PDF Downloader: Design Document

Section 1: Overview

This document describes the design and usage of two Python scripts developed to automate the downloading and organization of PDF filings from web sources. The solution is intended for users who need to regularly download, catalog, and parse documents for financial analysis.

Goals

- Automatically download .pdf files from specified web pages and organize them into a folder structure by section and year.
- Analyze downloaded PDFs to append relevant filing-type identifiers to their filenames.
- Provide a robust, repeatable system for non-technical users with minimal setup.

Components

1. **`pdf_downloader.py`:** Crawls the provided URLs and downloads all matching PDF documents.
2. **`update_filing_type_filenames.py`**: Post-processes downloaded PDFs to analyze contents and rename them with matching filing-type codes.

Section 2: Environment Setup

The scripts are designed to run inside a Python virtual environment.

Step-by-Step Setup (macOS/Linux)

1. **Install Python (if not already installed)**:
    - Download from https://www.python.org/downloads
    - Ensure `python3` and `pip3` are available in the terminal:

      ```
      python3 --version
      pip3 --version
      ```

2. **Navigate to the project folder**:

```
cd pdf_downloader_2025
```

3. **Run the setup script**:

```
bash setup.sh
```

The setup script:

- Creates a Python virtual environment in the `venv/` folder.
- Installs all required packages using pip.

4. **Activate the environment**:

```
source venv/bin/activate
```

5. **Launch the GUI (optional)**:

```
python gui_launcher.py
```

Section 3: Input Files

Filename	Description	Used By Script(s)
`base-url.csv`	Contains the root URL for scraping.	*`pdf_downloader.py`*
`sec-grp-url.csv`	Contains the SEC group URL for scraping and directory paths.	*`pdf_downloader.py`*, *`update_filing_type_filenames.py`*
`year-url.csv`	Contains the year URL for scraping and directory paths.	*`pdf_downloader.py`*, *`update_filing_type_filenames.py`*
`sec-grp-det.csv`	Maps each SEC group and SEC filing type.	*`update_filing_type_filenames.py`*

These files are located in the input folder and work together to build both the URL list and the directory structure for organizing downloads and renaming files.

Section 4: Script: `pdf_downloader.py`

Purpose

To crawl specified pages for PDFs and download them into structured folders based on section and year.

Folder Structure Created

```
downloaded_pdfs/
  ├── <sec-grp-url>/
  │   ├── <year-url>/
  │   │   └── downloaded-file.pdf
```

Key Features

- Supports multi-page pagination.
- Avoids re-downloading existing files.
- Tracks all downloaded metadata into a `filing_metadata.csv`.
- Logs progress and issues in the `logs/` directory.

Execution

```
python pdf_downloader.py
```

Section 5: Script: `update_filing_type_filenames.py`

Purpose

Post-processes previously downloaded PDFs, scans their text, and appends matching `filing-typ` values to the filename.

Folder Structure Used

The script operates on the output folder created by `pdf_downloader.py`:

```
pdf_downloader_2025/
├── downloaded_pdfs/
│   ├── <sec-grp-url>/
│   │   ├── <year-url>/
│   │   │   └── *.pdf
```

Matching Strategy

- Reads the first page of each PDF.
- Searches for keywords defined per section in `sec-grp-det.csv`.
- Appends the matched keyword to the filename (if not already present).

Example:

```
Original: 2025-03-12_abc123.pdf
Renamed:  2025-03-12_abc123_10-K.pdf
```

Execution

```
python update_filing_type_filenames.py
```

Logging

Creates a structured CSV log:

`logs/filing_type_update_log_YYYYMMDD_HHMMSS.csv`

Containing:

- `original_filename`
- `new_filename`
- `matched_filing_type`

Section 6: Common Issues & Debugging

Problem	Solution
`ModuleNotFoundError` for a lib	Run `source venv/bin/activate` then `pip install -r requirements.txt`
PDF not renamed	Ensure filing-type keyword appears on the PDF's first page
GUI won't launch	Run with `python gui_launcher.py` inside the virtual environment
No files downloaded	Check base URL and pagination inputs in the CSVs

Section 7: Financial Statements Use Case

One practical use case for this system is extracting structured data from quarterly SEC filings, such as 10-Q documents. After downloading and renaming the relevant filings using the provided scripts, users can apply additional parsing to extract tables from the PDFs into spreadsheets.

The file ***`alphabet_q2_2025_financials_with_backlog.xlsx`*** showcases how financial tables from Alphabet Inc.'s Q2 2025 10-Q PDF were extracted and converted into an Excel file for structured analysis.

Income Statement

Summarizes Alphabet's income from operations.

Balance Sheet

Summarizes Alphabet's assets, liabilities, and equity as reported in the quarterly balance sheet. This tab reflects the company's financial position at the end of the reporting period.

Cash Flows

Summarizes Alphabet's cash inflows and outflows from operating, investing, and financing activities. This section aligns with the official quarterly statement of cash flows.

Segment Results

Breaks down Alphabet's revenue contributions by business segments, such as Google Services, Google Cloud, and Other Bets. This tab includes subtotals, growth trends, and inter-segment analysis.

Disaggregated Revenues

Provides a detailed breakdown of revenue categories and sources as disclosed in the notes to financial statements. Includes numerical allocations by business type or customer geography.

Performance Obligations (Backlog)

Includes backlog-related disclosures from Alphabet's financial footnotes, and presents remaining performance obligations across upcoming quarters. Useful for forecasting revenue timing.

This use case demonstrates how the output of the PDF downloader can be extended as a useful tool for financial analysis.

Section 8: Final Notes

This system is designed to operate offline once URLs and logic are configured. Users should periodically update the input source files to maintain accuracy.

Section 9: Revision Log

Version	Description of changes	Date	Author
1.0	Initial draft	7/27/2025	Erik Eckerson