# Data Harvesting & Structuring Assignment

**Due Date: _____ Assigned To: _____**

---

## 1. Target Websites

Please scrape and process documents from the following sources:

- https://sanskritdocuments.org/scannedbooks/asisanskritpdfs.html
- https://sanskritdocuments.org/scannedbooks/asiallpdfs.html
- https://indianculture.gov.in/ebooks
- https://ignca.gov.in/divisionss/asi-books/
- https://archive.org/details/TFIC_ASI_Books/ACatalogueOfTheSamskritManuscriptsInTheAdyarLibraryPt.1/
- https://indianmanuscripts.com/
- https://niimh.nic.in/ebooks/ayuhandbook/index.php

---

## 2. Exercise Tasks

Complete the following steps in your script or microservice. Organize your code clearly and include documentation for each step.

1. **Crawl & Download**
    - Recursively fetch each URL and all internal links.
    - Download any PDFs, EPUBs or HTML resources.
    - Respect `robots.txt` and include a 1–2 second delay between requests.
2. **Metadata Extraction**
    - From each page or document, extract:

        - **title**
        - **author/editor**
        - **publication_year**
        - **language**
        - **document_id** (construct a unique ID)
    - Compute a **SHA-256 checksum** of every downloaded file.
3. **JSON Structuring**
    - Normalize all fields and output a JSON record per document, for example:
      {
 "site": "ayushportal.nic.in",
 "document_id": "doc1234",
 "title": "Ancient Text on Ayurveda",
 "authors": ["Name Surname"],
 "pub_year": "1998-05-10",
 "language": "Sanskrit",
 "download_url": "https://.../doc1234.pdf",
 "checksum": "a1b2c3...",
 "scraped_at": "2025-04-26T10:15:00Z"

}

    - Ensure dates use **ISO 8601** and author names follow a consistent format.
4. **Text Extraction & OCR**
    - For each downloaded file:
        - If it already contains embedded text, extract it directly.
        - Otherwise run **Tesseract OCR** or **Apache Tika** to generate a text layer.
    - Attach the full extracted text to the JSON record under a `content` field.
5. **Delta Processing**
    - On subsequent runs, use HTTP **Last-Modified** headers or compare stored checksums.
    - Only re-download or re-parse documents that are new or have changed.

---

# 3. Test Cases

When you submit, we should be able to verify the following:

| Test Case | Input / Action | Expected Outcome |
| --- | --- | --- |
| **TC1: Crawl Basic Page** | Run crawler on `ayushportal.nic.in/default.aspx` | HTML saved; PDF links identified and downloaded. |

| | | |
|---|---|---|
| **TC2: Metadata JSON** | Process a sample PDF | JSON record contains all required fields with correct formats. |
| **TC3: OCR Extraction** | Provide a scanned-only PDF | `content` field contains readable text extracted by OCR. |
| **TC4: Checksum & Delta** | Modify a document and re-run | Script flags changed file (new checksum) and re-processes it. |
| **TC5: JSON Schema Validation** | Validate JSON output | All records pass schema validation (e.g., using JSON Schema). |

---

**Notes:**

- Include a brief **README** explaining how to install dependencies and run your script.
- Aim for **readable, maintainable code** with error handling and logging.
- Prepare to demonstrate or discuss your approach and any trade-offs made.

Good luck!