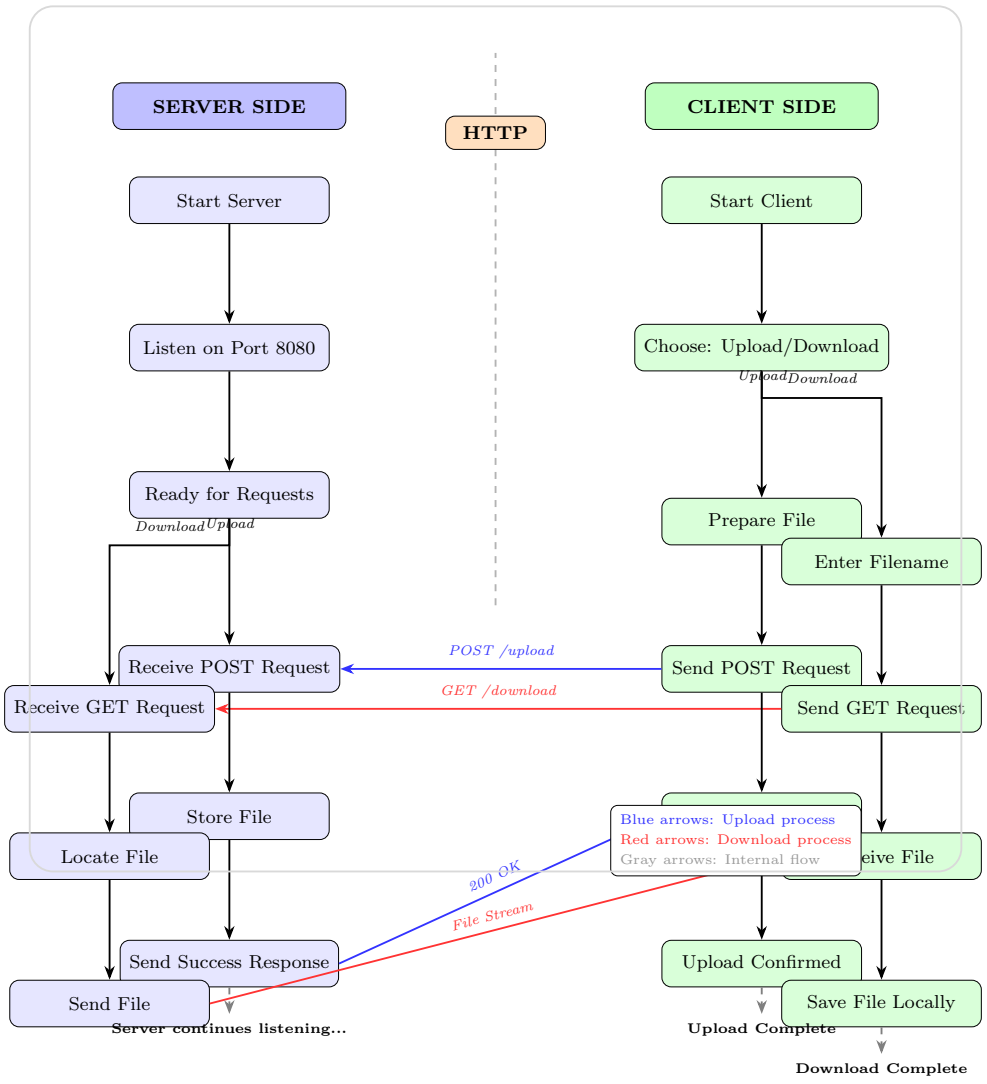
2 Objectives

The main objectives of this lab are:

- Understand network principles through socket programming
- Implement file transfer using TCP sockets
- Implement file upload with HTTP POST requests
- Implement file download with HTTP GET requests
- Compare both approaches' complexity and performance
- Analyze strengths/limitations of each method
- Gain practical network programming experience

3 Design Details

The implementation consists of a client and a server program. The server provides endpoints for file upload (HTTP POST) and file download (HTTP GET). The client interacts with the server to perform file transfers.



4 Implementation

The server was implemented using Java's built-in `HttpServer` class. The client was implemented using `HttpURLConnection`.

