# Distributed File System

# Group 2 Phase 1

Software Requirements Specification

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| mm/dd/yyyy | 1.0 | Initial Version | Group 2 |
|  |  |  |  |
| 02/23/2022 | 1.3 | Rewrite for clarity | Bryan Graves |
|  |  |  |  |
| 4/28/2022 | 1.4 | Update from Implementation | Quang Nguyen |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Purpose [4](#__RefHeading___Toc19440719)

1.1. Scope [4](#__RefHeading___Toc19440720)

1.2. Definitions, Acronyms, Abbreviations [4](#__RefHeading___Toc19440721)

1.3. References [4](#__RefHeading___Toc19440722)

1.4. Overview [4](#__RefHeading___Toc19440723)

2. Overall Description [5](#__RefHeading___Toc19440724)

2.1. Product Perspective [5](#__RefHeading___Toc19440725)

2.2. Product Architecture [5](#__RefHeading___Toc19440726)

2.3. Product Functionality/Features [5](#__RefHeading___Toc19440727)

2.4. Constraints [5](#__RefHeading___Toc19440728)

2.5. Assumptions and Dependencies [5](#__RefHeading___Toc19440729)

3. Specific Requirements [6](#__RefHeading___Toc19440730)

3.1. Functional Requirements [6](#__RefHeading___Toc19440731)

3.2. External Interface Requirements [6](#__RefHeading___Toc19440736)

3.3. Internal Interface Requirements [7](#__RefHeading___Toc19440737)

4. Non-Functional Requirements [8](#__RefHeading___Toc19440738)

4.1. Security and Privacy Requirements [8](#__RefHeading___Toc19440739)

4.2. Environmental Requirements [8](#__RefHeading___Toc19440740)

4.3. Performance Requirements [8](#__RefHeading___Toc19440741)

# Purpose

This document outlines the requirements for the Distributed File System (DFS).

## Scope

This document will catalog the user, system, and hardware requirements for the DFS system. It will not, however, document how these requirements will be implemented.

The system will be only comprised of a server service running on a central hub and client software running on endpoints. The files will be stored on endpoints.

Users will be authenticated to the system itself, and users will be created manually. Only two roles will be in scope user and admin.

## Definitions, Acronyms, Abbreviations

Client – A service running on an endpoint

DFS – Distributed File System

GUI – A graphical User Interface, is a form of user interface that allows users to interact with the program.

Server – A central service that

## References

Use Case Specification Document

UML Use Case Diagrams Document

Class Diagrams

Sequence Diagrams

## Overview

The distributed file transfer system is designed to let users upload and request files, which are stored in a hidden folder on various endpoints. The system will verify id and password before giving users access to the files.

# Overall Description

## Product Perspective

The distributed file system is a cloud adjacent method of client-server file storage. It is a system used to self-contain company wide data; retaining data in house ensures privacy and security needs are met.

## Product Architecture

The system will be organized into 3 major modules: the server module, the client module, and the persistence module.

## Product Functionality/Features

Users log in to the distributed file system via username/password combination. Users view a list of files stored within system and can access any available file and/or upload new files. Interactions between user, system and files are logged. (see section 3 of this document for more detailed requirements that address these features):

## Constraints

Location of files will be unknown to the user. Files uploaded by a given user will be stored in an unknown, separate, arbitrary location(s) within the company.

## Assumptions and Dependencies

It is assumed that company hardware is standardized; all users will run identical operating systems on corresponding hardware.

The system will not keep additional backup copies of files other then what is listed in the requirements and/or use case specifications.

It is ok to store user credentials on the server .

# Specific Requirements

## Functional Requirements

### Common Requirements:

* + - 1. Users will be allowed to log in using their issued id and pin, both of which are alphanumeric strings between 6 and 20 characters in length.
      2. The system will provide brief help on each screen that describe the purpose of each function within the system.
      3. Users will be able to list available files
      4. Users will be able to retrieve files.
      5. Users will be able to upload files.
      6. Server will log every action and save a log for at least 30 days.

### Server Module Requirements:

* + - 1. Server will keep a log of which users pushed a request.
      2. Server will send a list of available files to client
      3. Server will send a requested file to client
      4. Server will record new files and where they are stored
      5. Server will authenticate users
      6. Server will allow creation of new users
      7. Server will differentiate between users and admin users

### Client Module Requirements:

* + - 1. Client able to push requests for files
      2. Client able to look at list of available files and choose desired file from list
      3. Client can upload their files through the server on their personal node
      4. Client is not able to modify files that are not their own
      5. Able to log in using id and pin to gain access to server
      6. Client code will be able to save and update their file
      7. There will be a GUI that connect to the Client code
      8. Multiple service module to make testing easy.

### Persistence Module Requirements:

* + - 1. Persist log to a file system.
      2. Persist database to file system
      3. Able to pull log.

## External Interface Requirements

* + 1. There must be an easy to follow and easy to understand GUI for users.
    2. GUI will be Java program.

## Internal Interface Requirements

* + 1. Every action that server process will be log into Persistence module, with date and time stamp.
    2. Any file type and any size will be accepted.
    3. Any file that is uploaded will need to be saved to 2 or more nodes.
    4. The log will be saved as a .txt file up to 30 days.

# Non-Functional Requirements

## Security and Privacy Requirements

* + 1. Administrator generated username.
    2. Local Network Only, static IP address.
    3. Authentication and Authorization must only be handled by the server
    4. All requests must be authenticated from the server.

## Environmental Requirements

* + 1. Program will be written and operate in Java.
    2. Systems in Java must use at least JDK 11
    3. Server must use environment variables to configure names and port.

## Performance Requirements