CS425 - Project 1 Report

Nishant Gupta, nishgu@iitk.ac.in, 13447

August 19, 2016

1 Implemented Options

1.1 Mandatory

- Supports GET method
- Supports Persistent Connections and multiple clients
- Provides Content-length and content-Type fields in response
- Provides appropriate Status-Code and Response-Phrase values in response to errors.

1.2 Optional

- Allow the server port and document base directory to be initialized at start up
- Include the Date and Server fields in the Response message header
- Implemented the HEAD method
- Implemented the POST method
- Reply with a directory listing if a directory is the requested resource
- Reply with a hyperlinked directory listing if a directory is the requested resource

2 Optional feature details

2.1 Server port and root directory initialization :

You can optionally give port with `-p` and root with `-r`. Default values are: `PORT=9576` and `ROOT=./test`, make sure `test` directory exists.
eg: `bin/server -p 9577 -r test/`

2.2 Date and Server fields :

Server is sent as Alchemist and Date is sent in the format ”Thu, 18 Aug 2016 18:52:47 GMT”.

2.3 POST Method :

In, POST request, message-body is written to `abs-path` of URI relative to root directory.
See Appendix

2.4 Hyperlinked directory :

A clickable list of all the filenames is is returned on GET request for a directory. Spaces in filenames and directories are not supported.

3 Test results

3.1 Multiple Clients

See Multiple clients and persistent connections of Appendix for Verification image and log. It can be seen in chrome network inspector image that multiple requests complete at almost same time. This means that server supports multiple connections.
Also Stderr log of opening `index.html` in firefox shows that multiple requests were served to same client, which proves that persistent connections work and the clients requests are intermingled, which proves that multiple simultaneous connections work.

3.2 Webpage provided with project

Google Chrome Version 52.0.2743.116 (64-bit) was used for testing in this part

3.2.1 GET and Directory listing

These are Screenshots of Google Chrome showing index.html and directory listing working correctly.

3.2.2 POST request

After submitting form on `post_test.html`, following is the content of `test/post_test_file.txt`

```
TextEntry_1=Hola&TextEntry_2=This+is+some+text+for+testing+the+POST+method.%0D%0A&Item=Item_1
```

3.3 Testing using command line

httpie : A CLI HTTP client was used.

3.3.1 POST request

I generated a random file of 10^6 bytes and sent it using `httpie`, and compared the saved file to original file. They were same. POST request section of Appendix verifies that.

3.3.2 GET request

I made a GET request using httpie and it ran succesfully as can be seen in GET Request section of Appendix.

3.3.3 HEAD request

I made a HEAD request using httpie and it ran succesfully as can be seen in HEAD request section of Appendix.

3.4 Status Codes

I made different types of requests that would yield different status codes using httpie. They can be seen in Status Codes section of Appendix.

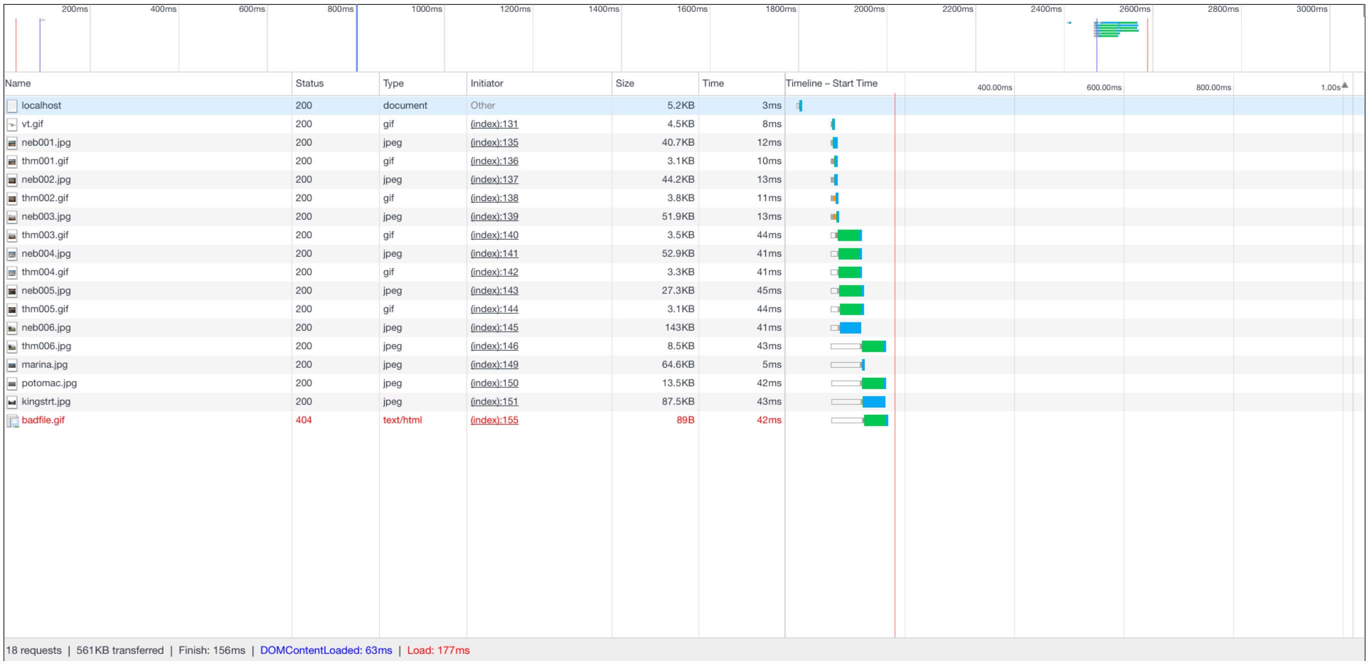
- I recognize GET, HEAD, POST, PUT, OPTIONS, CONNECT, DELETE. Only three of them are implemented, others give `501 : Not implemented`
- If anything other than these seven comes, it is BAD request and `400 : Bad Request` is returned. Bad request is also returned if any protocol other than HTTP/1.0 or HTTP/1.1 is requested
- I have put my entire parser in a try catch block, if parser fails I send `500 : Internal Server Error` in catch block.
- If Parser succeeds, I process corresponding request and then `404` or `200` is sent by corresponding method.

