HW2 – Simplified HTTP Retriever and Server

Important

Within this submission, I have included the following:

- some txt file and SecretFile.html to test the server and retriever
- a Retriever.cpp file and Server.cpp: the actual source code
- a BadRetriever.cpp file that is similar to Receiver.cpp but using malformed request: Post instead of GET. Its test case is inside the RetrieverTest
- a Build.sh script that will compile the three source code above: bash Build.sh
- a ServerCreate.sh script that will create the server, giving the port number: **bash ServerCreate.sh #portNumber. Run this before running the retriever**
- a RetrieverTest.sh script that will perform execution from 3-7, given the machine name and port number (need to be the same as the one with ServerCreate): **bash RetrieverTest.sh #machineName #portNumber**
- a RetrieverTestCase2.sh that will test retriever accessing a real web server, just need to provide the server name and file: **bash RetrieverTestCase2.sh #serverName #fileName**

To compile these manually, type

g++ -pthread -std=c++14 -o Server Server.cpp

g++ -std=c++14 -o Retriever Retriever.cpp

g++ -std=c++14 -o BadRetriever BadRetriever.cpp

./Server #portnumber

./Retriever #machineName(or address)   #fileName #portNumber   (Same as the server. In the case of communicating with actual web server, use port 80

Use similar format like Retriever for BadRetriever

The process is as followed:

Run the build.sh file → Run the server create → Run the RetrieverTest → Run RetrieverTestcase2

Retriever

The retriever is a simple client that will make an HTTP Get Request to a website, received the response and write its content to a text file. For setting up the socket, I just re-used to code for within HW1. And for parsing the header info and receiving the data, I followed a guide online and also through the slides An interesting thing about parsing the header and creating the request is the use of carriage return and new line character. When receiving a response, \r\n\r\n indicate the end of header info, same as making the request.

Below is a diagram that represent the flow of the program

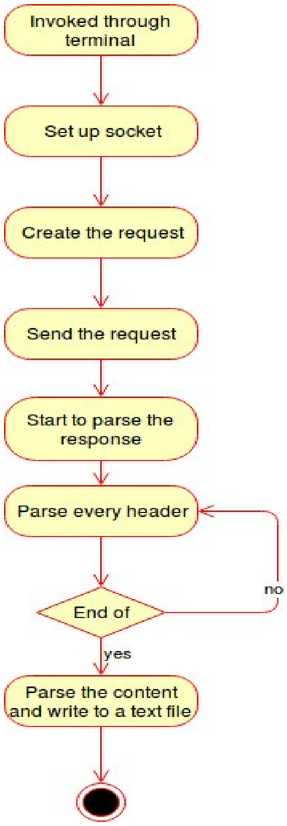
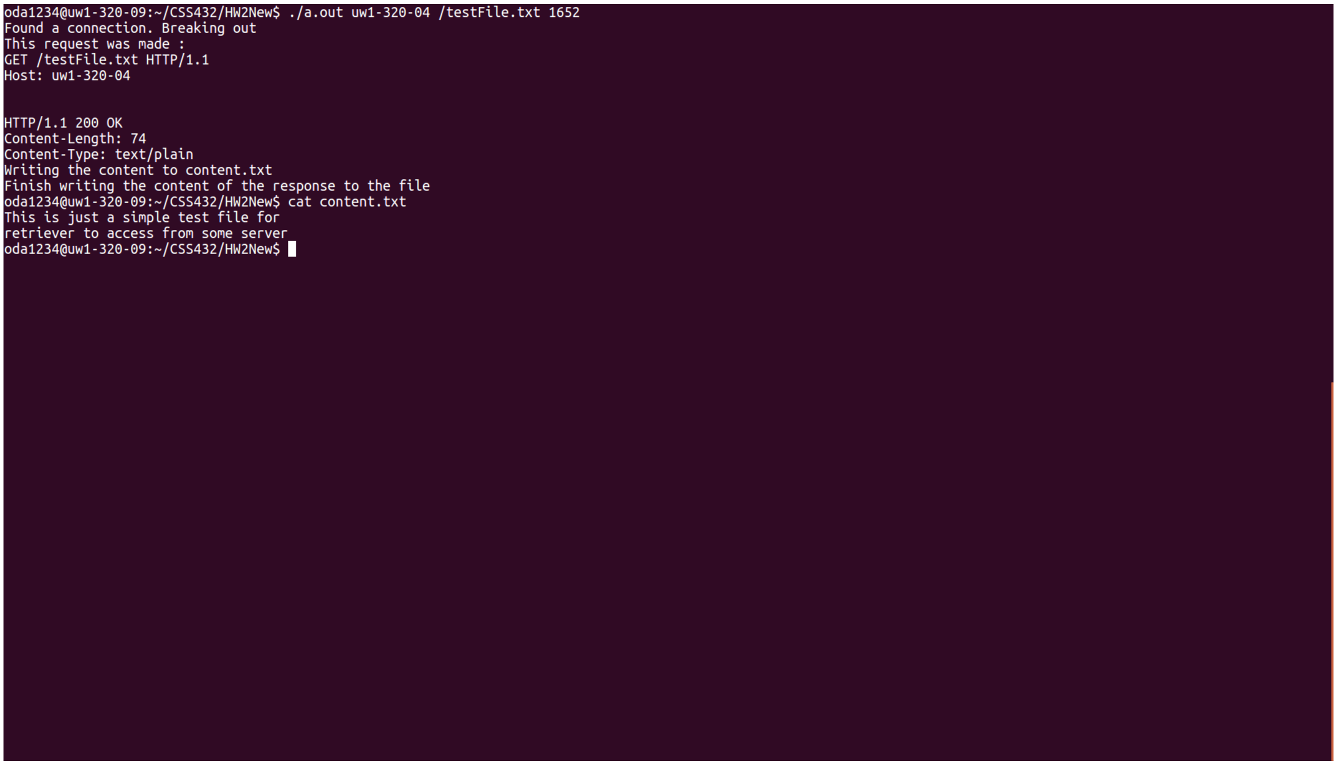


