LINUX

PROJECT REPORT

On

User Authentication System:

Submitted by

Bharath Krishna (24MCA2018 3)

in partial fulfillment for the award of the degree of

Masters of Computer Applications Chandigarh University

Aug 2024 – Nov 2024

Table of Contents

File Organizer in Linux

1. Introduction

1.1 Background

File management is essential in maintaining an organized workspace, especially in Linux environments where a large number of files may accumulate over time. Properly organizing files helps users find and manage them more efficiently, especially in shared or development-focused environments. Manually organizing files by type and extension can be a tedious task, which is why automation through scripting offers a streamlined solution.

1.2 Project Overview

The objective of this project is to develop a Bash script that automatically organizes files within a specified directory on a Linux system. This script categorizes files based on their extensions, moving them into designated subfolders. By leveraging the simplicity and power of Bash, the solution is lightweight, efficient, and compatible with most Linux distributions.

2. Project Objectives and Scope

2.1 Objectives

The primary goals of this project include:

- Automating the organization of files within a specific directory by their extensions.
- Creating a clear folder structure, where each folder represents a specific file type (e.g., "txt" for text files, "jpg" for images).
- Enhancing productivity by reducing clutter and enabling easy access to files based on their type.

2.2 Scope

This script is tailored for Linux environments, with specific testing on CentOS. However, it should work across most Unix-like systems with minor or no modifications. The script will:

1. Recognize a variety of file extensions.
2. Create subdirectories based on file types as needed.
3. Move each file to the appropriate subdirectory based on its extension.

3. System Requirements

3.1 Hardware Requirements

- A computer running a Linux-based OS with the Bash shell installed.
- Storage space to create directories and organize files.

3.2 Software Requirements

- **Operating System**: CentOS or any Linux distribution.
- **Bash**: This script uses Bash commands for execution, available by default on most Linux distributions.

4. Script Components and Code Explanation

4.1 Script Overview

This script is designed to:

1. Identify all files in a specified directory.
2. Check each file's extension.
3. Create a folder for each unique extension (if not already existing).
4. Move each file into the corresponding extension-based folder.

Here's the complete code:

```
!/bin/bash

# Define the source directory
SRC_DIR="./sample_files"

# Loop through all files in the source directory
for file in "$SRC_DIR"/*; do
    if [ -f "$file" ]; then
        # Get the file extension
        EXT="${file##*.}"

        # Create a directory for the extension if it doesn't exist
        mkdir -p "$SRC_DIR/$EXT"

        # Move the file into the corresponding directory
        mv "$file" "$SRC_DIR/$EXT/"
    fi
done

echo "Files organized by extension."
```

4.2 Explanation of Each Code Section

Defining the Source Directory

```
SRC_DIR="./sample_files"
```

This line specifies the directory containing files to be organized. Replace `"./sample_files"` with any desired directory path.

Looping Through Files

```
for file in "$SRC_DIR"/*; do
    if [ -f "$file" ]; then
```

This loop iterates through each item in the specified directory. The `if [ -f "$file" ];` condition checks if the item is a file, so the script only processes files and skips subdirectories.

Extracting File Extension

```
EXT="${file##*.}"
```

This line extracts the file extension by removing everything up to the last period in the filename.

Creating Subdirectories by Extension

```
mkdir -p "$SRC_DIR/$EXT"
```

The `mkdir -p` command creates a new directory named after the file extension if it does not already exist.

Moving Files to Corresponding Folders

```
mv "$file" "$SRC_DIR/$EXT/"
```

This `mv` command moves the file into its designated folder based on its extension.

5. Workflow and Execution

5.1 Execution Steps

1. Place the script in the root directory where files are located.
2. Make the script executable by running:

```
chmod +x file_organizer.sh
```

3. Run the script:

```
./file_organizer.sh
```

5.2 Example Directory Structure

If the source directory (`sample_files`) contains the files `report.pdf`, `photo.jpg`, and `document.txt`, running the script will create subfolders `pdf`, `jpg`, and `txt` and move each file to its respective folder.

6. Testing and Verification

6.1 Test Cases

1. **Basic Test**: Populate `sample_files` with various file types (e.g., `.txt`, `.jpg`, `.pdf`). Run the script and check that each file is moved to the appropriate subdirectory.
2. **Unsupported Extensions**: Add files with extensions not specified in the script, such as `.xyz`. These files should also be moved into corresponding folders (e.g., an `xyz` folder).
3. **Empty Directory**: Run the script in an empty directory to ensure it completes without errors.

6.2 Observed Results

The script successfully categorized and moved files by extension as expected, creating new folders for each unique extension.

7. Error Handling and Troubleshooting

7.1 Common Errors and Fixes

- **Permission Denied**: If permission is denied, use `chmod` to make the script executable, or run with `sudo` if necessary.
- **Invalid Path**: If the specified `SRC_DIR` path is incorrect, ensure the directory exists and adjust the path accordingly.
- **Handling Edge Cases**: Files without an extension may cause unexpected behavior. Consider adding logic to handle these cases by moving them into a generic "No Extension" folder.

7.2 Enhancing Error Handling

To improve reliability, we can modify the script to check for:

- Files without extensions.
- Existing files with the same name in the destination folder.
- Permissions before moving files.

8. Future Improvements

8.1 Recursive Sorting

Adding recursive file traversal would allow the script to sort files within subdirectories.

8.2 Additional Error Handling

Improving error handling can prevent the script from crashing due to unexpected inputs or file types. For instance:

- Adding a check for files with no extensions.
- Handling symbolic links and avoiding infinite loops.

8.3 Logging

Incorporating logging mechanisms can track file movements and errors, helping users troubleshoot issues. For example:

```
echo "$(date): Moved $file to $SRC_DIR/$EXT" >> file_organizer.log
```

8.4 Interactive User Input

A future improvement could involve prompting users for input, allowing them to specify which file types to sort or where the files should be moved.

9. Conclusion

9.1 Summary

This project successfully demonstrates a basic yet effective way of organizing files by extension in a Linux environment using a Bash script. This automation reduces manual effort, maintains an organized directory structure, and enhances user productivity.

9.2 Practical Applications

The file organizer script is particularly useful for professionals working in environments where data is frequently updated, such as software development, research, or creative fields with diverse file types. By categorizing files based on type, users can quickly locate, manage, and manipulate files.

9.3 Future Enhancements

To make this project more robust and user-friendly, future updates could include additional customization options, error-handling mechanisms, and logging features to provide better insight into the organization process.

10. Appendix

10.1 Code Listing

The following code listing includes the complete script used in this project:

```
!/bin/bash

SRC_DIR="./sample_files"
for file in "$SRC_DIR"/*; do
    if [ -f "$file" ]; then
        EXT="${file##*.}"

        mkdir -p "$SRC_DIR/$EXT"

        mv "$file" "$SRC_DIR/$EXT/"
    fi
done

echo "Files organized by extension."
```

10.2 References

- Linux Documentation Project: https://tldp.org/
- GNU Core Utilities Manual: https://www.gnu.org/software/coreutils/manual/
- Bash Scripting Guide: https://www.gnu.org/software/bash/manual/