**371 Project Test Procedures**

*All testing was done with the curl command line tool*

**200 Ok**

A computer screen shot of a black screen

Description automatically generated

**Procedure**:

send get request to localhost and port running application

**304 Not Modified**

**A screenshot of a computer

Description automatically generated**

**Procedure**:

send get request with “If-Modified-Since” header line. Our server simply checks if this header exists in the request and answers with 304 not modified.

**400 Bad Request**

**A computer screen with white text

Description automatically generated**

**Procedure**:

Send an HTTP request to the server that it is not able to handle. In this situation, we send an HTTP delete request and since our server is unable to handle it, a 400 error is returned.

**403 Forbidden**

Authorization Header present

**A computer screen with white text

Description automatically generated**

Authorization Header not present

**A screenshot of a computer program

Description automatically generated**

**Procedure**:

To simulate authorization being required, we created another html filed named test\_auth.html.

If this file is requested, our server checks if the Authorization header is present. If it is, then the file is served, otherwise a 403 error is returned

**404 Not Found**

**A computer screen with white text

Description automatically generated**

**Procedure:**

Request a file that does not exist in the server directory.

**411 Length Required**

Content Length provided

A screenshot of a computer

Description automatically generated

Content Length not provided

A computer screen shot of a black screen

Description automatically generated

**Procedure:**

Send a POST request to test\_content\_len\_req.html.

If the request contains the Content-Length header, then the file is served. Otherwise, a 411 error is returned.

Step three

a)The difference in request handling in a proxy server and a web server that host my files is the proxy cache, as the proxy server maintains a cache of previously requested resources. Since the request resource is in a cache, the proxy will return the cached copy to the client so it doesn’t have to request a call from the origin server. For the proxy cache we can add headers in the responses to determine these. This will include a Cache-Control and Expires in the response header when the page is first cached.   
  
To create a minimal proxy server using Cache-Control and Expires. First

I will create a proxy server that takes in a proxy ip, proxy port, the target ip and the target port.

Next the proxy will determine if the destination host is currently in the cache or not, if its currently in the cache, then it will return the cache back to the proxy port page. If not then we must connect to the target ip and target port, where we will add the Cache-Control and an Expiry date for the port.

b) In order to test the working proxy, first you must run both the current main.py and the proxy.py python file. Next you must access page using localhost:8080/test.html for the page to be cached. Using the google chrome network feature, this is what the network would look like for the test.html page

A screenshot of a computer

Description automatically generated

Now you have to open a separate page using the localhost:8080/test.html and check the network tab again. This is what the cache based will look like with the response headers and request headers.

A screenshot of a computer

Description automatically generated

From this image you can determine that there is a current Cache of the recently requested page, this can be determined from the request header that states that its currently a cache.

An alternative way to test the server-side functionality is to use Postman to simulate client request. Using Postman can simulate a client-side request and send the headers created in the proxy. Below is a request with the headers for the server and proxy.

Server:

A screenshot of a computer

Description automatically generated

Proxy:

A screenshot of a computer

Description automatically generated