Phaidra Metadata & License Auditor

Course: Data Steward Certificate

Developer: Jonas Kerschner (with his advisor Google Gemini)

Date: January 2026

Requirements & Problem Description

The objective of this project is to develop a robust metadata auditing tool for the Phaidra Repository. As a Data Steward, it is vital to monitor the technical formats (MIME-types) and legal accessibility (Licenses) of digital objects. Manually checking thousands of objects via the web interface is not feasible.

The solution is a Python-based automation tool that harvests metadata via the REST API, validates user inputs, categorizes legal statuses, and exports findings into both human-readable (PNG) and machine-readable (CSV) formats.

Technical Design

The software follows a sequential Waterfall lifecycle:

1. **Requirements:** Define date ranges and identify key metadata fields (dc_license, file_mimetype).
2. **Design**: Implement a modular architecture separating Data Acquisition, Analysis, and Visualization.
3. **Implementation:** Build a Python-based solution utilizing the requests library for API interaction and matplotlib for data representation.
4. **Verification:** Conduct boundary testing (e.g., checking the year 2008) to ensure system integrity.

Implementation Details

- **Input Validation:** The program utilizes nested conditional logic and while loops to ensure user inputs are numeric and within the operational bounds of the Phaidra system (>= 2008).
- **Data Acquisition:** A pagination strategy is employed to harvest records in batches of 100, ensuring memory efficiency and preventing server-side timeouts.

- **Metadata Fallback Logic:** To account for inconsistent metadata entry, the tool checks multiple fields (mimetype --> dc_format --> resourcetype) to determine the technical format.
- **License Classification:** Using a keyword-matching algorithm, the tool categorizes licenses into "Open Access" (e.g., CC-BY) or "Restricted" categories.

Output and Verification

The program generates four primary outputs:

1. **report_totals.png:** A bar chart illustrating the volume of data per format.
2. **report_licenses.png:** A grouped bar chart visualizing the ratio of Open Access compliance.
3. **phaidra_detailed_list.csv:** A granular audit log containing every unique PID.
4. **phaidra_summary_stats.csv:** A statistical summary table for executive reporting.

Boundary Value Test (Input Validation)

- **Scenario A:** User enters "2005".
    - **Result:** Program identifies the year is < 2008, prints a specific error message, and restarts the loop.
- **Scenario B:** User enters "Dog".
    - **Result:** isdigit() check fails. Program prints "not a year" and restarts the loop.
- **Scenario C:** User enters "2024".
    - **Result:** Input is accepted; API harvest begins.

Data Integrity Check

The numFound value from the initial API handshake was compared against the final number of rows in the phaidra_detailed_list.csv.

- **Finding:** The counts matched exactly, verifying that the **Pagination Loop** successfully captured every record without data loss.

Appendix:

The Hitchhiker’s Guide to the Phaidra Auditor (Logic Map)

If you find yourself confused by the code, Don’t Panic. Just follow this entry from the Guide:

The “Gatekeeper” (Input Validation)

The computer is like a **Vogonic security guard**.

- **The Loop:** It puts you in a small room and won't let you out until you fill out the "Year Form" correctly.
- **The digit check:** If you type "Forty-Two" instead of "42", the guard gets confused and makes you start over.
- **The 2008 check:** If you try to audit 1980, the guard tells you that the Phaidra galaxy hadn't been invented yet. Only a valid number >=2008 acts as the "Electronic Thumb" that lets the program move forward.

The “Harvester” (Data Acquisition)

Think of this as the **Heart of Gold's Infinite Improbability Drive**.

- **The Handshake:** The script sends a Sub-Etha signal to Phaidra asking: *"How many objects are in this sector of time?"*
- **The Total:** Deep Thought (Phaidra) returns a number.
- **The Pagination:** Since we can’t swallow the entire Encyclopedia Galactica at once, we use "Pagination." We take 100 items at a time, store them in our bucket (the "all_docs" list), and go back for more until we have it all.

The Towel (JSON File)

Before we do anything else we wrap our data in a towel (phaidra_audit_json).

- Why? If the “Vogons” (the internet connection) decide to demolish the connection halfway thoughm we still have our data wrapped up safely on our hard drive.
- The Utility: A Data Steward who knows where their towel is, os a person to be reckoned with. I allows us to analyze the data over and over again, without having to take the risk to listen to Vogonic Poetry ever again.

The “Sorting Office” (Analysis)

Once we have the data, we have to sort it.

- **The Fallback:** If an object doesn't have a label for its format, we check its second pocket (Mimetype), and then its third (Format). We don't stop until we know what it is.
- **The Babel Fish:** We translate the messy License text. We look for keywords like "CC-BY."
    - If found: It is categorized as **"Mostly Harmless"** (Open Access).
    - If not found: It is marked as **"Vogon-Style"** (Restricted).

The “Artist & Clerk” (Output)

Finally, we create the reports for the Galactic Board of Stewards.

- **The Clerk:** It writes two CSV files. One is a giant list of every PID (for the bureaucrats), and one is a summary (for the busy people).
- **The Artist:** It draws two pictures. One shows the total volume of the galaxy, and the other compares the "Open Access" planets to the "Copyright" planets.