# CS 401 Group Project

Software Requirements Specification

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 02/12/2022 | 1.1 | Initial Version | Miles Vizinau, Moises, Isaiah |
| 02/16/2022 | 1.2 | Edited Text | Miles Vizinau, Isaiah |
| 02/18/2022 | 1.3 | Included Use Case Templates | Issiah |
| 02/19/2022 | 1.4 | Edited Text, Included some Reference Diagrams | Miles, Isaiah, Moises |
| 02/20/2022 | 1.5 | Included Sequence diagram | Miles |
| 02/26/2022 | 1.6 | Modified Class Diagram and Module Requirements | Miles |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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. It will not, however, document how these requirements will be implemented.

## Definitions, Acronyms, Abbreviations

**Client:** An application that's installed the node that can be used to communicate with the server software

**Distributed File System (DFS)**: The main purpose of the Distributed File System (DFS) is to allows users of physically distributed systems to share their data and resources by using a Common File System

**Node**: The physical hardware that software can be downloaded on, like a computer

**Server**: A computer or computer program which manages access to a centralized resource or service in a network

**User**: The physical person operating the hardware

## References

Use Case Specification Document

UML Use Case Diagrams Document

Class Diagrams

Sequence Diagrams

## Overview

The Distributed File System (DFS) is designed to act as a private storage system for a single company. As information leaking is an ever-present danger this system will ensure that only individuals associated with the company can access the file system

# Overall Description

## Product Perspective

## Product Architecture

The system will be organized into 5 major modules: the User module, the Node module, the Client module, the Server module, and the File module

## Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

* + - All files are hidden on users’ nodes
    - Any file type should be supported
    - Client software allows the user to talk to the server software
    - Server software keeps track of where files are located
    - Verified users should be allowed to access the DFS

## Constraints

* + - Employees have to request files from the server
    - Only employees can access the system (EX: using employee id & password)
    - Server doesn’t store files

## Assumptions and Dependencies

* + - All employees have a verified ID and password
    - The DFS software is installed only on a company computer

# Specific Requirements

## Functional Requirements

### Common Requirements:

* All users will be given an employee ID and allowed to create a password
  + Password should be between 6 & 20 characters in length
  + Password should include at least 1 number
  + Password should include at least 1 uppercase
  + Password should include at least 1 special
* Users will request files from the server software using the client software

### User Module Requirements:

* + - Users should have a User ID and password
    - Users should have a way to tell if they’re also a supervisor
    - Users should have a way to tell if they’re currently using any client software
    - Users should only be able to have open 1 piece of client software at a time

### File Module Requirements:

* + - Must contain file size, name, type, and path
    - Must have a credential flag to know if the user has access to the requested file

### Client Module Requirements:

* + - Should provide an interface for user to log in and out
    - Should be able to open and close the server
      * Server should only close if there’s 0 clients logged in
    - Should be able to send requests to server software for file operations

**3.1.5. Server(s) Module Requirements:**

* + - Must have an event history
    - Must have a list of accessible files
    - Must have a list of client software currently accessing the server
    - Must be able to handle multiple clients at a time

**3.1.6. Node Module Requirements:**

* + - Must have a name
    - Must have a storage size
    - Must have a storage path
    - Only 1 user can be active on a node (computer) at a time

## External Interface Requirements

* + - The user must have direct access with the client software in order to have the server manage the files
    - The client software must be able to recognize users and non-users (A login system)

## Internal Interface Requirements

* + - The system must process data from the user to a node (a computer); which contains the client software. The client software must send a request to the server file system and send the data back to the user

# Non-Functional Requirements

## Security and Privacy Requirement

* + - Users with supervisor status should be given certain privileges
      * Access to event history of the DFS
      * Access to the complete file system

## Environmental Requirements

* Every computer the company issues will have access to the same files using the DFS software
* The system will utilize the Java programming language.

## Performance Requirements

**Use Case Specification (Description) Template**

Use Case ID: *{0}*

Use Case Name: *{File Upload}*

Relevant Requirements: \* *{3.1.1}*

Primary Actor: *{DFS Client}*

Pre-conditions: *{User must log in with valid ID and the file to upload must exist.}*

Post-conditions: *{If file does not already exist, file is uploaded to the DFS}*

Basic Flow or Main Scenario: *{Numbered flow of events: 1 The user logs in with ID, 2 The user requests to upload a file, 3 The user selects the file to upload, 4 The client uploads the file to the DFS*

Extensions or Alternate Flows: *{Alternatively, if the file already exists, prompt the user to choose whether they want to replace the identical file name or skip this action.}*

Exceptions: *{File no longer exists or cannot be found.}*

Related Use Cases: {File request}

**Use Case Specification (Description) Template**

Use Case ID: *{0}*

Use Case Name: *{File Request}*

Relevant Requirements: \* *{3.1.1}*

Primary Actor: *{DFS Client}*

Pre-conditions: *{User must log in with valid ID.}*

Post-conditions: *{If the file requested exists, the client will request it from the server and it will be returned to the user.}*