Illustration 1: Program flow for Retriever



```
oda1234@uw1-320-09:~/CSS432/HW2New$ ./a.out uw1-320-04 /testFile.txt 1652
Found a connection. Breaking out
This request was made :
GET /testFile.txt HTTP/1.1
Host: uw1-320-04


HTTP/1.1 200 OK
Content-Length: 74
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
oda1234@uw1-320-09:~/CSS432/HW2New$ cat content.txt
This is just a simple test file for
retriever to access from some server
oda1234@uw1-320-09:~/CSS432/HW2New$ 
```

Illustration 2: Retriever execution output and content of response

Server

The server also use similar code from HW1 to set up the server. It also use similar logic of parsing the header file from Retriever as well. Once it get a file name, it will read the file and put the file content inside the body of the response and send back to the client
Below is a diagram that represent the flow within the Server

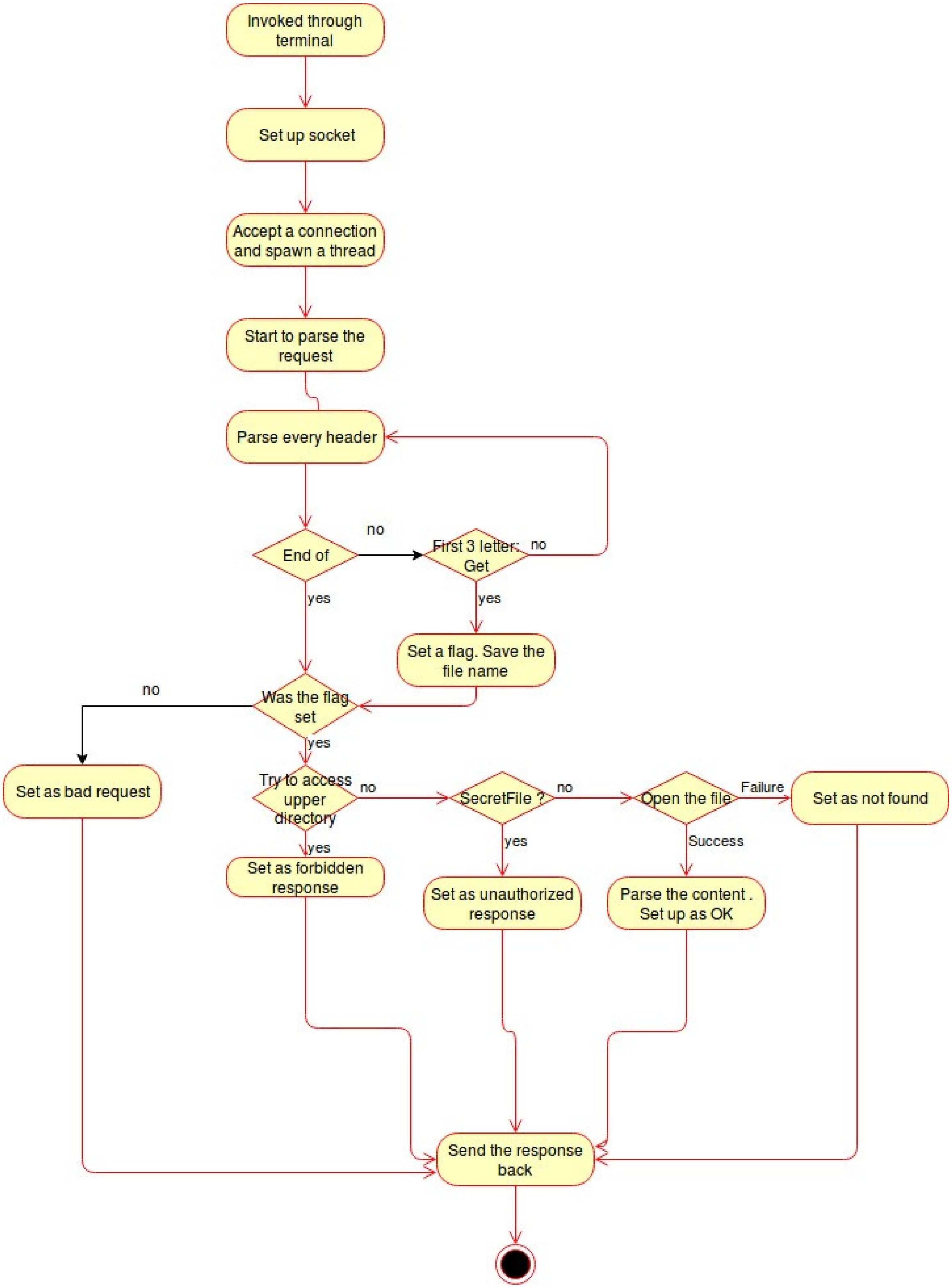
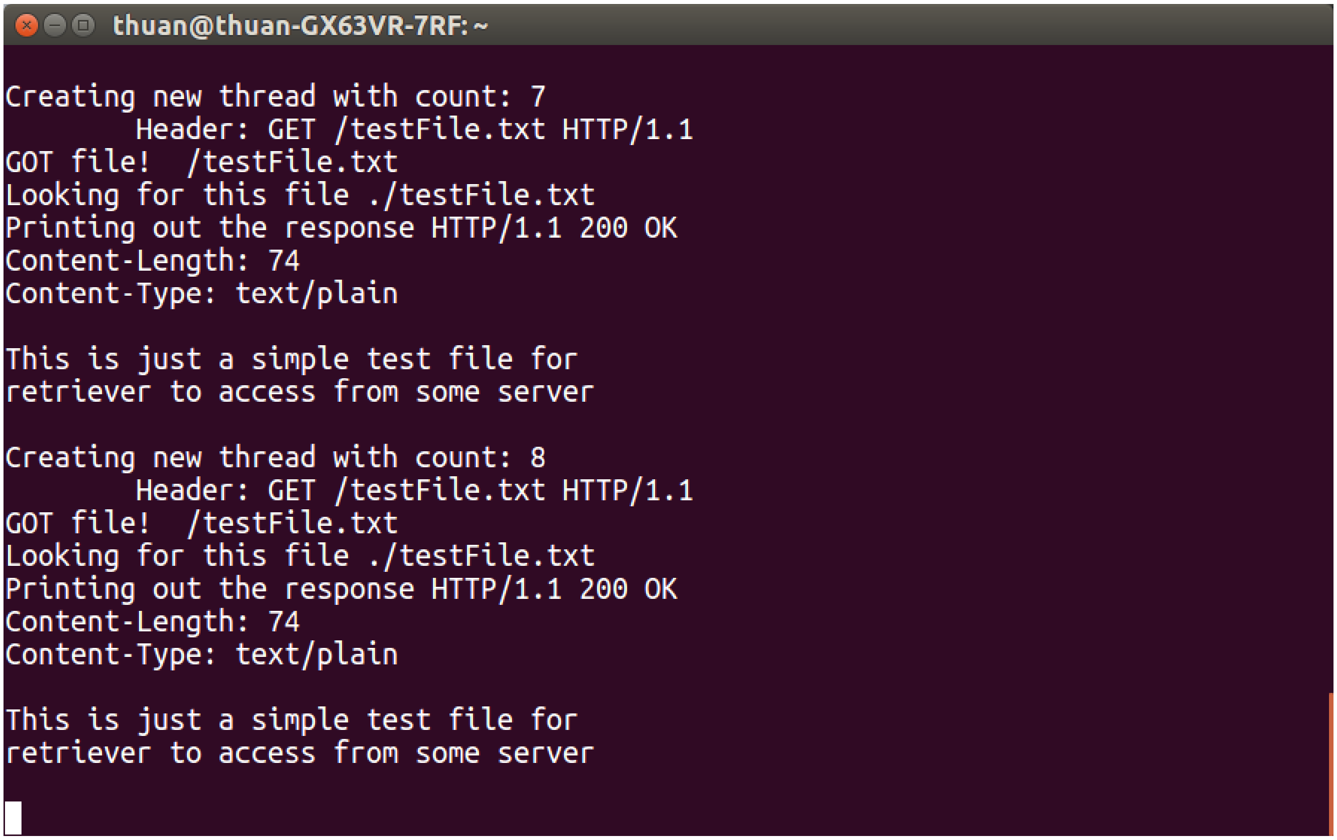