3.5 Summary

- I have not supported spaces and other special characters in resources requested, so it will give 404.
- Also, in hyperlinked directory listing, link will be wrong if there are spaces or other special characters in the name.
- Everything else is working as it should.
- I had no way to check Internal Server Error.
- All the other Status codes are working as they should.
- Multiple clients and persistent connections are working correctly, I have not implemented timeout.

4 Appendix

4.1 Multiple clients and persistent connections



Name	Status	Type	Initiator	Size	Time	Timeline – Start Time
localhost	200	document	Other	5.2KB	3ms	
vt.gif	200	gif	(index):131	4.5KB	8ms	
neb001.jpg	200	jpeg	(index):135	40.7KB	12ms	
thm001.gif	200	gif	(index):136	3.1KB	10ms	
neb002.jpg	200	jpeg	(index):137	44.2KB	13ms	
thm002.gif	200	gif	(index):138	3.8KB	11ms	
neb003.jpg	200	jpeg	(index):139	51.9KB	13ms	
thm003.gif	200	gif	(index):140	3.5KB	44ms	
neb004.jpg	200	jpeg	(index):141	52.9KB	41ms	
thm004.gif	200	gif	(index):142	3.3KB	41ms	
neb005.jpg	200	jpeg	(index):143	27.3KB	45ms	
thm005.gif	200	gif	(index):144	3.1KB	44ms	
neb006.jpg	200	jpeg	(index):145	143KB	41ms	
thm006.jpg	200	jpeg	(index):146	8.5KB	43ms	
marina.jpg	200	jpeg	(index):149	64.6KB	5ms	
potomac.jpg	200	jpeg	(index):150	13.5KB	42ms	
kingstrt.jpg	200	jpeg	(index):151	87.5KB	43ms	
badfile.gif	404	text/html	(index):155	89B	42ms	

