Well if a PUT request is initiated the server can only write to the target file WHILE the connection is open so if it ends early the content will not fully be written into the file. This was not the case in dog because in dog writing does not require a connection to be open only that the files are located within the same directory as the dog file so it can collect the stream of data.

Test cases:
GET request using populated file
GET request using empty file
PUT request using populated file
PUT request using empty file
Differentiated 201/200 status codes for PUT request based on created/existing file
Validate filename and writing 400 Bad Request based on that
500 Internal Server Error when client sends invalid request(not PUT/GET)
403 Forbidden if for PUT GET requests files without permission
404 if file has not been located in client directory or server directory
Checking for existence of Content-Length in PUT request
Parsing for Content-Length if it exists in PUT
Collecting and sending back in a response header Content-Length in GET request
Convert hostname to its IP address from user's input in stdin
Error response if no arguments (argc ==1) or too many (if argc > 3) are specified
Reading upto Content-Length if PUT requested specifies that else upto EOF