## Table of Contents

[**Table of Contents 1**](#_ltjxsms6shbp)

[**Introduction 2**](#_xw8eczjcevxh)

[**Requirements and Implementation 3**](#_47orvo2ephdy)

[**Files Storage 3**](#_q8mx2o3sw608)

[**Errors/Warning Monitor 3**](#_2j06m42c3vkm)

[**File Sharing 3**](#_kq1sbje4tur4)

[**Tools and Framework: 3**](#_k7u141qjxt11)

[**Use Cases 4**](#_b1r1nmqmaxzc)

[**Error Handling and User Interface 5**](#_rolx9xi4t52r)

[**Conclusion 5**](#_wt5cxce9tbry)

[**References: 7**](#_5wuenvriw50r)

## 

## Introduction

The project's focus is to develop a cloud-based file server that can be accessed through a desktop GUI application. As more people store their files in cloud services like Google Drive, Microsoft OneDrive, and Dropbox, a solution is needed to access these files across various devices and operating systems, such as Android and iOS mobile phones and Windows, Mac, and Linux PCs.

User management is a key part of this project, allowing users to create, delete, update, log in, and log out of their accounts. This ensures that each user can have a personalized experience.

The project also includes comprehensive file management features. Users can create new files, upload existing ones, delete files, rename them, and move or copy files as needed. These features are designed to make it easy for users to manage their files in the cloud, just as they would on their personal computers.

Storing user files in the same container that hosts the application is another important aspect of the project. This approach helps to keep files organized and accessible.

A system to monitor errors and warnings has been included in the project to help users understand if anything goes wrong. This promotes trust and transparency in the application.

File sharing is a feature that allows users to share their files with others. They can set permissions for reading and writing, making collaboration simple and controlled.

The project was developed using specific tools and frameworks. Docker containers were used for the environment, and Netbeans was chosen as the Java IDE. Scene Builder was used for designing GUIs, and JavaFX (Point) was the library used for creating them.

The project was completed using Java, with Maven for compiling. Care was taken to meet all the requirements, such as using error handling where needed and providing a simple UI that allows users to interact with the system.

## Requirements and Implementation

***User Management:***

Create: Enables users to register.

Delete: Allows deletion of user accounts.

Update: Facilitates updates to user information.

Login/Logout: Ensures secure authentication.

***File Management:***

Create New File: Users can add content and save.

Upload/Download Files: Seamless uploading and downloading from the hosting container.

Delete/Rename/Move/Copy Files: Full control over file operations.

## Files Storage

User files are stored in the application's hosting container.

## Errors/Warning Monitor

Real-time monitoring and display of errors and warnings.

## File Sharing

Share files with read/write permissions.

## Tools and Framework:

Environment: Docker containers are used for environmental consistency.

Java IDE: NetBeans, coupled with Maven for compilation.

GUI Design: Scene Builder for interface design.

Java GUI Library: JavaFX for GUI development.

## Use Cases

**Use Case: User Registration**

***Actor***: New User

***Precondition***: User must not have an existing account

***Flow***:

- User selects the "Create Account" option.

- User enters required details (e.g., username, email, password).

- System validates the information and confirms the registration.

- User is redirected to the login page.

***Postcondition***: User has successfully created an account and can log in.

**Use Case: File Upload**

***Actor***: Registered User

***Precondition***: User must be logged in

***Flow***:

- User selects the "Upload File" option.

- User browses and selects a file from their local system.

- System uploads the file to the cloud container and confirms the upload.

***Postcondition***: The file is successfully uploaded and stored in the user's cloud container.

**Use Case: File Sharing with Permissions**

***Actor***: Registered User

***Precondition***: User must be logged in and have files to share

***Flow***:

- User selects a file and chooses the "Share File" option.

- User enters the email of the other user and sets permissions (read/write).

- System sends a sharing notification to the recipient.

***Postcondition***: File is shared with another user with the specified permissions.

**Use Case: Error Monitoring**

***Actor***: Registered User

***Precondition***: User must be logged in

***Flow***:

- User encounters an error or warning (e.g., failed upload, deletion).

- System logs the error and displays it in the user's error/warning monitor.

- User can view detailed information about the error.

***Postcondition***: User is informed of the error and its details.

**Use Case: Renaming a File**

***Actor***: Registered User

***Precondition***: User must be logged in and have files to rename

***Flow:***

- User selects a file and chooses the "Rename File" option.

- User enters a new name for the file.

- System updates the file with the new name and confirms the change.

***Postcondition***: The file is successfully renamed in the user's cloud container.

## Error Handling and User Interface

The application embodies robust error handling to ensure a smooth user experience. The UI is designed to be intuitive and interactive, fostering user engagement.

## Conclusion

This project is a notable achievement in cloud-based file management. (Learning) It has carefully addressed vital functions such as user and file management, error monitoring, and file sharing. By using up-to-date tools like Java IDE: Netbeans, Scene Builder, JavaFX, and the required Docker containers (Venti), the project meets current industry standards and looks ahead to future developments. (Bigelow)

The project's success in creating a cloud-based file server highlights a deep understanding of the connection between technology and user requirements. By following professional standards and using advanced technologies, the project has fulfilled its goals and positioned itself as an essential part of the growing technological field.

In conclusion, this final report sums up the project from start to finish. The focus on innovation, efficiency, standards, and user-centered design is central to this work. It results in a product that not only reflects current practices but also points the way for future growth in cloud-based file management.

## References:

Bigelow, Stephen J. “What is Docker and How Does It Work?” *TechTarget*, https://www.techtarget.com/searchitoperations/definition/Docker. Accessed 14 August 2023.

Learning, Great. “What is File Handling in Java?” *Great Learning*, https://www.mygreatlearning.com/blog/what-is-file-handling-in-java/. Accessed 18 August 2023.

Point, Java. “JavaFX Tutorial - javatpoint.” *Javatpoint*, https://www.javatpoint.com/javafx-tutorial. Accessed 14 August 2023.

Venti, Cameron. “What is Docker?” *Opensource.com*, https://opensource.com/resources/what-docker. Accessed 14 August 2023.