Basic Flow or Main Scenario: *{Numbered flow of events: 1 The user logs in with their ID, 2 The user requests to retrieve a file, 3 The client checks with the server if the file exists, 4 If it does, file is returned to the user who requested it.}*

Extensions or Alternate Flows: *{Alternatively, if the file does not exist, then inform the user.}*

Exceptions: *{Error: file does not exist.}*

Related Use Cases: {File upload}

**Use Case Specification (Description) Template**

Use Case ID: *{1}*

Use Case Name: *{Supervisor privileges}*

Relevant Requirements: \* *{4.1}*

Primary Actor: *{User/Supervisor}*

Pre-conditions: *{Supervisor must enter valid supervisor ID}*

Post-conditions: *{If ID is valid, the user will be granted access to the event history of the DFS as well as the entire file system.}*

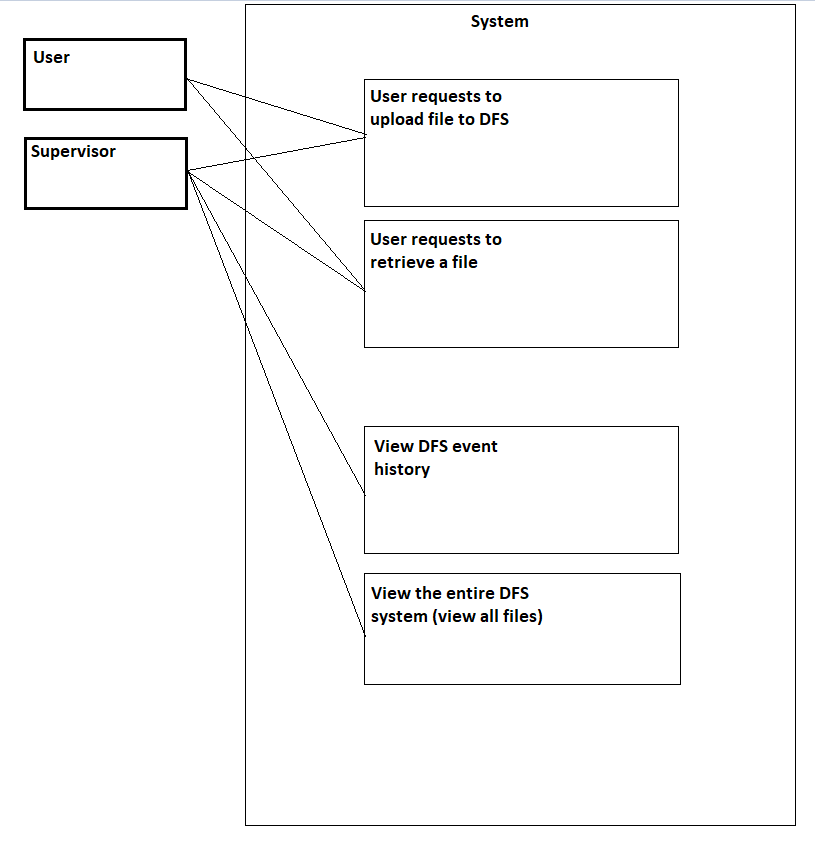
Basic Flow or Main Scenario: *{Numbered flow of events: 1 The user logs in with their supervisor ID, 2 The user requests access to the event history or to the entire file system, 3 Access granted}*

Extensions or Alternate Flows: *{N/A}*

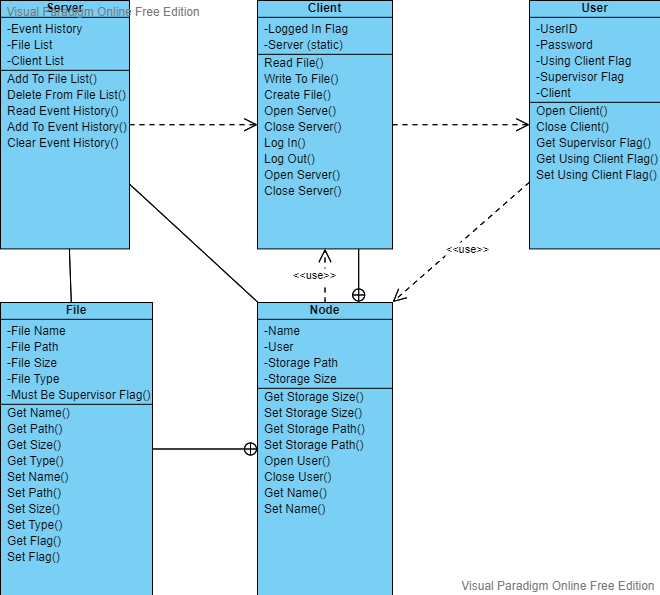
Exceptions: *{Invalid supervisor ID}*

Related Use Cases: {N/A}

**Use Case Specification Diagram(s)**



**Class Diagram(s)**



**Sequence Diagram(s)**