Server Client Screenshot

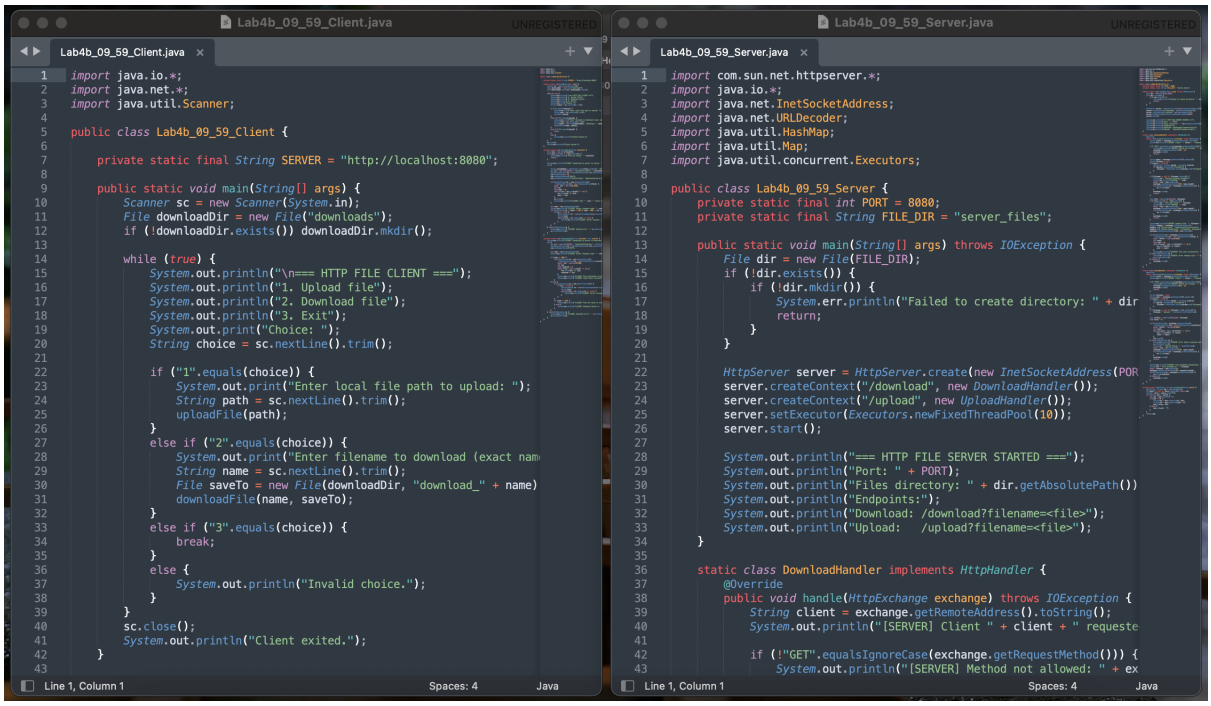


Figure 1: HTTP File Server and client running and handling requests

GitHub Repository

The full code is available at:
https://github.com/Jubair-Adib/Networking_Lab/tree/main/Lab4

5 Result Analysis

The implementation successfully allowed file upload and download using HTTP. Below are the observations:

- Upload: Client uploaded `a.txt` (9073 bytes) to the server.
- Download: Client downloaded `Routine.png` (435363 bytes) and other files successfully.
- Error Handling: The server correctly identifies files not found and sends appropriate response codes.

