Project 1
Concurrent Web Server Using BSD Sockets

CS 118
Winter 2016

Meghana Ginjpalli
804588573

High Level Design

This project uses socket programming that can read and process HTTP requests from a client. The web server then sends back a response to the client with the respective headers and data. This project was developed on a Linux VM using Ubuntu and the client in this case is the Firefox browser. Requests are made through the browser URL by typing:

http://localhost:<port number>/test.html

The server initially listens for any clients that are trying to connect. When the connection is successfully created, the socket processes the HTTP request coming from the client. This request is also displayed in the console. As each request is read, the HTTP request message is first shown in the console and attempts to get the name of the requested file and display its contents to the browser.

In this project, there is a function called void read_request(char* url, int request) that first checks to see if the name of the file is correct and exists. If it exists, it opens and reads in the file data. If it does not exist, then the client receives a message saying that the file was not found. Another function called void parse_request(int request) is responsible for reading and processing the data requested.

The web server can handle html, jpg, and gif files. As of right now, this webserver can only serve one client at a time but can accept multiple HTTP requests. To see sample output, keep reading below.

Difficulties

Getting used to C syntax and trying to determine which functions are better to use when it comes to manipulating strings and buffer took some figuring out. Also, there were also some issues trying to get the client request files to display on the browser, especially the gif and jpg files. I was able to fix this issue after increasing the buffer size. Also, when initially loading the server through the url, the HTTP request displays a message saying that a favicon.html file was not found. You can ignore this error as it is just saying it cannot find a logo image for the url.

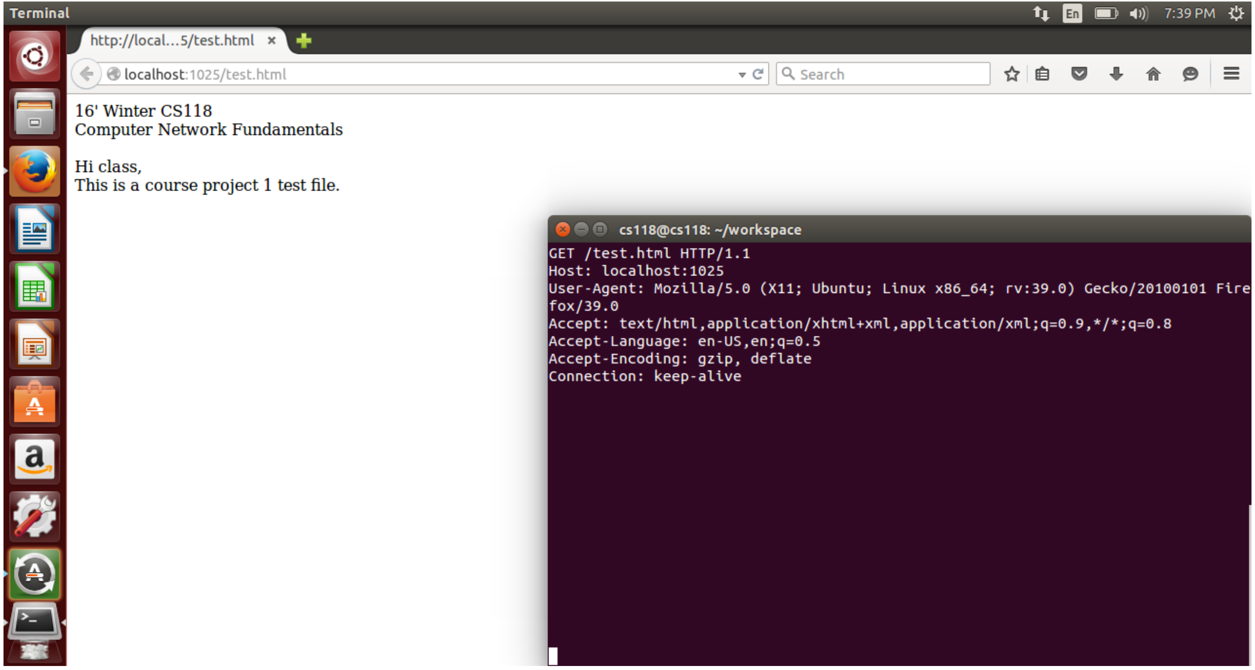
How to Compile and Run the Webserver

1. Navigate to the folder containing webserver.c and the Makefile.
2. Type “make” to compile the file, you can also type “make clean” to clean up any preexisting object and executable files
3. Type “./webserver <port number>” to start up the webserver
4. Open the browser, and enter http://localhost:<port number>/test.html. In this case, test.html is the file name, but you can also enter other jpg or gif files such as dog.gif or ucla.jpg.

