**INTERNSHIP PROJECT DOCUMENT**

**ON**

**FILE ORGANIZER**

Submitted by

(Intern)

**P.Katyayani Sai**

Submitted to

**Founder- Kanduri Abhinay**

**CTO - Saniya Begum**

**PROJECT-1**

**Title:Fibonacci Sequence Generator**

**Project Description:**

The Automated File Organizer Web Application is a Python-based solution that allows users to effortlessly organize their files via a simple web interface. Leveraging the Flask framework for web functionality and OpenCV for optional image processing, this project enables users to upload files to a server, where they are automatically sorted into folders based on their file types (e.g., images, videos, documents, audio, archives, and other unrecognized types).

**Packages and Libraries Used:**

**1.Flask**: Flask is a lightweight web framework used to create the web interface, handle HTTP requests, and manage file uploads.

**2.OpenCV (cv2)**: OpenCV is an open-source computer vision library. In this project, it’s used optionally for image processing tasks like resizing uploaded images before organizing them.

**3.OS:** This standard Python library provides functions to interact with the operating system. It's used for directory and file handling tasks, such as creating folders, listing files, and getting file paths.

**4.shutil**: A Python standard library module used for high-level file operations, such as moving files from the uploads directory to the appropriate organized folder.

**Tools used:**

**Jupyter Notebook:**

An interactive environment where code is written, executed, and documented, often used for exploratory data analysis and model development.

**Python**:

**Python 3.x**: The primary tool for writing and executing the code. Python’s simplicity makes it ideal for implementing algorithms like the Fibonacci sequence.

### ****Flask****: Flask is the primary web framework used to set up a minimal server, create routes for file uploads, and manage user interaction.

### ****OpenCV (cv2)****: Used for optional image processing, such as resizing images before organizing them into folders.

### ****HTML****: Used to create a simple file upload form in the web interface.

### **Algorithm Used:**

### ****File Type Categorization Algorithm**** : Automatically sorts files into predefined categories based on file extensions.

### ****Image Processing Algorithm (using OpenCV):****

* **Purpose**: Prepares images for storage by resizing them to a standard resolution (800x800 pixels).

### ****File Movement Algorithm:****

* **Purpose**: Transfers files from the uploads folder to categorized folders within the organized directory.

### ****Web Request Handling and Response Algorithm (Flask):****

* **Purpose**: Manages HTTP requests and responses, enabling users to interact with the file organizer via a web interface.

### ****Directory Management Algorithm:****

* **Purpose**: Ensures all necessary folders are created dynamically as files are organized.

**Software Used:**

* **Python**
* **Jupyter Notebook/IDE:**

**Jupyter Notebook:** An interactive development environment often used for data exploration, visualization, and iterative development. It allows for running code in cells and integrating markdown documentation.

**What I Have Analyzed from this project?**

I focused on several key components that contribute to its efficient workflow for automated file sorting and organization. The project utilizes a file type categorization algorithm that effectively groups files based on their extensions, using a predefined dictionary to ensure structured organization across different categories such as images, videos, documents, and more. Additionally, the optional integration of OpenCV for image processing allows for the resizing of images before storage, standardizing their dimensions and enhancing organization.

**Advanced Steps or Future Scope:**

### 1. ****Advanced File Categorization**** : Implement machine learning algorithms to improve file categorization accuracy. By training models on user-uploaded files, the application can learn and adapt to user preferences and better classify files that may not fit standard categories.

### **2**.****User Authentication and Management**** : Introduce user authentication features to allow multiple users to maintain their own file organization preferences. This could include user profiles, saved settings, and history of uploaded files.

### **3**. ****Enhanced File Processing**** : Allow users to process large batches of files with more advanced options, such as applying specific rules or filters before categorization.

### **4. **Integration with Cloud Storage :** Cloud Syncing**: Enable users to sync their organized files with cloud storage solutions (e.g., Google Drive, Dropbox) for easier access and backup.

### ****5. Search and Filter Functionality**** : Implement search features that allow users to find files based on metadata (e.g., creation date, file size) or even content (for text documents).

### **6.**User Interface Improvements**** : Enhance the web interface with a responsive design to ensure compatibility across devices, including smartphones and tablets.

### 7. ****Backup and Recovery Options**** : Implement features for automatic backups of the organized directories to prevent data loss.

**Source:**

* Research papers and articles
* Chatgpt
* Youtube
* **Python Official Documentation**