Intelligent File Management System: Design and Implementation of File Organizer Pro

Operating Systems Project Report

Suyash Parmar, Harsh Gupta, and Piyush Kaushal
Rishihood University, Delhi NCR, India
Course: Operating Systems
February 2026

Abstract— **The exponential growth of digital files on personal computers has made manual file organization a tedious and error-prone process. This paper presents the design and implementation of "File Organizer Pro," a modern, web-based intelligent file management system [1]. The application leverages a Flask-based Python backend coupled with a responsive HTML/CSS/JavaScript frontend to provide users with an intuitive interface for managing local storage [1]. Key features include automated file sorting, duplicate file detection using SHA-256 hashing, empty directory cleanup, and a recursive local search engine [1].**

I. INTRODUCTION

With the increasing reliance on computers for both professional and personal tasks, users accumulate vast amounts of digital data [1]. Disorganized file systems lead to decreased productivity, wasted storage space, and frustration [1]. Traditional desktop utilities often feature outdated user interfaces or require complex command-line interactions [1]. This project introduces File Organizer Pro, an automated system engineered to simplify file management by bridging robust Python automation scripts with a modern web interface [1].

II. SYSTEM ARCHITECTURE

The architecture follows a client-server model deployed locally on the user's machine [1].

A. Backend Server

The backend is powered by Flask, a lightweight Python web framework [1]. It handles core logic and interfaces with the host operating system using standard Python libraries such as *os*, *shutil*, and *hashlib* [1]. The backend exposes several RESTful API endpoints:

- **/api/organize**: Sorts files within a target directory based on user-defined rules [1].
- **/api/clean/duplicates**: Identifies and removes duplicate files using SHA-256 cryptographic hashing [1].
- **/api/clean/empty_folders**: Recursively finds and deletes empty directories [1].
- **/api/search**: Performs high-speed recursive local search using *os.walk* [1].

B. Frontend Interface

The frontend is constructed using HTML5, CSS3, and JavaScript (ES6) [1]. It utilizes a "glassmorphism" design aesthetic and an animated orb background [1]. JavaScript's fetch API is utilized to communicate asynchronously with the Flask backend, while a polling mechanism queries the */api/logs* endpoint every second to simulate a live terminal output [1].

III. IMPLEMENTATION DETAILS

A. Organization Algorithms

The core organization logic allows users to group files by File Type or Date (e.g., 2023/10-October/) [1]. The application maintains a *history.json* ledger, recording original and new paths of every moved file to provide a robust "Undo" mechanism [1].

B. Storage Analytics

A dynamic storage widget displays root disk usage [1]. It retrieves real-time statistics using *shutil.disk_usage* and renders an animated SVG semicircle gauge via JavaScript [1].

IV. RESULTS AND DISCUSSION

A. Usability

Migrating from a Tkinter-based desktop GUI to a Flask-driven web app dramatically improved the user experience [1]. The modern interface is visually appealing and runs seamlessly in any modern browser on localhost [1].

B. Performance

Operations such as file sorting and empty folder removal execute in linear time:

$$T(n) = O(N) \tag{1}$$

where N is the number of files or directories [1]. The duplicate removal script uses a size-first grouping strategy to minimize expensive file hashing, ensuring efficient cryptographic analysis [1].

V. CONCLUSIONS

File Organizer Pro successfully addresses digital clutter by automating common file management tasks [1]. Through careful architectural planning, the system combines Python's computational power with the aesthetic versatility of modern web technologies [1]. Future work could include scheduled background tasks and cloud storage integration [1].

ACKNOWLEDGMENT

Developed as a comprehensive solution for local directory management and system optimization [1].

REFERENCES

[1] S. Parmar, P. Kaushal, and H. Gupta, "Intelligent File Management System: Design and Implementation of File Organizer Pro," Project Report, 2026.

[2] Pallets Projects, "Flask Web Framework Documentation," 2024.

[3] Python Software Foundation, "The Python Standard Library," 2024.