Aaron Nguyen
1585632
Assignment 3 Writeup

Testing

For this lab the majority of the testing involved unit testing each function and using curl and netcat to test my code. I also used cout and printf to a good amount throughout my code to see the progression of my code as it is executed. If any error occured I would be able to identify at what position malfunctioned. Curl was used in the later stages of the program test GET and PUT header processing and responses in the server when ALIAS/PATCH was implemented. The unit tests were used on the internal functions of the main functions that took care of PATCH. Most of the testing was spent using netcat to make sure the program was still functional while also testing multithreading.

Explain the difference between fully resolving a name (to an httpname) when the name is created and the approach that you're taking for this assignment. Give an example of when it might be useful.

In my program to handle naming and aliases I would always send the filename in the alias_name check function in my program, and check for aliases and original filenames. In the creation of the aliases I would always parse out the existing/old and new file names/aliases. I also used the new alias as the key to my hashmap and the existing filename as my value. The overall functionality of my program creates a dictionary. Which has many uses in many applications. For example in phones the contact list, the name of a person is the key and phone number is the value. It can also be said the name is the alias and the phone number is "existing filename."

What did you learn about system design from this class? In particular, describe how each of the basic techniques (abstraction, layering, hierarchy, and modularity) helped you by simplifying your design, making it more efficient, or making it easier to design.

In this class I learned about the intricacies that go into system design which was for the most part very interesting. Also it was nice that I was able to apply some of the things taught in lecture in my program. Taking this class definitely made me a better programmer. Before I never really planned a general structure to my code, so the end result usually ended relatively messy

and difficult to read code. In addition because I began to plan out my code I also started to create more layers to my code by creating more unique functions that modularized my whole code. In addition in each function most of the time would continue to call more unique functions to complete the tasks to keep the code clean. Which was my attempt to apply layering to my code, which for the most part greatly improved the readability and structure of my code. With all of this put together it made it very easy to fix, improve, or add more functionality to my code due how it was structured and designed based on the some of the concepts learned in class.