Illustration 3: Program flow for Server



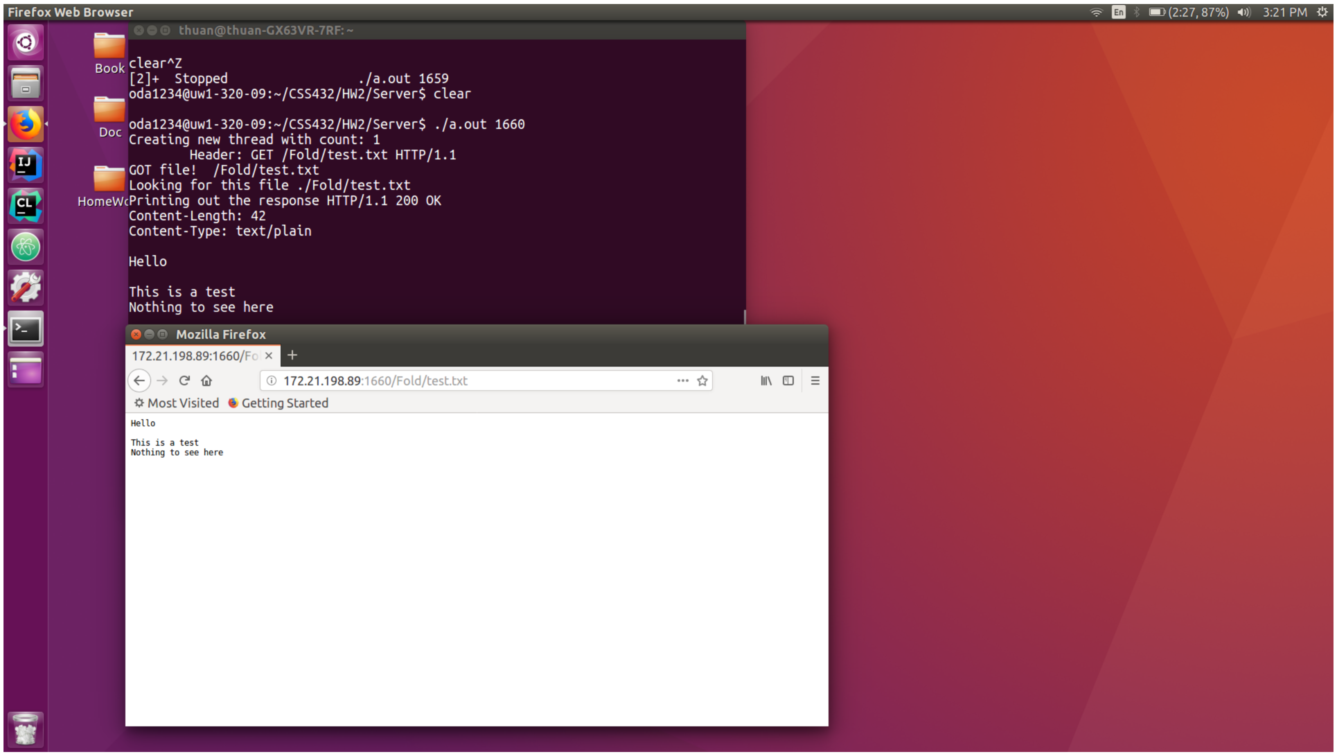
```
Creating new thread with count: 7
        Header: GET /testFile.txt HTTP/1.1
GOT file!  /testFile.txt
Looking for this file ./testFile.txt
Printing out the response HTTP/1.1 200 OK
Content-Length: 74
Content-Type: text/plain

This is just a simple test file for
retriever to access from some server

Creating new thread with count: 8
        Header: GET /testFile.txt HTTP/1.1
GOT file!  /testFile.txt
Looking for this file ./testFile.txt
Printing out the response HTTP/1.1 200 OK
Content-Length: 74
Content-Type: text/plain

This is just a simple test file for
retriever to access from some server
```

Illustration 4: Server execution output for retriever request above

Execution output

1) Real Web browser accessing my server

Using ifconfig to look up the ip address and append the port number



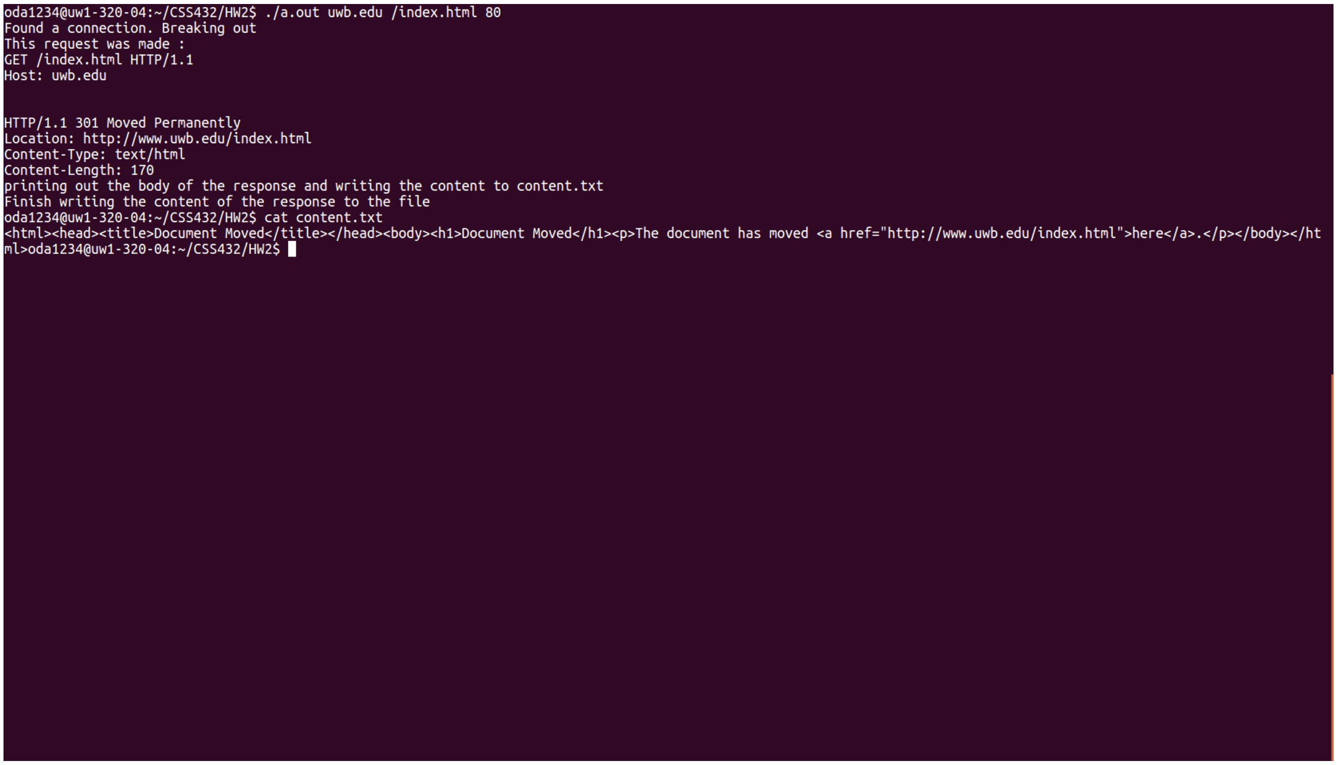
```
clear^Z
[2]+  Stopped                 ./a.out 1659
oda1234@uw1-320-09:~/CSS432/HW2/Server$ clear

oda1234@uw1-320-09:~/CSS432/HW2/Server$ ./a.out 1660
Creating new thread with count: 1
        Header: GET /Fold/test.txt HTTP/1.1
GOT file!  /Fold/test.txt
Looking for this file ./Fold/test.txt
Printing out the response HTTP/1.1 200 OK
Content-Length: 42
Content-Type: text/plain

Hello

This is a test
Nothing to see here
```

```
Hello

This is a test
Nothing to see here
```