Sample Outputs

After running “make” and “./webserver 1025”, these are the following outputs you should be seeing:

In the url, type http://localhost:1025/test.html. This is what you should see in the browser and console. As you can see, the console indicates that it has received the request and the browser displays the data of this request:

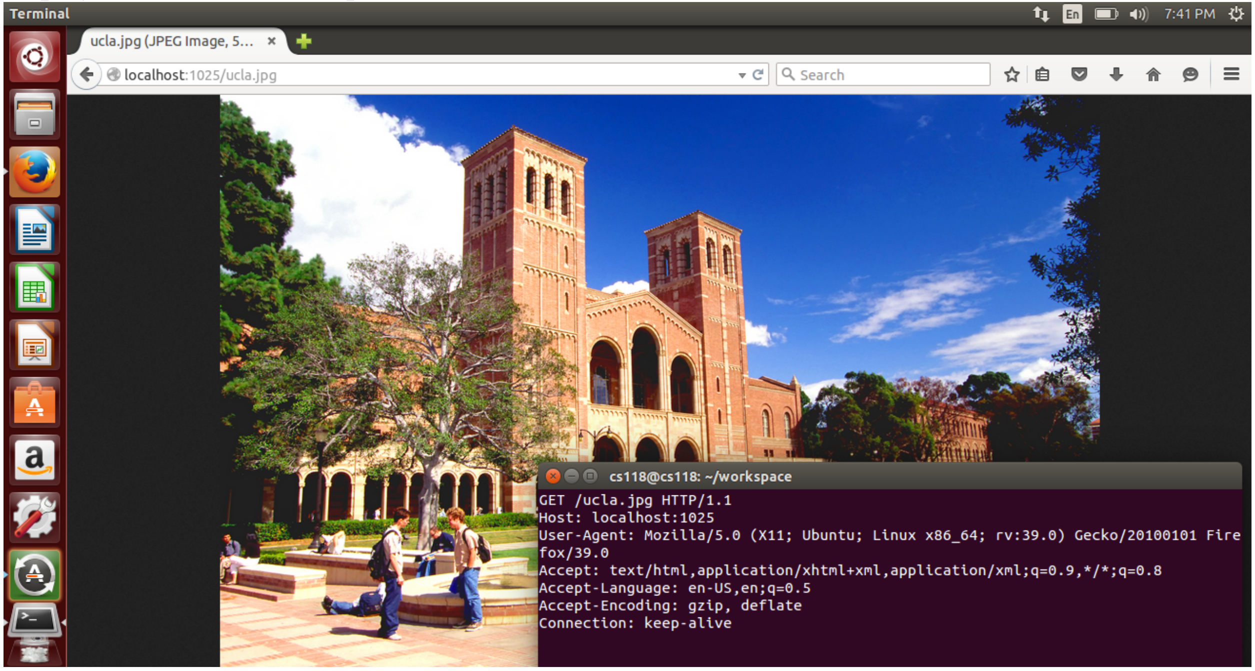


16' Winter CS118
Computer Network Fundamentals

Hi class,
This is a course project 1 test file.

