Software Requirements Specification

For

File Sharing System

Version 1.0

Prepared By: Pranav Barathwaj P, 20BCE1351

Panav Sinha, 20BCE1640

Under the Supervision of Dr.M.Braveen

24/09/2021

Acknowledgement

This SRS is the result of a class project carried out by VIT Chennai's Fall 2021-22 class of CSE3001

– Software Engineering under the supervision of Dr. M. Braveen.

Table of Contents

1.Introduction

1.1 Purpose

The purpose of this project is to create a simple file sharing system that shares any format of files between any number of systems. A system can be a laptop, mobile phone etc. Basic feature involved in this system is the secure connection between a server system and various client systems, through which transfer of files can take place.

There are three subsystems present and they include:
-The Client System
-The Server System
-The Data Transfer

1.2 Document Convention

This document will use IEEE format. For clarity, acronyms and technical jargon, deemed uncommon by the author, will be annotated and included in the glossary. The format for headings is as followed:
Major headings are in **bold 17pt font**, and concurrent headings in **bold 14 pt font**. Sections are in the format of x.y, where x and y are real, positive integers.

1.3 Intended Audience and Reading Suggestions

This Software Requirements Specification document is intended for software engineers, system testers and software designers in developing, testing, and producing the File Sharing System. It is suggested to read the sections sequentially, and to reference the appendices as one progresses, in order to clarify jargon terms and definitions. The project is also a prototype for systems present within the similar domain.

1.4 Project Scope

This SRS details the development of the File Sharing System project and the three subsystems. The main feature includes file sharing across a central server present in a network, consisting of appropriate security protocols, to protect it from hackers. The system is developed for flexible use, that is the ability to operate and transfer files across any OS. The FSS uses a GUI.

The scope of the Server System is to call on different devices within the network and establish itself as a unique server. This enables other client systems to connect to the server system.

The scope of the Client System is to call on different devices and search for an appropriate server. By following appropriate security protocols, the client can establish a connection and send files across the server to any other clients present within the same client-server architecture.

The scope of the Data Transfer is the transfer of files across different client systems present within the network. The transfer of files follows HTTPS protocol, which is encrypted and secure. The concept of a shared clipboard also persists.

1.5 References

HTTPS - https://en.wikipedia.org/wiki/HTTPS
TCP/IP - https://www.techtarget.com/searchnetworking/definition/TCP-IP
GoLang GUI - https://github.com/webview/webview
GoLang Documentation - https://golang.org/doc/

2. Overall Description

2.1 Product Perspective

The File Sharing System is a web-based system using Client-Server Model that comprises of three subsystems. The system allows the user to connect to a server by following a process that establishes a secure connection. This enables the user to transfer files across the network. The same system can also disconnect from the server and transfer its connection to another server.
The FSS provides an infrastructure that provides a secure connection (HTTPS) and allows the user to interact with a GUI. It also enhances flexibility by operating across various Operating Systems and devices.

2.2 Product Functions

The FSS has three subsystems and these subsystems shall perform the following functions:

-The Server System is an important module that hosts file sharing within a client-server model. This module gets requests from all devices within the network to identify its uniqueness. If it has a unique name, it proceeds connecting with client systems by generating a random code number through which client systems can connect.

-The Client System acts as the source and sink in a file sharing architecture. This system gets requests from all devices within the network to find an appropriate server through which data transfer can take place. After entering an appropriate random code word generated by the server system, it establishes its connection.

-The Data Transfer module presents the logical aspect of data transfer. This module follows HTTPS for encrypted and secure transfer of files. This takes place only when the above-mentioned modules are connected. A shared clipboard is present to ease the transfer between users.

2.3 User Classes and Characteristics

This system has three types of user classes: Server user, Client User and Power User.
Server User:
-This user operates over the server domain, having the ability to initiate the presence of a server in a network.
-This server can send get requests to other machines with the network to create a client-server architecture.
-The user can set the server's name and password.
-The user can be involved in data transfer apart from the clients.
-The user can choose to shut the server down.

Client user:
-This user operates the client system, having the ability to act as a client in a given client-server network.
-The client can enter the server's name and associated and code word in order connect with the server.
-The client can send data files to be transferred across the system.

Power user:
-This user can be either a server user or a client user.
-If the user is a server, he/she can get access to all the logs, thus viewing real-time and past actions occurring in various systems present within the client-system architecture.
-If the user is a client, he/she can get access to all the logs, thus viewing real-time and past actions only within its system.

