PROJECT

Tittle: Word Replacement in Linux

Table of Contents

Introduction

In system administration, managing and processing text files efficiently is essential for tasks like configuration management and log analysis. This project aims to create a Linux-based utility to search and replace specified words across multiple text files within a directory. By automating this task, we simplify word substitution processes, improve consistency across files, and reduce manual work for repetitive tasks.

Objectives

- **Primary Objective**: Develop a command-line script to search for a word in files and replace it with a specified alternative.
- **Additional Goals**:

- Make the script user-friendly, allowing users to input search terms, replacement terms, and directories interactively.
- Ensure file backups are created to prevent accidental data loss.
- Log the operations for easy reference and troubleshooting.

1. **Operating System**: Fedora Linux (or any compatible Linux environment)
2. **Tools and Commands**:
   a. `nano` or `vim` for script editing
   b. Basic Linux commands: `grep`, `sed`, `echo`
   c. `chmod` for setting script execution permissions
3. **Directory Structure**:
   a. Main project folder to store the script and a sample directory with text files.

Implementation Steps

Step 1: Setup and File Creation

1. Created a project directory `search_replace_project` and a subdirectory `sample_texts`.
2. Created sample text files in `sample_texts` containing the target word for testing.

Example of a sample file (file1.txt):

```
arduino
Copy code
```

```
This is a test file containing the word Linux.
```

Step 2: Script Creation

The script, `search_replace.sh`, was developed to:

- Take user input for the search term, replacement term, and directory path.
- Verify the existence of the directory specified.
- Search for files containing the specified word.
- Confirm with the user before replacing occurrences of the word.
- Log details of modified files.

Project Requirements Operating System:

Fedora Linux (or any compatible Linux environment) Tools and Commands: nano or vim for script editing Basic Linux commands: grep, sed, echo chmod for setting script execution permissions Directory Structure: Main project folder to store the script and a sample directory with text files.

Script Code:

```
bash
Copy code
#!/bin/bash
# Script to search and replace words in text files

echo "Enter the word to search for:"
read search_term

echo "Enter the word to replace it with:"
read replacement_term

echo "Enter the directory path containing the files:"
read directory_path
```

```
if [ ! -d "$directory_path" ]; then
    echo "Directory does not exist. Please check the
path and try again."
    exit 1
fi

matching_files=$(grep -rl "$search_term"
"$directory_path")

if [ -z "$matching_files" ]; then
    echo "No files contain the word '$search_term'."
    exit 0
else
    echo "Files containing '$search_term':"
    echo "$matching_files"
fi

echo "Do you want to replace '$search_term' with
'$replacement_term' in the above files? (y/n)"
read confirmation
if [ "$confirmation" != "y" ]; then
    echo "Replacement canceled."
    exit 0
fi

for file in $matching_files; do
    sed -i.bak "s/$search_term/$replacement_term/g"
"$file"
    echo "Replaced in $file"
```

```
done

echo "$(date): Replaced '$search_term' with
'$replacement_term' in files:" >> replacement_log.txt
echo "$matching_files" >> replacement_log.txt
echo "Changes logged in replacement_log.txt"
```

Step 3: Execution

- Set executable permissions:

```
bash
Copy code
chmod +x search_replace.sh
```

- Executed the script and provided input when prompted:
  - Search term: `Linux`
  - Replacement term: `Fedora`
  - Directory path: `~/search_replace_project/sample_texts`

Step 4: Verification and Logging

- Verified replacements by checking the modified files.
- Checked `replacement_log.txt` to confirm that the operation details were recorded.

Results and Observations

The script successfully performed search and replace operations:

- All files containing the specified word were located and listed.
- Each occurrence of the search term was replaced with the replacement term, and `.bak` backup files were created.
- A log file recorded the date, search term, replacement term, and modified files.

6. Limitations and Future Improvements

- **Limitations**:
  - The script performs a global replacement on each line without user confirmation for each occurrence within a file.
  - The current version handles text files only.
- **Future Improvements**:
  - Add regex support for advanced search patterns.
  - Provide an interactive confirmation for each replacement within a file.
  - Extend functionality to include file types other than `.txt`.

Conclusion

This project demonstrates the power and flexibility of Linux command-line tools for automating repetitive tasks like word replacement across files. The `grep` and `sed` commands proved essential, and the logging and backup features added reliability. This project serves as a foundation for more complex text processing and administrative scripts in Linux environments.

APPENDIX

```
~/search_replace_project
├── search_replace.sh
├── replacement_log.txt
└── sample_texts
    ├── file1.txt
    ├── file2.txt
    └── file3.txt
```