CS 118 Project 1

Concurrent Web Server using BSD Sockets

Connor Fong 303991911

Kai Wang

Server Design

We used the code provided by the TA as the base for our design, which created the appropriate sockets and bindings needed for connection. Then, it would wait until a connect signal is received, in this case from the web browser, and create the connection. We coded in the “dostuff” function which parses the request made by the client, and creates an HTTP response with the requested file. The file requested is read in binary mode, so that any embedded images in the requested HTML file will be output correctly. Since we chose to use the “fstream” library to read files, we had to modify the Makefile provided so that the program would compile using G++, rather than GCC

Manual

To run the server, you must type “make” in the terminal and allow the program to compile. After it has compiled, it can be run by typing “./serverFork <port number>”, which creates the server at the input port number. The port number must be defined, or the server will not run and an error will be output. Once the server is running, open a web browser and type in the URL “<machine name>:<port number>/test.html”. The web page should display a message, a .jpg, and a .gif. You can view the .jpg or .gif files independently by replacing “test.html” with “test.jpg” or “test.gif”.

Test Cases



In this case, I set up the server in the terminal and connected to it in Mozilla using the process described in the manual. This is the HTTP request found in the console after the connection had been made. Here we can see information about the browser accessing the server, the port number, and shows the files embedded in the HTML code that the server must display.



The above image shows the result of connecting to the server from the Mozilla web browser. This displayed the plain text found in the HTML file, .jpeg image, and .gif file.