

Project Title:

Advanced File Organizer Bot

Problem Statement:

Modern users often deal with cluttered download folders or directories full of random files. Manually organizing them by file type, size, or extension is time-consuming. This project addresses this problem by providing a real-time automated file organizer that monitors a folder and instantly sorts new files as they arrive.







Project Overview:

This Python-based desktop application allows users to select a folder and automatically organize files based on:

- File extension (e.g., .jpg, .pdf)
- File type (e.g., image, document, video)
- File size (small, medium, large)

It also provides real-time monitoring using the Watchdog library, meaning new files are sorted instantly as they are added. The tool comes with a Tkinter-based GUI, allowing users to start organizing without typing a single command.

Key Features:

Feature	Description
 Folder Selection	User selects any folder via a GUI
 Sort by Extension	Sorts files into folders like JPG_Files, PDF_Files, etc.
 Sort by File Type	Categorizes files as Image, Video, Document, or Other
 Sort by File Size	Categorizes files as Small, Medium, or Large
 Real-Time Monitoring	Uses Watchdog to monitor folder and organize new files as they appear
 GUI Interface (Tkinter)	Easy-to-use interface with checkboxes and buttons

Feature	Description
⚠ Error Handling	Skips folders or files that cause errors and prints the issue

Technologies Used:

Technology Purpose

Python 3 Core logic and programming language

Tkinter Building the graphical user interface (GUI)

Watchdog Real-time monitoring of file system changes

shutil, os File operations like moving and creating directories

filedialog GUI dialog for folder selection

Core Logic:

- **Organizing Strategy:**
 - By extension: Groups files like pdf, jpg, etc.
 - By type: Uses predefined extensions to classify into Image, Video, Document.
 - By size: Groups as Small (<1MB), Medium (1–10MB), Large (>10MB)
 - **Live Monitoring:**
 - Uses Observer and FileSystemEventHandler from watchdog to listen for new files.
 - When a new file is created, it is instantly categorized based on selected options.
 - **GUI Flow:**
 - Users choose folder → select sorting preferences → click "Start Monitoring"
 - Once active, the tool silently organizes incoming files.
-

What You Learned:

- Building GUI applications with checkboxes, buttons, and dialogs using Tkinter

- Implementing event-driven file system monitoring using Watchdog
 - Performing file operations (move, rename, create folders) using os and shutil
 - Structuring Python applications with clear logic separation
 - Handling file type inference, extension parsing, and robust exception handling
 - Delivering a real-world productivity automation tool
-

How to Explain It in an Interview (1–2 min):

“I created a desktop application in Python that automates file organization using a real-time monitoring system. The user selects a folder and chooses how they want to organize files — by extension, type, or size. The tool immediately sorts existing files and then continuously watches the folder using the Watchdog library. Whenever a new file is added, it’s automatically moved to the correct folder. I built the GUI using Tkinter with checkboxes and buttons for customization. This project helped me understand OS-level automation, live file system events, and user interface design in Python.”