Document Retrieval Automation (DRA)

Problem Statement

- The current process of accessing a web portal to manually log in, navigate through multiple web sections, identify and download document attachments and then organize and upload these files to OneDrive or SharePoint is time-consuming.
- This repetitive task consumes significant manual effort, particularly in sorting documents such as invoices and reports based on content.
- There is a need to automate this workflow using a Python-based solution to improve accuracy, reduce processing time, and enable scalable handling of document retrieval and storage.

Proof of Concept (PoC) – GitHub Repository Automation

This Proof of Concept (PoC) demonstrates the feasibility of automating a manual file retrieval and organization task using Python. The goal is to simulate a real-world workflow where a script reads input data from an Excel file, navigates a web portal, checks for the presence of a specific file attachment, downloads it if available, and organizes it based on file type.

In this PoC:

- An Excel file containing a list of GitHub repository URLs is used as input.
- The script automatically opens each repository using a web automation tool (Selenium).
- It checks for the existence of a target file (e.g., report.pdf) in a predefined location within each repository.
- If the file is found, it is downloaded and saved locally.
- Downloaded files are then sorted into folders based on file type (e.g., PDF reports).
- This PoC serves as a foundational test for a more complex automation workflow involving secured portals, multiple navigation steps, file classification, and cloud upload (e.g., OneDrive or SharePoint), to be implemented in future stages.

Tools to Use for PoC

Steps	Tools / Python Library
Write the Python script	Visual Studio Code + Python
Read Excel	pandas
Navigate GitHub	selenium
Download File	requests or browser click
Organize File	os, shutil
Optional Upload to OneDrive	*Skip for now*

PoC Task Definition

Purpose: Automate a script that:-

1. Reads repo names from an Excel file.
2. Logs into GitHub (optional if public repos).
3. Visit each repo.
4. Checks if a specific file (e.g., `report.pdf`) exists.
5. If it does, download it.
6. Sorts the file into folders based on type (e.g., `.pdf` = Reports).
7. (Optional) Uploads to OneDrive later.

Development Environment Setup

Step 1: Set Up Python Environment

- Install Visual Studio Code as an Integrated Development Environment (IDE) to write the code or script.
- Install Python version 3.11: https://www.python.org/downloads/release/python-3110/

  ⚠️ **IMPORTANT !**
  - When installing, **make sure to check this box**:
    ✅ **"Add Python to PATH"** (at the bottom of the install window)
  - This ensures you can run Python from the Command Prompt or Terminal.

Step 2: Install Required Python Libraries

Run the following commands **one at a time** in your terminal.

```
pip install pandas
pip install openpyxl
pip install selenium
pip install requests
```

Step 3: Download ChromeDriver

Selenium needs a driver to control the Chrome browser.

1. Find your Chrome version

- Open Chrome
- Go to `chrome://settings/help`
- Note the version number (e.g., 114.0.5735.199)

2. Download matching ChromeDriver

- Go to: https://googlechromelabs.github.io/chrome-for-testing/
- Click your version (e.g., 114.0.5735.199)
- Download the ChromeDriver for your OS
- Extract the ZIP file
- You should get a file called chromedriver.exe (on Windows)