I started testing my program by trying to figure out how to establish a connection with the client curl. I would simply print out the file descriptors of each socket function to see if I was doing it right. And when I got further along, I started simple and saw if I could send messages to the client and receive curl commands from the client side. I would test with simple files further along and I introduced a larger file too. I would test filenames if they existed, were too short, illegal, etc. But it took me a while to understand how curl was sending the http headers and it took me a while to realize that doing curl commands was actually sending the GET and PUT requests. I thought GET and PUT were stuff we were supposed to give as the client, but curl does that for me. I didn't realize until I saw it in -v mode. This program was really hard to test, and implement, and I wasn't able to get a lot of the features done. But I was able to get the basic idea down.

What happens in your implementation if, during a PUT with a Content-Length, the connection was closed, ending the communication early? This extra concern was not present in your implementation of dog. Why not? Hint: this is an example of complexity being added by an extension of requirements (in this case, data transfer over a network).

In my implementation, if the connection ended early, my program would just terminate without finishing the task. This extra concern was not present in dog.cpp because it was guaranteed that it would execute. However, because there is a dependency between the client and server, there's no guarantee that one will not lose connection of the other. The data transferring from one side to the other could get caught off halfway, and depending on implementation, nothing could happen or just what was read and loaded could be sent over causing a partial transfer.