Illustration 5: Using a real web browser to access server

2) My retriever accessing a real server

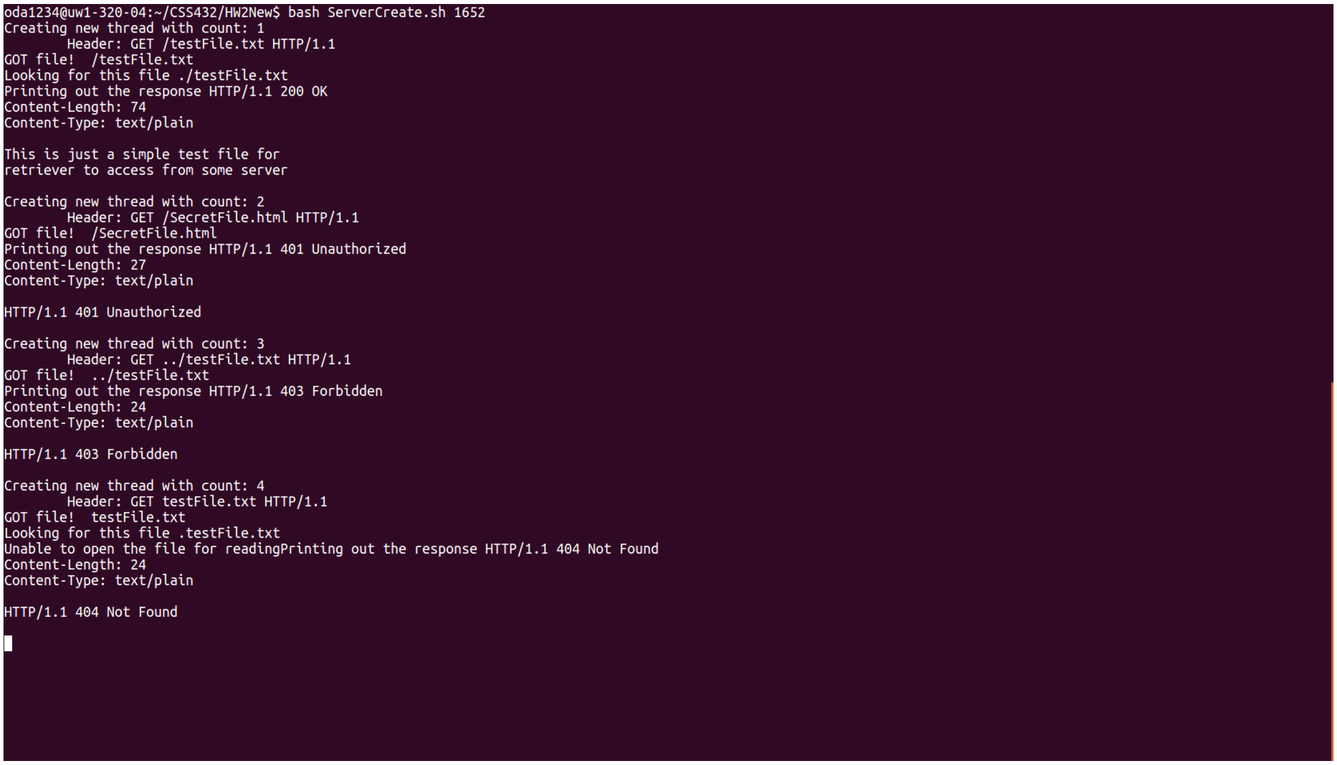
P.S: I talked about this to Professor Peng since I expected to see the html of the front page. He told me that it is not because of my request but because how the server is configured (something about proxy server) that make the index.html page to be in another place



```
oda1234@uw1-320-04:~/CSS432/HW2$ ./a.out uwb.edu /index.html 80
Found a connection. Breaking out
This request was made :
GET /index.html HTTP/1.1
Host: uwb.edu


HTTP/1.1 301 Moved Permanently
Location: http://www.uwb.edu/index.html
Content-Type: text/html
Content-Length: 170
printing out the body of the response and writing the content to content.txt
Finish writing the content of the response to the file
oda1234@uw1-320-04:~/CSS432/HW2$ cat content.txt
<html><head><title>Document Moved</title></head><body><h1>Document Moved</h1><p>The document has moved <a href="http://www.uwb.edu/index.html">here</a>.</p></body></ht
ml>oda1234@uw1-320-04:~/CSS432/HW2$ 
```