Terminal Outputs

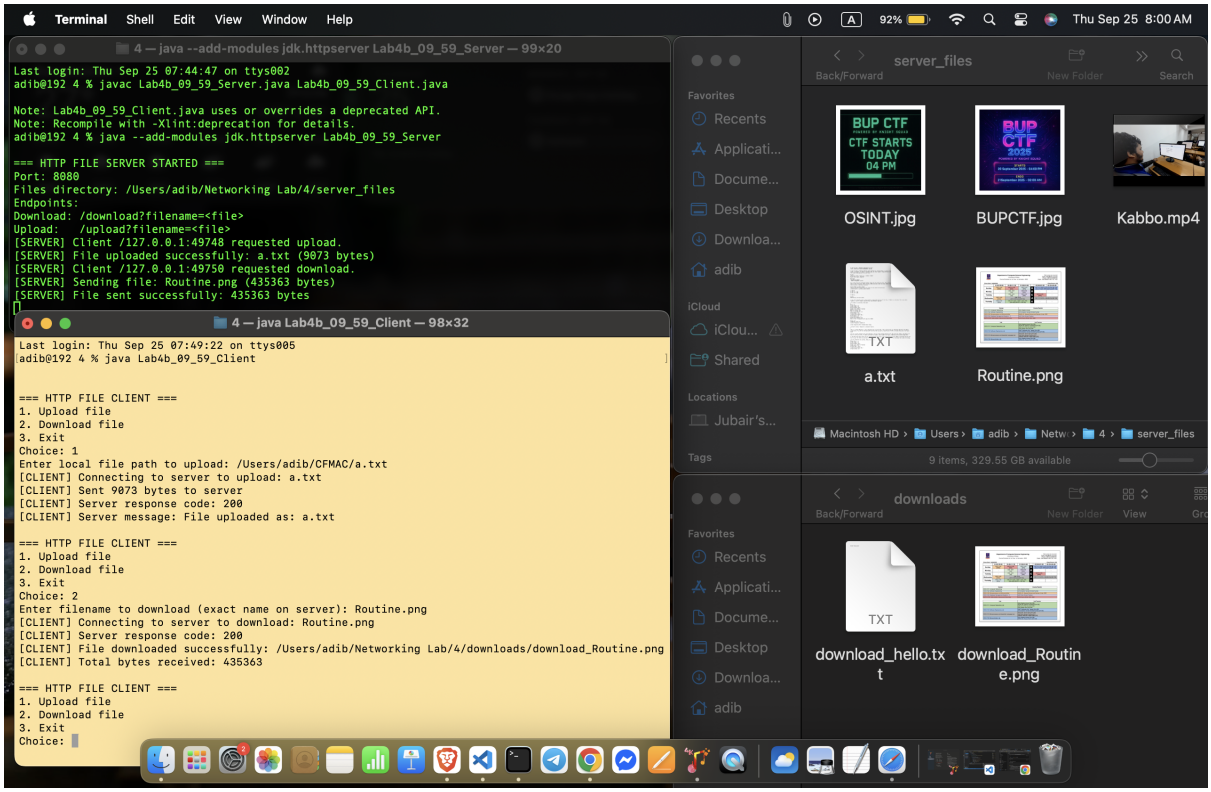


Figure 2: Successful file upload/download in terminal

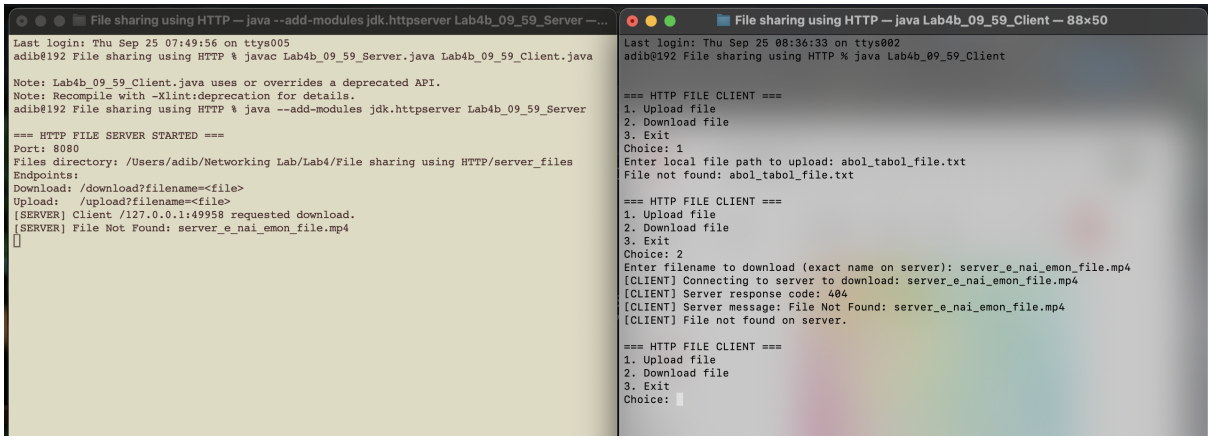


Figure 3: File not found / wrong file detection in terminal

Lab Video

Watch the demonstration video:

Lab Video on YouTube

6 Discussion

File transfer using raw sockets provides low-level control, but it requires implementing many details such as message framing, error handling, and reliability. On the other hand, HTTP provides a standardized way of transferring files through GET and POST methods.

Advantages of HTTP-based transfer

- Simpler implementation using built-in libraries.
- Compatible with web browsers and tools like `curl`.
- Easier to extend with additional features (authentication, REST API, etc.).

Limitations of Socket programming

- More error-prone and requires custom protocol handling.
- Not easily integrated with web technologies.

Learning Outcomes

During this lab, we learned how to implement file transfer using both socket programming and HTTP. We faced issues with port conflicts (Address already in use) and file path handling, which were solved by changing ports and carefully managing directories.

7 Conclusion

This lab successfully demonstrated file transfer using both socket programming and HTTP GET/POST requests. HTTP-based transfer proved to be more flexible and user-friendly compared to raw socket programming, offering better integration with web technologies and standardized error handling. However, socket programming provides finer control over the transmission process and can be more efficient for specialized applications. Both approaches have their merits and are suitable for different use cases in modern network programming.

THE END