# Automated File Organizer - Python Task

## Python Code

import logging  
import shutil  
from pathlib import Path  
  
# Configure logging  
logging.basicConfig(  
 filename='organizer.log',  
 level=logging.INFO,  
 format='%(asctime)s - %(levelname)s - %(message)s'  
)  
  
# File type mapping  
CATEGORIES = {  
 ".py": "Python\_Code",  
 ".txt": "Documents",  
 ".jpg": "Images",  
 ".png": "Images",  
 ".pdf": "Documents",  
 ".csv": "Data",  
 ".xlsx": "Data",  
 ".mp3": "Music",  
 ".mp4": "Videos",  
 ".docx": "Documents"  
}  
  
def organize\_directory(source: Path, dry\_run: bool = False):  
 if not source.exists():  
 return f"Error: Source directory does not exist: {source}"  
  
 summary = {}  
 for item in source.rglob("\*"):  
 if item.is\_file():  
 ext = item.suffix.lower()  
 category = CATEGORIES.get(ext, "Other")  
 target\_dir = source / category  
 summary[category] = summary.get(category, 0) + 1  
  
 try:  
 if not target\_dir.exists():  
 if not dry\_run:  
 target\_dir.mkdir(parents=True, exist\_ok=True)  
  
 target\_path = target\_dir / item.name  
 counter = 1  
 while target\_path.exists():  
 new\_name = f"{item.stem}({counter}){item.suffix}"  
 target\_path = target\_dir / new\_name  
 counter += 1  
  
 if not dry\_run:  
 shutil.move(str(item), str(target\_path))  
  
 except Exception as e:  
 logging.error(f"Error moving {item}: {e}")  
  
 return summary  
  
# Create dummy dataset and run  
base = Path("sample\_files")  
base.mkdir(exist\_ok=True)  
dummy\_files = [  
 "report.txt", "data.csv", "script.py", "photo.jpg", "music.mp3",  
 "video.mp4", "doc.docx", "notes.pdf", "diagram.png"  
]  
  
for file in dummy\_files:  
 file\_path = base / file  
 if not file\_path.exists():  
 file\_path.write\_text(f"Dummy content for {file}")  
  
# Run the organizer  
output\_summary = organize\_directory(base, dry\_run=False)  
print(output\_summary)

## Program Output

{'Documents': 6, 'Data': 2, 'Python\_Code': 2, 'Images': 4, 'Music': 2, 'Videos': 2}