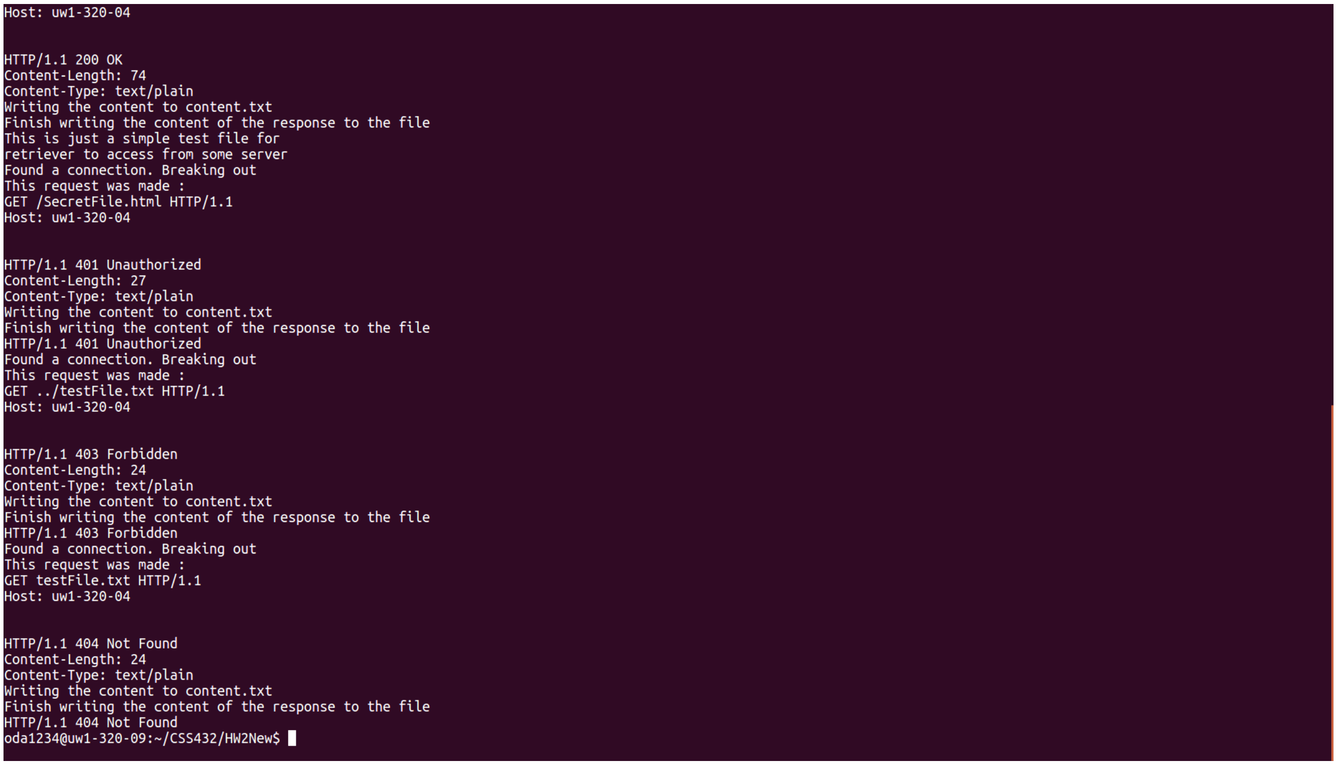
Illustration 6: Retriever access a real website

3), 4), 5), 6) will be grouped together. Below will be two screenshots of the server response and also the retriever response



```
oda1234@uw1-320-04:~/CSS432/HW2New$ bash ServerCreate.sh 1652
Creating new thread with count: 1
        Header: GET /testFile.txt HTTP/1.1
GOT file!  /testFile.txt
Looking for this file ./testFile.txt
Printing out the response HTTP/1.1 200 OK
Content-Length: 74
Content-Type: text/plain

This is just a simple test file for
retriever to access from some server

Creating new thread with count: 2
        Header: GET /SecretFile.html HTTP/1.1
GOT file!  /SecretFile.html
Printing out the response HTTP/1.1 401 Unauthorized
Content-Length: 27
Content-Type: text/plain

HTTP/1.1 401 Unauthorized

Creating new thread with count: 3
        Header: GET ../testFile.txt HTTP/1.1
GOT file!  ../testFile.txt
Printing out the response HTTP/1.1 403 Forbidden
Content-Length: 24
Content-Type: text/plain

HTTP/1.1 403 Forbidden

Creating new thread with count: 4
        Header: GET testFile.txt HTTP/1.1
GOT file!  testFile.txt
Looking for this file .testFile.txt
Unable to open the file for readingPrinting out the response HTTP/1.1 404 Not Found
Content-Length: 24
Content-Type: text/plain

HTTP/1.1 404 Not Found