All access and data transfers/receiving is logged, in order to maintain a level of transparency, in order to prevent abuse of the system, and in order to hunt down any unauthorized users or hackers.

2.4 Operating Environment

Operating environment for the File Sharing system is as listed below.

-Client/server system
-Operating system: Windows and Linux
-32bit/64bit architecture
-Network Connected
-HTTPS (Hyper Text Transfer Protocol Secure)
-Platform: GUI rendered using Edge(Windows), gtk (Linux)

2.5 Design and Implementation Constraints

-Since system calls for different OS and architecture is unique. Therefore, it must be compiled and built in that specific OS and architecture.
-Some OS ignore requests from Broadcast IP by default, in such case, manual 'get requests' are required.
-The security design must be very intact to prevent hackers steal data from open ports.
-If the system is slow in responding or sending requests, the settings of the Server have to be changed to adhere to such systems, thus affecting the performance.
-In order to keep track of all changes made in the upcoming future, we plan to implement a subversion/source control system, mostly likely GitHub, where we pull/push code commits to/from the GitHub server. The source code, as well as the

current folder/file structure, will be able to be uploaded and fetched from our Github account.

2.6 User Documentation

The application will come with a “User Manual” tab, which will allow users to access the offline and online HTML. This will give the customer a clear view as to how the system works. It will also provide a step-by-step procedure as to how to handle the system. This document will describe the functionality of each subsection in detail.

2.7 Assumptions and Dependencies

-Device must be network configured.
-Server system should be an administrator.
-The need for an administrator to reduce firewall and opening of a port.
-The system is dependent on Golang webview library.
-It is assumed that the User will read the User Manual thoroughly and Agree to the Terms and Conditions after careful consideration.

3. External Interface Requirements

3.1 User Interfaces

-Frontend: HTML/CSS
-Backend: Golang

3.2 Hardware Interfaces

-A system with a NIC.
-A system that supports Edge(Windows) and gtk/cocao (Linux).
-A system with adequate storage.

3.3 Software Interfaces

-OS: We have chosen Windows/Linux operating systems for their support and user-friendliness.
-Golang: To implement the project and create a user-friendly interface, this language is chosen. They also provide an extended support to all packages, hence increasing the longevity of the project.
-HTML/CSS: To view the contents of the system, we use HTML.

3.4 Communication Interfaces

This project supports all types of web browsers. Communication interfaces will use HTTPS for data transmission.

4. System Features

4.1 Server System

4.1.1 Description and Priority

This module is the most important because of its core action of being able to facilitate the sharing of data, software and hardware resources. The server system's main role is to establish a secure connection to prevent hackers from exploiting the system. This is done by establishing its uniqueness by sending get requests across the network, thus identifying other devices, and its uniqueness. If it is not unique, the module identifies the presence of another server with similar name or a malicious user, thus sending an alert to the genuine user. This feature enhances the security of the system. After having established connection with different clients, the server acts a median for data communication, using HTTPS.

4.1.2 Functional Requirements

-The user should be able to create a server with a name and password.
-This server system should establish its uniqueness within the network by sending get requests across the network. If the server finds itself not unique, an alert is sent to the genuine user.
-The server system should be able to receive and connect with the client systems to transfer data. This can be done using HTTPS (Hyper Transfer Text Protocol Secure).
-The user should be able to shut the down the server system, thus shutting down the client-server network.

4.2 Client System

4.2.1 Description and Priority

The client system module acts as the two ends to the system. This module also focusses on security. This is achieved by making sure the client system is connected to a genuine server. This can be established by sending get requests to different devices and thus finding the suitable server system. This module is acts as a source for data transfer. In the presence of a server system, this module can send data files to another system having the same module. This is achieved only when modules of systems are connected to the server and takes place using HTTPS.

4.2.2 Functional Requirements

-The user should be able to create a client system.
-The user should be able to enter an appropriate server name and password in order to send files across that particular server.
-The user should be able to copy data files onto the system, which can be transferred to other client systems.
-This module should be able to send and receive data from the server. This can be done using HTTPS (Hyper Text Transfer Protocol Secure).

4.3 Data Transfer Module

4.3.1 Description and Priority

This module comprises of the data transfer across the two client systems governed by the server system. This secure transfer of data is achieved by using HTTPS (Hyper Text Transfer Protocol). The data upon encapsulating uses the TCP/IP model in Transport Layer in order to initiate the communication between the hardware and software interface. The module also comprises of shared clipboards.