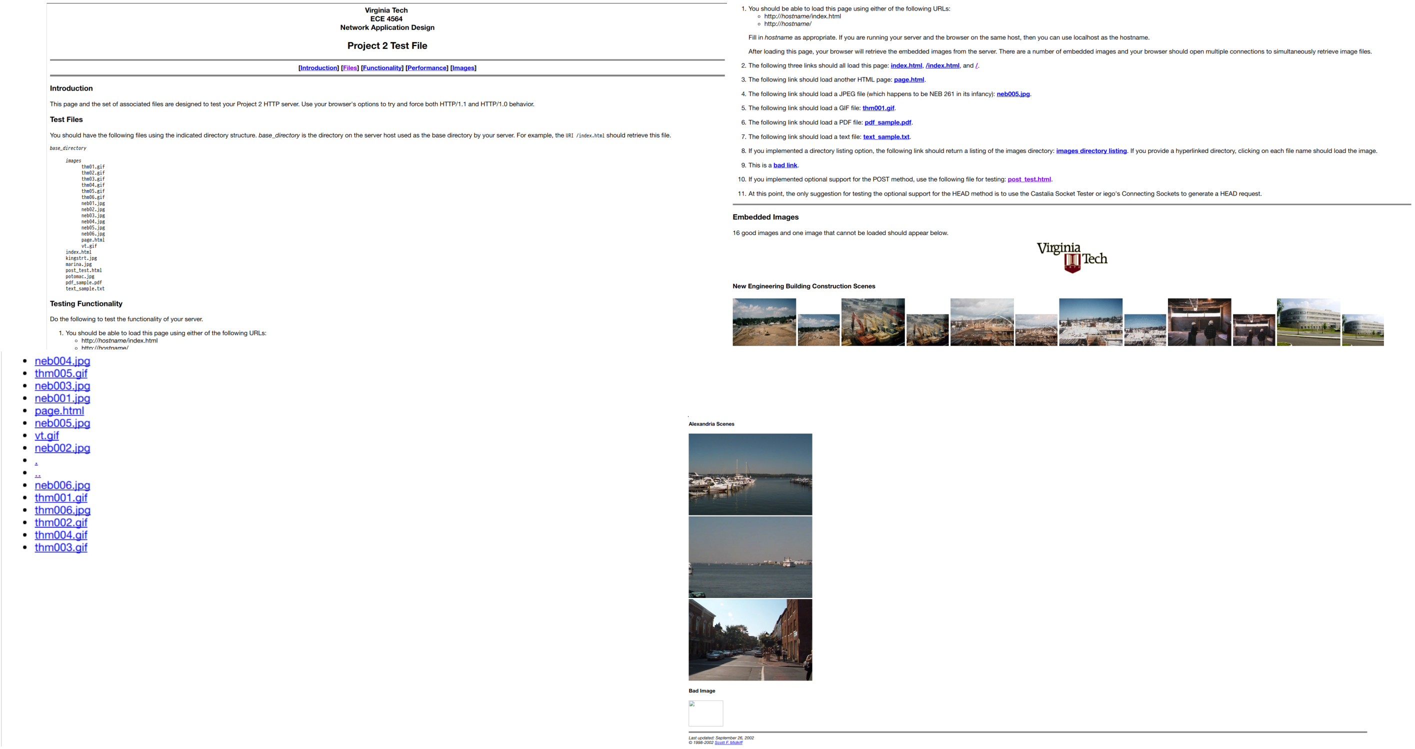
18 requests | 561KB transferred | Finish: 156ms | DOMContentLoaded: 63ms | Load: 177ms

Following is the condensed stderr log of opening `index.html` in firefox.

```
Connected to client 6
Request 1 from Client 5
Request 2 from Client 5
Connected to client 6
Connected to client 7
Connected to client 8
Connected to client 9
Connected to client 10
Request 1 from Client 10
Request 1 from Client 9
Request 1 from Client 8
Request 1 from Client 6
Request 1 from Client 7
Request 3 from Client 5
Request 2 from Client 8
Request 2 from Client 10
Request 2 from Client 9
Request 2 from Client 7
Request 2 from Client 6
Request 3 from Client 6
Request 4 from Client 5
Request 3 from Client 8
Request 4 from Client 6
Request 5 from Client 5
Request 5 from Client 6
Closed Client
Closed Client
Closed Client
Closed Client
Closed Client
Closed Client
```

4.2 GET Request :

GET request made using Chrome



Virginia Tech
ECE 4564
Network Application Design

Project 2 Test File

[Introduction] [Files] [Functionality] [Performance] [Images]

Introduction

This page and the set of associated files are designed to test your Project 2 HTTP server. Use your browser's options to try and force both HTTP/1.1 and HTTP/1.0 behavior.

Test Files

You should have the following files using the indicated directory structure. *base_directory* is the directory on the server host used as the base directory by your server. For example, the `URI /index.html` should retrieve this file.

```
base_directory

    images
        thm01.gif
        thm02.gif
        thm03.gif
        thm04.gif
        thm05.gif
        thm06.gif
        neb01.jpg
        neb02.jpg
        neb03.jpg
        neb04.jpg
        neb05.jpg
        neb06.jpg
        page.html
        vt.gif
    index.html
    kingstrt.jpg
    marina.jpg
    post_test.html
    potomac.jpg
    pdf_sample.pdf
    text_sample.txt
```

Testing Functionality

Do the following to test the functionality of your server.