```

Illustration 7: Server response for multiple request from retriever

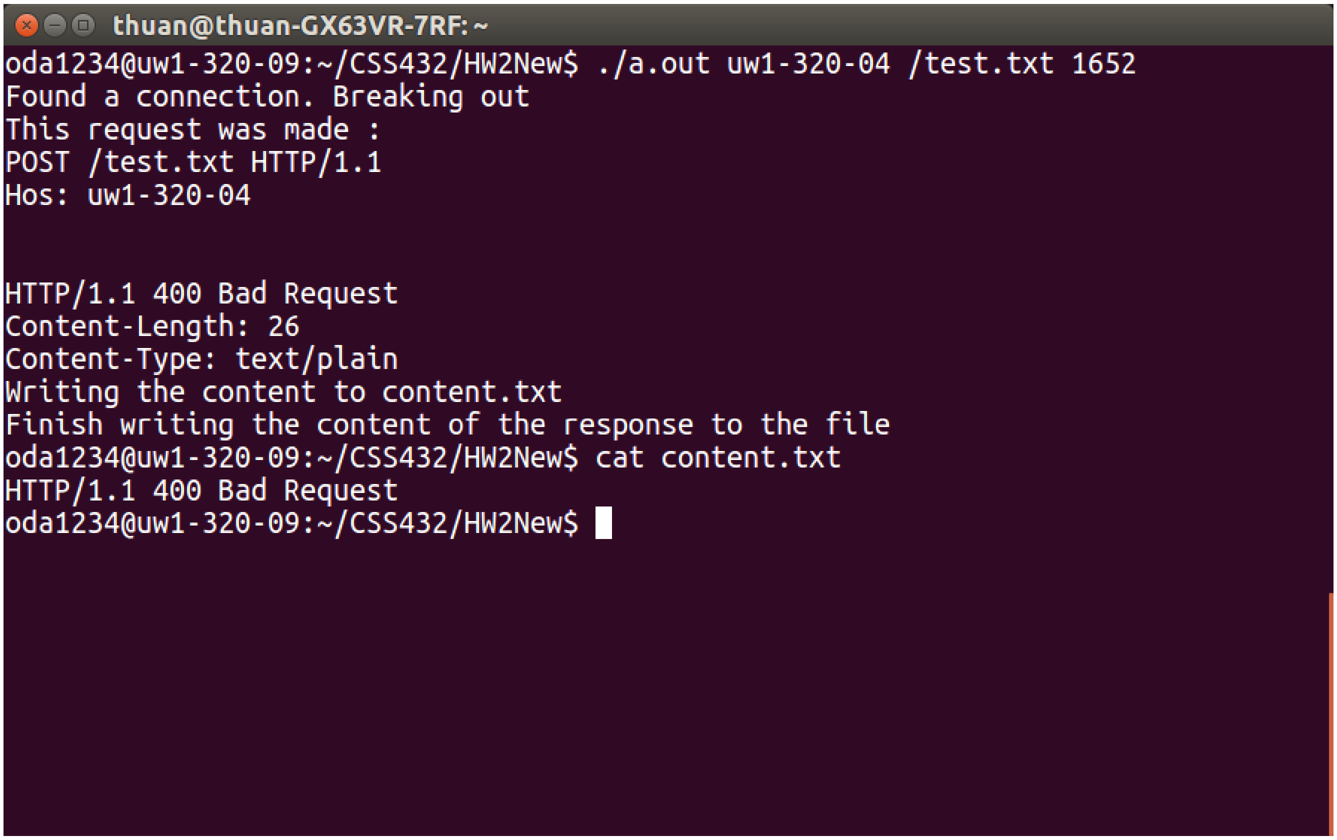


```
Host: uw1-320-04


HTTP/1.1 200 OK
Content-Length: 74
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
This is just a simple test file for
retriever to access from some server
Found a connection. Breaking out
This request was made :
GET /SecretFile.html HTTP/1.1
Host: uw1-320-04


HTTP/1.1 401 Unauthorized
Content-Length: 27
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
HTTP/1.1 401 Unauthorized
Found a connection. Breaking out
This request was made :
GET ../testFile.txt HTTP/1.1
Host: uw1-320-04


HTTP/1.1 403 Forbidden
Content-Length: 24
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
HTTP/1.1 403 Forbidden
Found a connection. Breaking out
This request was made :
GET testFile.txt HTTP/1.1
Host: uw1-320-04


HTTP/1.1 404 Not Found
Content-Length: 24
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
HTTP/1.1 404 Not Found
oda1234@uw1-320-09:~/CSS432/HW2New$ 
```

