|  |
| --- |
| Iterative Socket Server |
| CNT 4504 Scott Kelly |
| Vicente Salcedo N01127407 Andrew Duchaussee N01139153 Nicholas Anderson n01439343 |

The Iterative Socket Server’s primary objective is to process requests from clients in a serial manner. It will service one connection and respond to that request before accepting a new client connection. The main objective of the single threaded program is to receive data from a client and to respond to the request without worrying about any other programs that might be trying to utilize the resource at the same time. The goal of a multi-threaded program is to send requests and create multiple client sessions.

An Iterative Socket Server can also be utilized to help provide an audience with a tool to practice coding as a group, gain a better understanding of how servers are constructed, and how they can be stress tested, as was done for this project. The following reading material will explain how the programs where designed and created. Afterwards it will detail how the programs were tested, what data was collected, and how it was used to improve the final product.

The single thread program is very simple and focuses around a single while loop which runs the processes. It receives a request from a client. It analyzes the request and determines what is being asked of the server. Once it determines the proper response, it responds by returning the appropriate data based on the received request.

The multi-threaded program is much more complicated. A class was created to represent a single thread in the multi-threaded program. When the class is instantiated for each thread, it establishes a new socket and it records when it started data collection. The client picks a random request which it sends to the server. The server formats the stream output so that the information is readable by the user.

Both the server code and the client code was designed in a much segmented way to help each code block stay focused on its allocated objective. Formatting the code this way made it very easy for an outside programmer to come along and read the code and to determine what task each code block performed. This enabled teammates to assist each other when trying to get the project working as a whole. Each block of code has a well-defined purpose. If anything goes wrong, it is very easy to pinpoint which code block is creating the error. This design allowed for streamlined code design, as well as a more efficient debugging process for the overall project.

To test the Iterative Server we devised a test plan to follow to ensure that it met the requirements that we had used to design the program. Because we had a test plan with predetermined outputs, when the program failed, we were able to include error warnings displayed by the program to make us aware that there was a failure and we needed to make program modifications. Once we were able to execute our complete test plan with no error warnings we were confident that our program and the connections that it made were as we had designed it. We chose to test it in this manner because by having a solid test plan in place before all development was done allowed us to make sure all requirements were fulfilled and were performed accurately and it actually helped minimize development time because we were more efficient in our development process.

In order to do data analysis, we collected turnaround time and average turnaround time to help solve the fundamental questions provided by this project. With turnaround time we set up a system to collect each inputs time and to add it to a variable which was incremented for each subsequent input. When execution was complete, the final number being how long the program took to complete every input. The Average turnaround time data was very easy to get after we had already gotten turnaround time because all that was need was to calculate the total and then find a way to divide by the number of inputs given, after that all we need to do was display information.

1. What affect, if any, does increasing the number of clients have on the Turn-around Time for individual clients?
   * Increasing the number of clients has a negative impact on the turnaround time for individual clients. We found the reason for this occurrence is the fact that each client has to wait for the one before it to output its answer for it can start adding to the overall time with each client in the row.
2. What affect, if any, does increasing the number of clients have on the Average Turn-around Time?
   * Increasing the number of clients has no impact on the on the Average Turn-around Time. The conclusion we had for this claim been that each client will complete its task and add on its time but because the average will in the end divide by how many times the function ran the average time will not change for each unit added.
3. What is the primary cause of the effect on individual client Turn-around Time and Average Turn-around Time?
   * The biggest factor our team has found through our testing is the type of request. Different request require different types of data collection and some requests take much more time than others. Very clearly some functions such as net stack took a very long time to send back its information whereas most of the others took next to no time at all.

This project allowed our group to learn and master new skill required to complete this assignment. The first thing we required was collaboration tools that were needed to work as a team in a distributed environment. We needed something so that we could all work on the code and not get lost nor duplicate the effort of another teammate. The collaboration tools we used where git, github, and Visual Studios. We used the code live share function and the “easy upload” function for sharing code and do side by side coding which was a very big game changer that made working together easy. We used git for code management. When one person finished working on a section of code, they would commit it to the git repository so that the others could come and see what they did and continue their work with the newly modified code base.

Another topic learned in the execution of this project was how a socket server works. We had thought that it was big complex thing, but it was rather simple in the end but can see how much harder this can get down a few levels.