****

**De La Salle University- Manila**

**Gokongwei College of Engineering**

LBYCPA1

Programming Logic and Design Laboratory

Project Proposal

<File Organizer>

<Curt Ethan C. Lim>

<Lance Altheus A. Obispo>

<Gabrielle D. Simbajon>

**Project Description**

Overview

The main objective of this project is to create a program that can automatically organize files in a directory. As time passes, the files in a directory can pile up and become disorganized, making it hard to find specific files. The "File Organizer" program aims to make this process easier by recognizing different file types in the directory based on their file extensions, such as .mp3, .docx, .pdf, .png, etc. The program will then create separate folders for each file type and move the corresponding files to their appropriate folders.

The organization of files based on their file types simplifies the process of searching for specific files, saving users time and effort. Moreover, the program can potentially boost system performance by reducing the operating system’s file system load. The program will allow users to sort files based on various criteria, such as the file's modified date, size, or name. It also allows filtering specific file types from being sorted, such as temporary or system files.

The "File Organizer" program offers a useful tool for managing files. It can help streamline the process of locating files, especially for users with large numbers of files in their directories. Additionally, the program provides users with the ability to customize the organization process, giving them greater control over their file management.

Statement of the Problem

As time passes, users accumulate a lot of files in their directory, making it difficult to locate specific files. Organizing these files manually into folders based on their file type or extension can be an inconvenience, resulting in a cluttered directory. This can make it difficult for users to find the files quickly and efficiently they need. The main objective of this project is to develop a program that will automatically sort and organize files into folders based on their file type or extension. This will simplify the process of finding and accessing files, saving users time in manually organizing their files.

Technical Objectives

1. Develop a program that organizes files in a directory based on their file type or file extension.
2. Create a user-friendly interface that allows users to select the directory to be sorted and specify any filtering options.
3. Implement error handling to handle common exceptions that may occur during the file organization process.
4. Ensure the program is cross-platform compatible and can run on different operating systems.

Implementation

* **Requirements Gathering**: The first step in developing the file sorting and organizing program is to gather the requirements from the stakeholders or end-users. This includes identifying the file types that need to be sorted, selecting the target directory to be organized, and deciding on any filtering options that should be incorporated. (Optional)
* **Design**: After gathering the requirements, the program's design must be developed. This involves selecting the programming language and libraries to be used, designing the user interface, and determining the data structures and algorithms that will be utilized to sort and organize files. (Python)
* **Development**: The development phase involves writing the code for the program based on the design. This includes building the user interface, reading files in the target directory, sorting them based on their file type or extension, creating folders for each file type, and moving files to their respective folders.
* **Testing**: Once the program has been developed, it must be tested to ensure it is correct, accurate, and performs well. This includes testing the program on various operating systems and directories with different file types and sizes.
* **Deployment**: After the program has been tested and validated, it can be deployed to the target system. This involves installing the necessary dependencies and libraries, configuring the program settings, and verifying that the program functions properly.
* **Maintenance**: The maintenance phase is critical in ensuring the program remains useful and effective. This involves addressing and resolving bugs, incorporating new features as required, and updating the program to handle new file types or extensions. Improvements to the user interface may also be made based on user feedback.

The program will start by prompting the user to specify the directory to be sorted. After the user has selected the directory, the program will create folders for each file type or file extension found in the directory. If a folder already exists for a specific file type, the program will skip creating a new folder.

