UW Automation Program

README Documentation

Generated on: July 11, 2025 at 11:43 AM

Repricing Automation Toolkit (Enhanced)

This package contains the fully optimized and enhanced version of your Python-based Repricing Automation system, with improvements in structure, logging, performance profiling, and configuration management.

Contents

File	Purpose
app.py	Main GUI entry point with Excel templating, async paste, audit logging, dark mode, and real-time progress
merge.py	Merges two datasets, applies formatting
openmdftier.py	Processes Open MDF Tier logic
openmdfbg.py	Processes Open MDF Brand/Generic logic
tierdisruption.py	Standard tier disruption processor
bgdisruption.py	Brand/Generic disruption processor
sharxlbl.py	Generates SHARx Line By Line output
eplslbl.py	Generates EPLS Line By Line output
mphelpers.py	Multiprocessing helper for identifying reversals and matching claims with 'OR' logic
utils.py	Common utilities (logging, ID standardization, filtering)
configloader.py	Centralized config resolution via OneDrive
audithelper.py	Safe wrapper around audit logging (makeauditentry)
excelutils.py	Async/safe Excel handling via xlwings with COM fallback
profilerunner.py	Runs performance profiling on app.py using cProfile

---

Setup Instructions

1. Environment Requirements

- Python 3.13.5 (64-bit) or newer
- Dependencies:
- pandas, openpyxl, xlsxwriter, customtkinter, plyer, pywin32, numba, pyarrow, xlwings

Use pip to install all dependencies:

```
  pip install pandas openpyxl xlsxwriter plyer numba pywin32 customtkinter xlwings pyarrow
```

> Windows-only features like win32com.client require pywin32.
> Async Excel support via xlwings will fallback to COM automation if necessary.

Usage Guide

Launch the Application

```
python app.py
```

Run a Performance Profile

```
python profile_runner.py
```

Generates profilestats.prof and prints slowest calls.

Audit Logging

All scripts log user actions/errors to:

```
%OneDrive%/True Community - Data Analyst/Python Repricing Automation Program/Logs/audit_log.cs
```

If unreachable, logs fall back to:

```
local_fallback_log.txt
```

Configuration Management

To manage file paths centrally, update:

```
file_paths.json
```

Or adapt configloader.py to replace JSON with .env or .toml later if preferred.

Additional Notes

- Logs are rotated (utils.log) to avoid file bloat.
- All enhancements maintain full backward compatibility with your original logic.
- UI now supports full dark/light mode toggle and async Excel pasting with live progress.
- For setup, troubleshooting, and advanced configuration, see SETUPGUIDE.txt and the walkthrough documentation.

For questions or future updates, contact your IT department or the program maintainer.