Illustration 8: Retriever request and response received from server

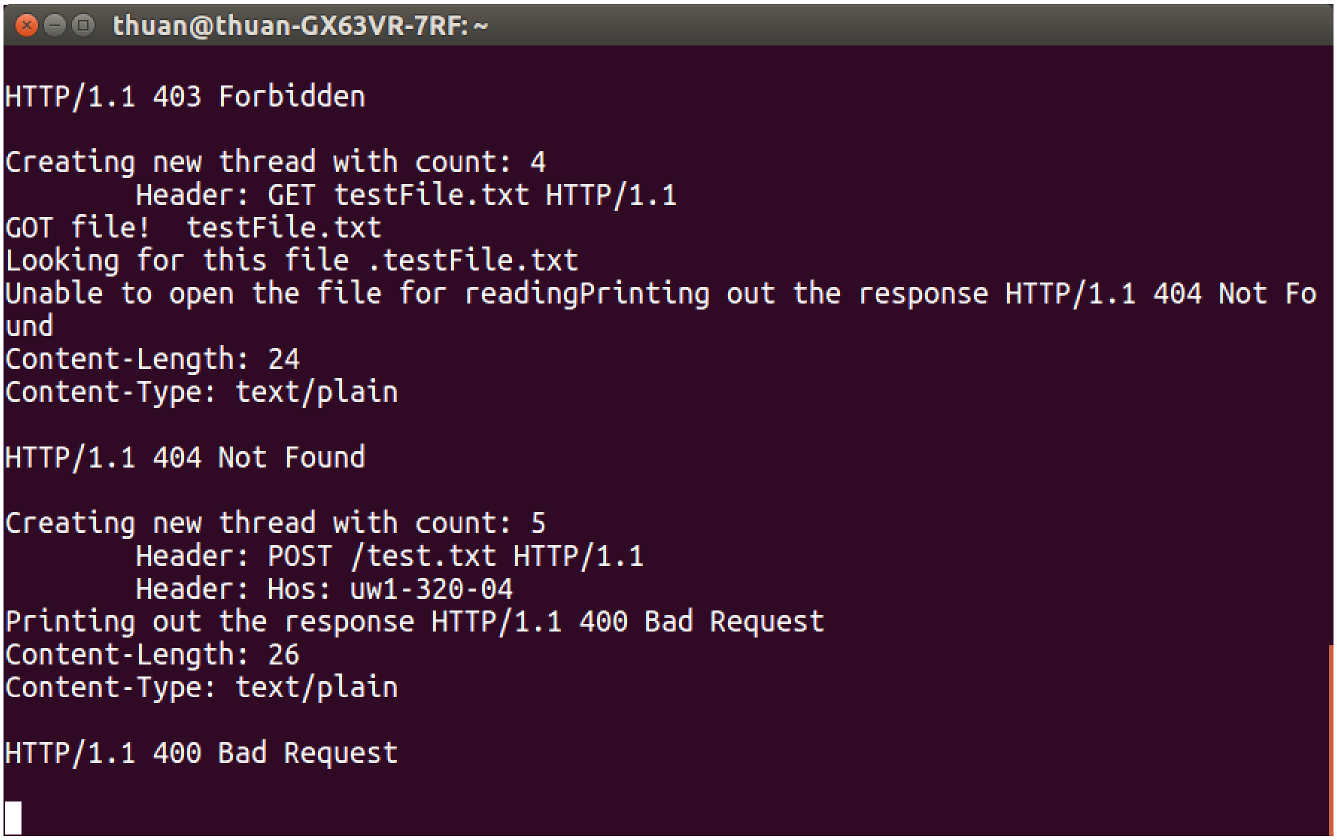
7) Retriever sending malformed request to server

For this part, since my Retriever was constructed to send using the correct format, I will demo this by alternating the original retriever to post request. I will provided a bad version of the retriever



```
oda1234@uw1-320-09:~/CSS432/HW2New$ ./a.out uw1-320-04 /test.txt 1652
Found a connection. Breaking out
This request was made :
POST /test.txt HTTP/1.1
Hos: uw1-320-04


HTTP/1.1 400 Bad Request
Content-Length: 26
Content-Type: text/plain
Writing the content to content.txt
Finish writing the content of the response to the file
oda1234@uw1-320-09:~/CSS432/HW2New$ cat content.txt
HTTP/1.1 400 Bad Request
oda1234@uw1-320-09:~/CSS432/HW2New$ 
```

Illustration 9: Retriever sending a bad request



```
HTTP/1.1 403 Forbidden

Creating new thread with count: 4
        Header: GET testFile.txt HTTP/1.1
GOT file!  testFile.txt
Looking for this file .testFile.txt
Unable to open the file for readingPrinting out the response HTTP/1.1 404 Not Fo
und
Content-Length: 24
Content-Type: text/plain

HTTP/1.1 404 Not Found

Creating new thread with count: 5
        Header: POST /test.txt HTTP/1.1
        Header: Hos: uw1-320-04
Printing out the response HTTP/1.1 400 Bad Request
Content-Length: 26
Content-Type: text/plain

HTTP/1.1 400 Bad Request

```

Illustration 10: Server receive a bad request