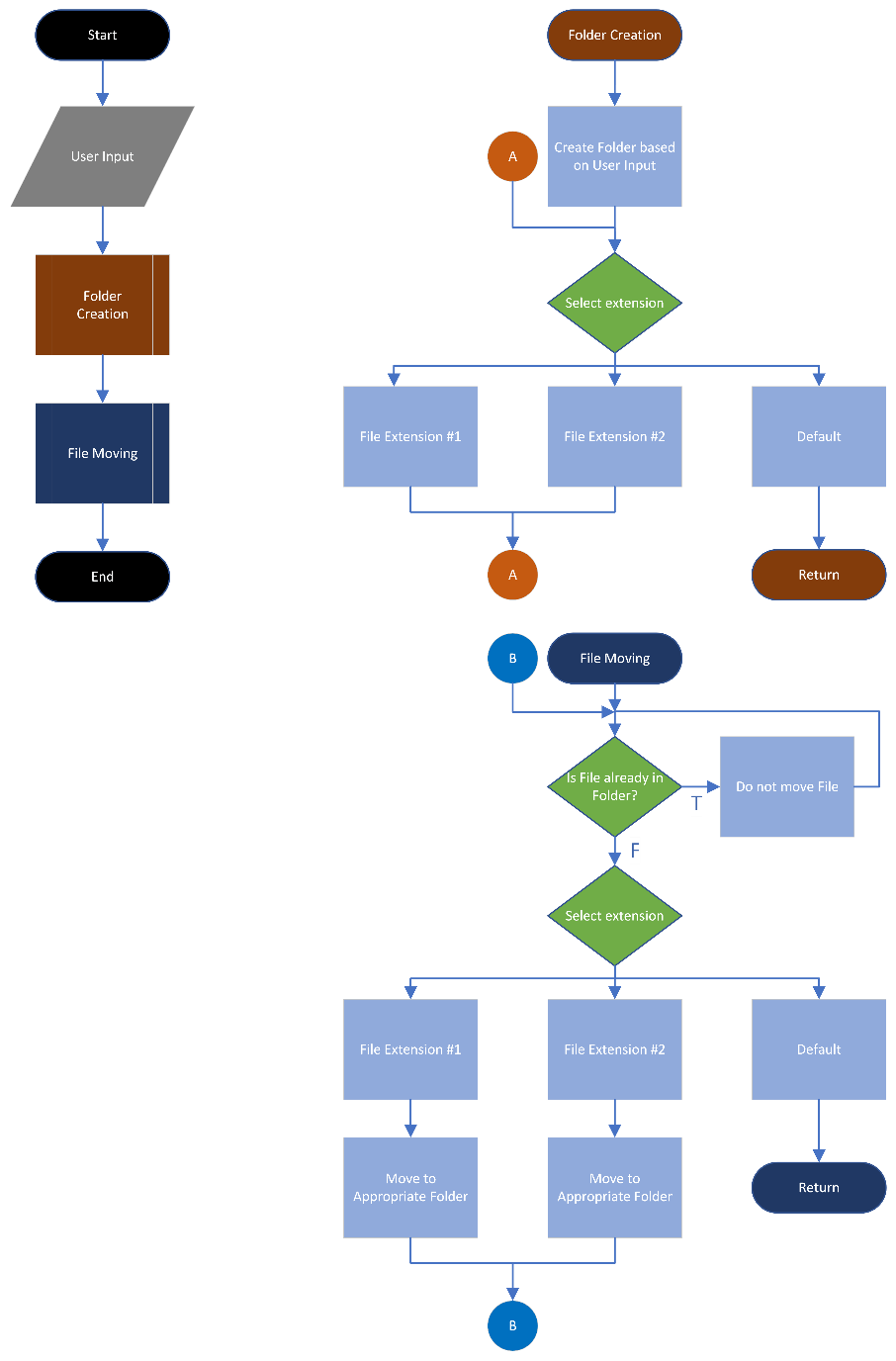
IPO

The input of the "File Organizer" program is the directory path of the folder to be organized. The user provides this input through a user-friendly interface. The program accepts the input and checks for the files in the directory. The input is a critical component of the program since the correct directory must be provided to organize the files accurately.

The "File Organizer" program automatically creates a folder for each file type (e.g., .mp3, .mp4, .docx, etc.) and moves the files to their corresponding folder based on their file extension. The process involves reading the files in the directory provided by the user and checking their file extensions. The program then creates folders for each file type that does not already have a folder and moves the files to their corresponding folders. The program has built-in error handling to handle common exceptions that may occur during the file organization process, such as skipping files that are already in their respective folders. The program's process is critical since it automates the tedious and time-consuming task of organizing files, making it more efficient and less prone to error.

The output of the "File Organizer" program is a well-organized directory where each file type has its own folder, and all files are stored in their corresponding folder based on their file extension. The program's output saves users time by making it easy for them to find and access their files. The program's well-organized output is critical to its functionality, making it easy for users to locate their files and access them quickly.

Methodology



**User Input**: The user inputs the directory path of the folder to be organized using a user-friendly interface.

**Folder Creation**: After sorting the files, the program creates a folder for each file type found in the input directory. This is done using Python's “os” module, which allows us to create directories.

**File Moving**: Once the folders are created, the program moves each file to the folder corresponding to its file type. This is done using Python's “shutil” module, which allows us to move files between directories.

**Error Handling**: To ensure the program runs smoothly, the program has built-in error handling to handle common exceptions that may occur during the file organization process. For example, if a file is already in the folder, the program will skip moving it.

The following Python concepts will be used to develop the project:

**File I/O**: The program will use file input/output functions to read and write files.

**OS Module**: The program will use the OS module to create and delete folders, move files, and check file paths.

**SHUTIL Module**: The project will use the SHUTIL module since it provides a high-level interface for efficiently copying and moving files and directories on the file system.

**GUI Programming**: The program will use GUI programming concepts to create a user-friendly interface for inputting the directory to be organized.

**Error Handling**: The program will use try/except statements to handle exceptions that may occur during the file organization process, such as a file not found, or a file already exists in the folder.

**Schedule of Activities**

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Start Date (MM/DD/YYYY) | End Date (MM/DD/YYYY) | Assigned Person/s |
| Requirements Gathering | 03/16/2023 | 03/17/2023 | Lim, Simbajon |
| Design | 03/18/2023 | 03/20/2023 | Lim, Obispo, Simbajon |
| Development - User Interface | 03/21/2023 | - | Obispo |
| Development - Folder Creation | - | - | Obispo |
| Development - File Moving | - | 03/30/2023 | Obispo |
| Testing | 03/30/2023 | 03/31/2023 | Lim, Simbajon |
| Deployment | 04/01/2023 | 04/03/2023 | Obispo |
| Maintenance | 04/03/2023 | - | Lim, Obispo, Simbajon |

\* Each Development stage can be condensed into a whole stage hence the empty cells

\*\* Maintenance stage will not have an end date since it is maintaining the program to its optimal condition

**References**

Chen, A. (2020, June 18). Python for beginners: How to write a simple file organizer code. Dev Genius. <https://blog.devgenius.io/python-for-beginners-how-to-write-a-simple-file-organizer-code-fd6a12eb4b3d>.

GeeksforGeeks. (n.d.). OS Module - Python examples. Retrieved March 16, 2023, from <https://www.geeksforgeeks.org/os-module-python-examples/>.

GeeksforGeeks. (n.d.). shutil module in Python. Retrieved from <https://www.geeksforgeeks.org/shutil-module-in-python/>.

Loeber, P. (2022, June 5). File organization with Python [Blog post]. Retrieved from <https://www.python-engineer.com/posts/file-organization/>.

Al Sweigart (2019). Automate the Boring Stuff with Python, 2nd Edition, Chapter 10: Organizing Files. No Starch Press. <https://automatetheboringstuff.com/2e/chapter10/>.

"TutorialsTeacher." tutorialsTeacher, [www.tutorialsteacher.com/python/os-module](http://www.tutorialsteacher.com/python/os-module).