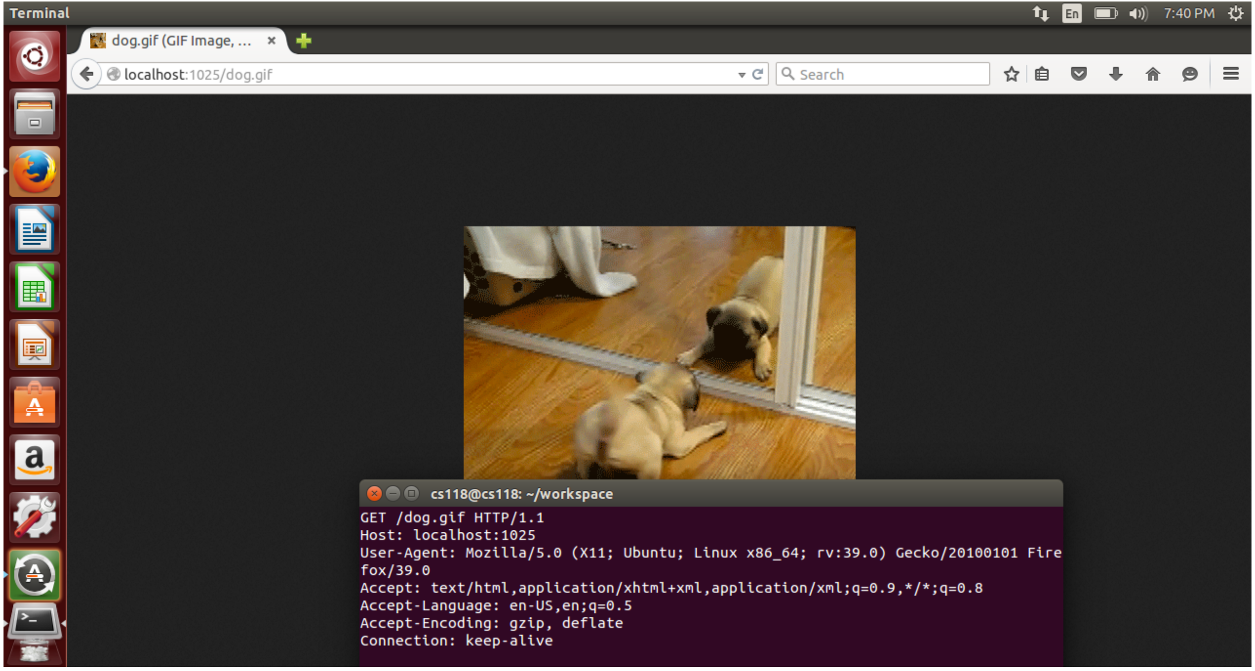
```
GET /test.html HTTP/1.1
Host: localhost:1025
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:39.0) Gecko/20100101 Fire
fox/39.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

In the url, type http://localhost:1025/ucla.jpg. This is what you should see in the browser and console. As you can see, the console indicates that it has received the request and the browser displays the data of this request:



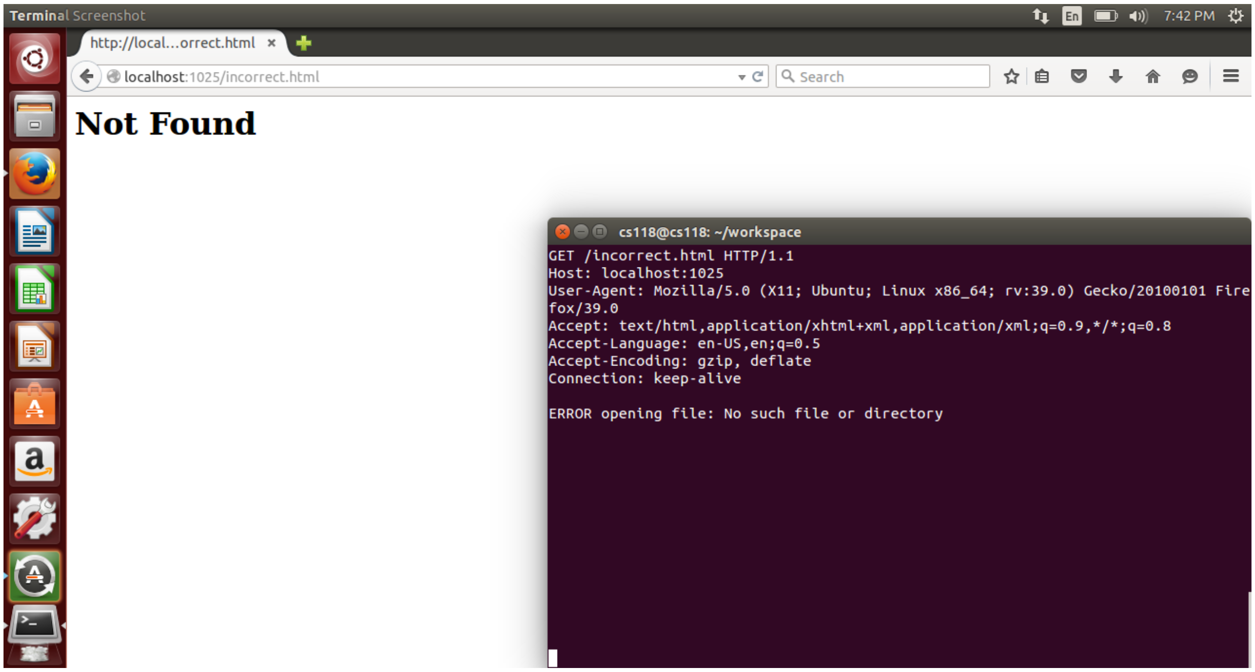
```
GET /ucla.jpg HTTP/1.1
Host: localhost:1025
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:39.0) Gecko/20100101 Fire
fox/39.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

In the url, type http://localhost:1025/dog.gif. This is what you should see in the browser and console. As you can see, the console indicates that it has received the request and the browser displays the data of this request:



```
GET /dog.gif HTTP/1.1
Host: localhost:1025
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:39.0) Gecko/20100101 Fire
fox/39.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

If we had typed an incorrect filename that did not exist, this is how it would look in the browser and console:



Not Found

```
GET /incorrect.html HTTP/1.1
Host: localhost:1025
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:39.0) Gecko/20100101 Fire
fox/39.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive

ERROR opening file: No such file or directory
```