1. You should be able to load this page using either of the following URLs:
   - http://*hostname*/index.html
   - http://*hostname*/

1. You should be able to load this page using either of the following URLs:
   - http://*hostname*/index.html
   - http://*hostname*/

   Fill in *hostname* as appropriate. If you are running your server and the browser on the same host, then you can use localhost as the hostname.

   After loading this page, your browser will retrieve the embedded images from the server. There are a number of embedded images and your browser should open multiple connections to simultaneously retrieve image files.

2. The following three links should all load this page: index.html, /index.html, and /.
3. The following link should load another HTML page: page.html.
4. The following link should load a JPEG file (which happens to be NEB 261 in its infancy): neb005.jpg.
5. The following link should load a GIF file: thm001.gif.
6. The following link should load a PDF file: pdf_sample.pdf.
7. The following link should load a text file: text_sample.txt.
8. If you implemented a directory listing option, the following link should return a listing of the images directory: images directory listing. If you provide a hyperlinked directory, clicking on each file name should load the image.
9. This is a bad link.
10. If you implemented optional support for the POST method, use the following file for testing: post_test.html.
11. At this point, the only suggestion for testing the optional support for the HEAD method is to use the Castalia Socket Tester or iego's Connecting Sockets to generate a HEAD request.

Embedded Images

16 good images and one image that cannot be loaded should appear below.

New Engineering Building Construction Scenes

Alexandria Scenes

Bad Image

Last updated: September 26, 2002
© 1998-2002 Scott F. Midkiff

- neb004.jpg
- thm005.gif
- neb003.jpg
- neb001.jpg
- page.html
- neb005.jpg
- vt.gif
- neb002.jpg
- .
- ..
- neb006.jpg
- thm001.gif
- thm006.jpg
- thm002.gif
- thm004.gif
- thm003.gif

GET request made using httpie

```
 ~/c/n/project1 $ http GET  :9577/post_test_file.txt Connection:close -v
GET /post_test_file.txt HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: close
Host: localhost:9577
User-Agent: HTTPie/0.9.4



HTTP/1.1 200 OK
Connection: Close
Content-Length: 93
Content-Type: text/plain
Date: Thu, 18 Aug 2016 18:52:47 GMT
Server: Alchemist

TextEntry_1=Hola&TextEntry_2=This+is+some+text+for+testing+the+POST+method.%0D%0A&Item=Item_1
```

4.3 POST request :

POST Request made using httpie

```
 ~/c/n/project1 $ base64 /dev/urandom | head -c 1000000 >! file.txt
 ~/c/n/project1 $ cat file.txt | http POST :9577/file.txt -p=Hhb
POST /file.txt HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 1000000
Content-Type: application/json
Host: localhost:9577
User-Agent: HTTPie/0.9.4

HTTP/1.1 200 OK
Content-Length: 36
Content-Type: text/html

<h1>Content written Succesfully</h1>

 ~/c/n/project1 $ diff file.txt test/file.txt
 ~/c/n/project1 $
```

4.4 HEAD request :

HEAD request made using httpie

```
 ~/c/n/project1 $ http HEAD  :9577/ Connection:close -v
HEAD / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: close
Host: localhost:9577
User-Agent: HTTPie/0.9.4



HTTP/1.1 200 OK
Connection: Close
Content-Length: 5211
Content-Type: text/html
Date: Thu, 18 Aug 2016 18:57:32 GMT
Server: Alchemist
```

4.5 Status Codes :

Following are some of httpie requests that showcase diffrent status codes

- Not Implemented

```
 ~/c/n/p/doc $ http PUT  :9577/ -v
PUT / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 0
Host: localhost:9577
User-Agent: HTTPie/0.9.4

HTTP/1.0 501 Not Implemented
Connection: keep-alive
Content-Length: 29
Content-Type: text/html

<h1>501: Not Implemented</h1>
```

- Bad Request

```
 ~/c/n/p/doc $ http PUTY  :9577/ -v
PUTY / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 0
Host: localhost:9577
User-Agent: HTTPie/0.9.4

HTTP/1.0 400 Bad Request
Connection: keep-alive
Content-Length: 25
Content-Type: text/html

<h1>400: Bad Request</h1>
```

- Not Found

```
 ~/c/n/p/doc $ http GET  :9577/fdajlkdasj -v
GET /fdajlkdasj HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:9577
User-Agent: HTTPie/0.9.4

HTTP/1.0 404 Not Found
Connection: keep-alive
Content-Length: 23
Content-Type: text/html

<h1>404: Not Found</h1>
```

4.6 Source code :

```
C++ code goes here.
```