Below is the information pertaining to the contributions made by each team member and the test cases ran on the client and server.

**Overview:**

For this project, both team members decided to work on their own versions of the HTTP server and client code implementation with the plan to identify the better of the two codes to use for the submission. Both team members were successful in developing and implementing their code with successful compilation of their codes. As a team, we decided to use Neha’s HTTP Proxy server and client code for the submission. Attached you will find the HTTP server/client code, ChatGPT associated codes, and this README file. For the rest of the project, James completed the README file associated with the attached programs – to include running the test cases.

**Precursor:**

Prior to running any test cases, the team needed to verify that all code was working correctly and would compile. The *makefile* successfully creates the associated executable files with respect to the HTTP Proxy Server and HTTP Proxy Client. Command for running the makefile is “*make”*.

**Beginning Stages:**

Overall syntax for running the executables:

Server: *./<executable> <IP address> <Port Number>*

Client: *./<executable> <IP address> <Port Number> <URL>*

**Verification Test case from Recitation Slides:**

**A computer screen shot of a computer code

Description automatically generated**A black screen with white text

Description automatically generated

**Test Cases:**

***Test Case 1 Cache Hit:***

Using the chached URL from the verification test:

Client:

A computer screen with white text

Description automatically generated

Server:  
A computer screen shot of a black screen

Description automatically generated

***Test Case 2 Request Not in Cache:***

**Client:**

A screen shot of a computer

Description automatically generated

**Server:**

A screen shot of a computer

Description automatically generated

***Test Case 3 Miss with 10 items in Cache:***

**Client (current working directory):**

A screen shot of a computer

Description automatically generated

**Server:**

A screenshot of a computer program

Description automatically generated

The above snippets reflect that there has been more than 10 requests made to the server; the server code doesn’t show as many requests. Additionally, with more than 10 cache requests, the server prompts for a cache miss, while maintaining the information for the original cache/URL requests.

***Test Case 4 Stale Expires Header:***

**Client:**

A computer screen shot of a computer code

Description automatically generated

**Server:**

A computer screen with white text

Description automatically generated

We verified the above URLs to ensure they were expired; however, the server automatically updated them and provided them to the client with fresh data and updated the cache appropriately.

***Test Case 5 Stale Entry without Expires Header:***

**A computer screen shot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedClient: Server:**

We came back to do this test case last on cache entries which were expired. The output reflects different access and expiration times for the client to access the data from the server compared to “Test Case 4”.

***Test Case 6 Cache Entry without Expires Header:***

**Client:**

A black screen with white text

Description automatically generated

**Server:**

A screenshot of a computer

Description automatically generated

The above website was last updated in 2022; The below screenshot reflects this URL doesn’t have an “expires” header:

A computer screen with white text

Description automatically generated

**Test Case 7 Multiple Simultaneous Requests:**

A screenshot of a computer program

Description automatically generated

***Test Case 8 (BONUS) If-Modified-Since:***

**Client:**

A computer screen shot of a computer code

Description automatically generated

**Server**:

A computer screen shot of a computer

Description automatically generated

Since the client requested the same URL twice, the server reflected that it was already present in the cache and reflects the “If-Modified-Since” statement along with the date that it corresponds to.

This concludes the read me an all eight test cases.