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
`openmdf_tier.py`	Processes Open MDF Tier logic
`openmdf_bg.py`	Processes Open MDF Brand/Generic logic
`tier_disruption.py`	Standard tier disruption processor
`bg_disruption.py`	Brand/Generic disruption processor
`sharx_lbl.py`	Generates SHARx Line By Line output
`epls_lbl.py`	Generates EPLS Line By Line output
`mp_helpers.py`	Multiprocessing helper for identifying reversals and matching claims with 'OR' logic
`utils.py`	Common utilities (logging, ID standardization, filtering)
`config_loader.py`	Centralized config resolution via OneDrive
`audit_helper.py`	Safe wrapper around audit logging (`make_audit_entry`)
`excel_utils.py`	Async/safe Excel handling via xlwings with COM fallback
`profile_runner.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:

```bash
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

```bash
python app.py
```

Run a Performance Profile

```bash
python profile_runner.py
```

Generates `profile_stats.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.csv
```

If unreachable, logs fall back to:

```
local_fallback_log.txt
```

Configuration Management

To manage file paths centrally, update:

```json
file_paths.json
```

Or adapt `config_loader.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 `SETUP_GUIDE.txt` and the walkthrough documentation.

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