4.0: *Software Development Plan*

**4.1 Plan Introduction**

This Software Development Plan provides the details of the planned development process for the client server application which sends files from a client to a server. Written from first principles, the application is split into two main programs: the client and the server. The client will take in files the user wishes to send over to the client, encode them into transmittable data, and attempt to send the file over an established connected. The server will be listening for incoming connection requests, either approve or deny various handshake requests, decode data back into the data’s initial state, and store this data is a pre-determined directory. In order to complete the application at hand, the following tasks listed in the project deliverables must be completed with success.

**4.1.1 Project Deliverables**

* Week 2
  + Project Proposal Document
    - A high-level overview of the application that is going to be developed, the primary features, the incentive behind the project, expectations and more of the basic behind the project
* Week 3
  + Golang Simple HTTP Client/Server
    - Create a simple, high level client, server application that are capable of communication with one another
* Week 4
  + Computer Networks research
    - Extensive research to gain a strong understanding of computer network fundamentals, with a focus on the Network Layer and the Transport Layer of the TCP/IP stack
    - Utilize various text books and free, online resources to gain the appropriate knowledge
* Week 5
  + Requirements Specification Document (Initial)
    - This document establishes the tools needed to create the application at hand and allows the client to be aware of all of the necessities
  + Golang research
    - Gain a proficient level of knowledge of the programming language Go
    - Have a strong understanding of the native package “net” to gain access to UNIX programming functions and objects
* Week 7
  + TCP Header
    - Develop a TCP Header struct that is capable of holding all of the necessary information to replicate a traditional TCP header
* Week 8
  + Sockets
    - Develop a client that is capable of targeting a specified socket
    - Develop a server that is capable of listening to a specified socket
* Week 9
  + Software Development Plan Document (Initial)
    - A detailed document describing the process of creating the application
    - Will establish deadlines and expectations of when certain aspects of the application and documentation are to be completed
* Week 10
  + Encoding/Decoding
    - Develop an encoder that can encoding a TCP Header object into some transferable, binary value
    - Develop a decoder that can translate an incoming, binary value into some TCP Header object
* Week 11
  + Software Development Plan Document (Updated)
    - Any needed changes to the Software Development Plan Document that were not implemented in the initial submission
* Week 12
  + 3-way handshake
    - Program a 3-way handshake required to be conducted in order for a client to connect to a server
  + Basic message
    - Send a basic message within a TCP Header object from a client to a server
* Week 13
  + Requirements Specification Document (re-submit)
    - Any needed changes to the Requirements Specification Document that were not implemented in the initial submission
  + Send a file
    - Send a file from within a TCP Header object from a client to a server
* Week 15
  + Multiple clients
    - Be able to handle multiple clients connected to one server and have the server capable of handling various amounts of traffic
* Week 16
  + Bugs/Errors
    - Clean up any final issues/bugs within the application
  + FINAL Product Delivery (Final Report and Code)
    - Have the client server application complete and fully up to date on its Github repository
    - Complete the final report
* Week 17
  + FINAL Project Presentation
    - Create a poster displaying an overview of the application, technologies that are utilizes, and more to allow guests to understand the project
* Ongoing
  + SCRUMs
    - Updates on what was completed from the last SCRUM, what is currently being worked on, and what is going to be worked on in the near future
  + Written Status Reports
    - Updated every other Tuesday starting Week 8
    - A detailed outline of what exactly has been completed since the last Written Status Reports, and what is expected to be worked on next

**4.2 Project Resources**

Listed down below are the hardware and software resources required for the success of the project.

**4.2.1 Hardware Resources**

*Development Machine*

|  |  |
| --- | --- |
| **Category** | **Requirement** |
| Processor | Intel Pentium 4 Processor or Later |
| RAM | 128 MB |
| Hard Drive Space | 256 MB |
| Display | Any |

*Client Machine*

|  |  |
| --- | --- |
| **Category** | **Requirement** |
| Processor | Intel Pentium 4 Processor or Later |
| RAM | 64 MB |
| Hard Drive Space | 128 MB |
| Display | Any |

*Server Machine*

|  |  |
| --- | --- |
| **Category** | **Requirement** |
| Processor | Intel Pentium 4 Processor or Later |
| RAM | 64 MB |
| Hard Drive Space | 256 MB |
| Display | Any |

**4.2.2 Software Resources**

*Development Machine*

|  |  |
| --- | --- |
| Category | Requirement |
| CLI | Bash |
| Operating System | Mac OS X10.13.3 or later; Ubuntu 16.04.3 or later |
| Text Editor | Any |

*Client Machine*

|  |  |
| --- | --- |
| Category | Requirement |
| CLI | Bash |
| Operating System | Mac OS X10.13.3 or later; Ubuntu 16.04.3 or later |
| Text Editor | Vim |

*Server Machine*

|  |  |
| --- | --- |
| Category | Requirement |
| CLI | Bash |
| Operating System | Mac OS X10.13.3 or later; Ubuntu 16.04.3 or later |
| Text Editor | Vim |

**4.3 Project Organization**

The client server application project is split into the following major functions: research, TCP Header, sockets, client/server, and communication. Research involves teaching oneself the skills required for this project. Some of these skills include computer networks fundaments, proficiency in Golang, and more. Following that, the TCP Header struct needs to be create. This is the key object that will be used in both the client and the server in order to have a uniform struct for communication. Next, sockets must be created in order for clients and server to know where to live on their respective machines. Encoding and decoding algorithms will be needed in order to process data to and from these sockets. With these backend settings completed, the client and server can be completed. A 3-way handshake will need to be developed in order to ensure that only desired clients are connecting to a server. Lastly, communication between a client and a server will be completed. This involves initially sending a basic TCP Header from a client to a server. Next, a file will be sent from a client to a server and stored on the server’s end. The server will then be adjusted to handle the ability of listening to multiple clients.

**4.4 Project Schedule**

This section provides schedule information for the client server application project.

**4.4.1 GANTT Chart**

Refer to the file **gant-chart.xlsx** within **documents/** of the client-server-application repository.

**4.4.2 Task/Resource Table**

Everything regarding the client server application will be directly handled by Thomas O’Brien. He will reach out to his professors, fellow students, textbooks, and the internet if help is required.