HTTPS uses an encryption protocol to encrypt communications. The protocol is called Transport Layer Security (TLS), although formerly it was known as Secure Sockets Layer (SSL). This protocol secures communications by using what's known as an asymmetric public key infrastructure. This type of security system uses two different keys to encrypt communications between two parties:

1. The private key - this key is controlled by the owner of a website and it's kept, as the reader may have speculated, private. This key lives on a web server and is used to decrypt information encrypted by the public key.
2. The public key - this key is available to everyone who wants to interact with the server in a way that's secure. Information that's encrypted by the public key can only be decrypted by the private key.

4.3.2 Functional Requirements

-This module should be able to follow the protocols mentioned and transfer data from client to client using a server.

-The user should be able to access shared clipboards in order to ease transfer of files.

5. Other Non-Functional Requirements

5.1 Performance Requirements

The software should have high performance and low failure rates. The hardware and software should be able to transmit/receive data from different systems with high efficiency. Machines should have all recent Windows/Linux updates installed. Machines must have firewalls installed and active virus scanning software in usage. Data receiving/transmitting should be done using high security transmission.

5.2 Safety Requirements

The software must be used by the User using application configured based on the OS and architecture. This will provide high level of security and the User will get the best experience while using them. Also, the User must not share their private credentials which can be used to log into their networks. The User is strongly advised to go through the User Manual first.

5.3 Maintenance Requirements

The software must be made in such a manner that at a later point of time, software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment. This means that all new features must be easily incorporated to the pre-existing software system without hassle and providing discomfort to the User, through updates. The system must be able to handle a growing amount of work by adding resources to the system

5.4 Security Requirements

The software should have secure connection that prevents any form of malicious attacks on data transferred. This is achieved by using an appropriate protocol. The presence of a power user, where he/she can see the logs of the system enhances the security of the system. The identification of a unique server is also an important feature that comes along.

5.5 Reliability

The software must have 24 X 7 availability to the user, provided the user has a stable network connection. The software uses TCP to encapsulate the data, which further uses HTTPS in application layer to provide secure connection. Therefore, there is no loss of data, thus making it reliable.

Appendix A: Glossary

1. Broadcast IP - a network address used to transmit to all devices connected to a multiple-access communications network.

2. CSS - Cascading **S**tyle **S**heets - is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.

3.Microsoft Edge - a cross-platform web browser created and developed by Microsoft.

4. GitHub - is a provider of Internet hosting for software development and version control using Git.

5. Golang - Go is an open-source programming language that makes it easy to build simple, reliable, and efficient software.

6. GUI – **G**raphical **U**ser **I**nterface - a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation, instead of text-based user interfaces, typed command labels or text navigation.

7. GTK – GIMP **T**ool **K**it - free and open-source cross-platform widget toolkit for creating graphical user interfaces (GUIs).

8. HTML – Hyper **T**ext **M**arkup **L**anguage - is the standard markup language for documents designed to be displayed in a web browser.

9. HTTPS – Hyper **T**ext **T**ransfer **P**rotocol **S**ecure - an extension of the Hypertext Transfer Protocol. It is used for secure communication over a computer network, and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using Transport Layer Security or, formerly, Secure Sockets Layer.

10. IEEE – Institute of **E**lectrical and **E**lectronics **E**ngineers - a professional association for electronic engineering and electrical engineering.

11. NIC – Network **I**nterface **C**ard – is a computer hardware component that connects a computer to a computer network.

12. OS - Operating **S**ystem **-** system software that manages computer hardware, software resources, and provides common services for computer programs.

13. Protocol - a system of rules that allows two or more entities of a communications system to transmit information via any kind of variation of a physical quantity.

14. **SRS - S**oftware **R**equirements **S**pecification - is a description of a software system to be developed. It is modelled after business requirements specification, also known as a stakeholder requirements specification.

15. SSL/TLS - Secure **S**ockets **L**ayer/ **T**ransport **L**ayer **S**ecurity - protocols for establishing authenticated and encrypted links between networked computers.

16. TCP/IP - Transmission **C**ontrol **P**rotocol/ **I**nternet **P**rotocol - is the set of communications protocols used in the Internet and similar computer networks. The current foundational protocols in the suite are the Transmission Control Protocol and the Internet Protocol.