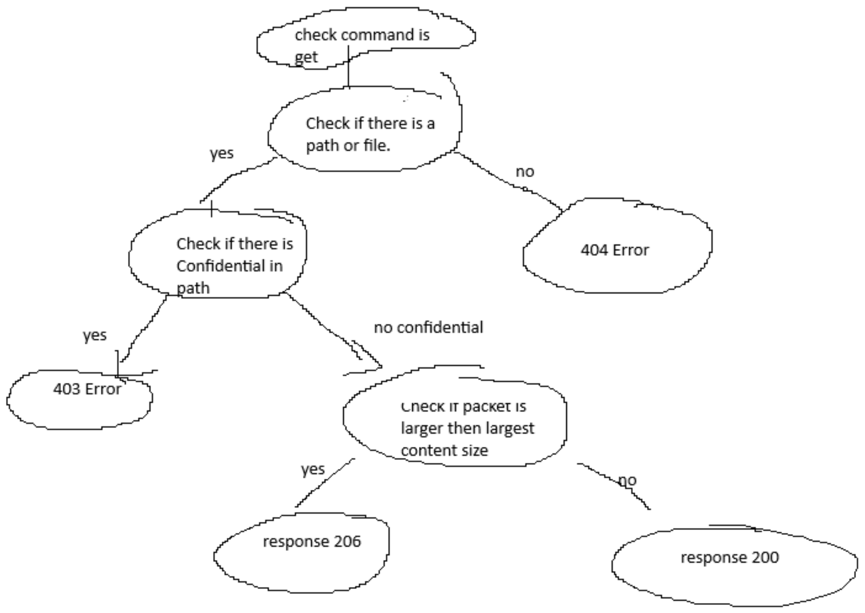
Mel Steppe, Khoi Phan
Professor Gadre
EE 419
10 June 2025

Project 4: HTTP Pseudo Streaming

This project was done jointly between the team of Mel Steppe (#2133209) and Khoi Phan (#2121579).

a. Your server design, model and components (add pictures if it helps)

Our Vod server design is a HTTP based “video on demand” server that listens on a specific port for TCP connections and provides the client with files from a local “content” directory with the HTTP GET request. We started by initializing a socket and loading in the files from the content directory using the “os” library package. The server then listens for TCP requests that are of the “GET” type. It parses the HTTP method, URL, HTTP version, and headers to determine the appropriate response based on the content’s status , for example if the file is too large or has a range request header, it will use the 206 response. Below is the general decision tree we used to create the underlying logic of our code. Here we can see how the different responses are decided on. First we check the command, check if the path exists, check if it is confidential, then check for the range header, and finally check the size of the packet.


check command is get
Check if there is a path or file.
yes
no
404 Error
Check if there is Confidential in path
yes
no confidential
403 Error
Check if packet is larger then largest content size
yes
no
response 206
response 200

Mel Steppe, Khoi Phan
Professor Gadre
EE 419
10 June 2025

The server uses “mimetypes” to identify the content type of the requested file, with a dictionary of common files types as a backup method should the mimetypes fail. Currently the server operates sequentially instead of using multithreading or asynchronous handling. The responses also follow standard HTTP protocol for their formatting.

b. How did you handle multiple clients? How many clients do you think your server can handle
Effectively?

Our system handles clients sequentially and with blocking. While this means it can only handle one “active” client at a time, the `self.http_socket.listen(10000)` means there can be up to 10,000 clients in the queue which is seen by the system as accepting 10,000 clients at a time. The later clients will have more significant time delays especially for large file sizes or slow internet connections. For small file sizes and in testing scenarios like on gradescope, we can reasonably handle the 10,000 quickly, but in a real world scenario the vod server would likely struggle with this and be more effective at around 20-30 clients.

c. Libraries used (optional; name, version, homepage-URL; if available).

We primarily used the standard socket, sys, datetime, and os libraries. In addition to these libraries we imported “mimetype” from the python standard library to handle translating the different content types such as video files.

d. Extra capabilities you implemented

We did not add any extra capabilities outside of the requirements on the project specification.