{SUMMER INTERNSHIP PROJECT REPORT}

Group-21

Document Extractor: Document Content Extraction and Analysis using NLP

Tanisha Sharma,

B.Tech Computer Science with Spl. In AI and Robotics

Vellore Institute of Technology, Chennai

Project Guide / Mentor Name:

Dr. Diptendu Dutta

Period of Internship: 19th May 2025 - 15th July 2025 (Do not change the dates)

Report submitted to: IDEAS – Institute of Data Engineering, Analytics and Science Foundation, ISI Kolkata

1. Abstract

This project involved developing a full-fledged NLP application in Python. Document Extractor is a web-based tool that can extract and process data from multiple document types, i.e., PDF, DOCX, PPTX, and Excel sheets. The app uses popular libraries such as PyMuPDF, pdfplumber, tabula-py, Camelot, and Streamlit for an interactive development environment. It simplifies the automatic extraction of text, tables, images, and metadata, with structured outputs and visualizations. In addition, the solution provides data validation, error management, and result aggregation modules. Legal document review, research support, and business data analysis can be facilitated in the application by integrating content processing workflows.

2. Introduction

Document content extraction is a difficult task across various domains like data science, legal informatics, and business intelligence. Conventionally, extracting structured data from various document formats is a time-intensive process involving much manual intervention. This project overcomes this difficulty by designing an end-to-end automated pipeline for document processing and analysis.

Technologies used:

- Python (version 3.10)
- Streamlit (Web-based UI)
- PyMuPDF, pdfplumber, tabula-py, Camelot (PDF parsing)
- OpenCV and Pillow (image processing)
- python-docx, python-pptx, openpyxl (processing of Office documents)
- pandas, NumPy (data manipulation)

Training topics which were taught during the initial two weeks:

- Introduction
- Introduction to Data Science
- How to do a Research Project
- Data Visualization - Power BI
- Career & Life-Design
- Streamlit
- ML
- Deep Learning
- Gen AI and LLM Foundations

3. Project Objective

- Create a single platform to extract text, table, image, and metadata from PDFs and Office documents.
- Offer an interactive web interface for uploading, processing, and rendering extracted data.
- Have advanced error handling to handle varying document layouts and structures.
- Offer downloadable summaries and structured datasets (CSV/JSON) for later analysis.
- Compare and verify extraction accuracy with several libraries (e.g., tabula-py vs. Camelot).
- Make it easier to use by developing a no-code user interface for non-technical end-users.

4. Methodology

1. **Requirements Analysis:**

    - Identified the need for a multi-format document extractor.
    - Evaluated libraries supporting extraction.

2. **Environment Setup:**
    - Created a Python environment.
    - Installed dependencies listed in requirements.txt (Streamlit, PyMuPDF, etc.).
3. **Architecture Design:**
    - pdfextract.py: Implemented the ComprehensiveDocumentExtractor class for modular extraction logic.
    - app.py: Developed the Streamlit web application.
4. **Module Implementation:**
    - **PDF Parsing:** Extracted text, tables, images using PyMuPDF, pdfplumber, tabula-py, Camelot.
    - **DOCX/PPTX/Excel Processing:** Parsed content and metadata.
    - **Image Extraction:** Saved embedded images to structured directories.
    - **Data Validation:** Implemented heuristics to verify tables and avoid false positives.
5. **User Interface:**
    - Streamlit UI with upload widgets, processing progress, and result tabs.
    - Custom CSS for styling.
6. **Testing and Validation:**
    - Processed sample documents with known content.
    - Compared table extraction accuracy across libraries.
    - Verified handling of scanned PDFs.
7. **Output Generation:**
    - Created JSON/CSV summaries.
    - Enabled downloads of extracted text and tables.
    - Generated summary reports automatically.
8. **Version Control:**
    - Codebase maintained on GitHub and executed on Streamlit.

Links:-

Github:-

https://github.com/tanisha1030/pdf-extract

Streamlit:-

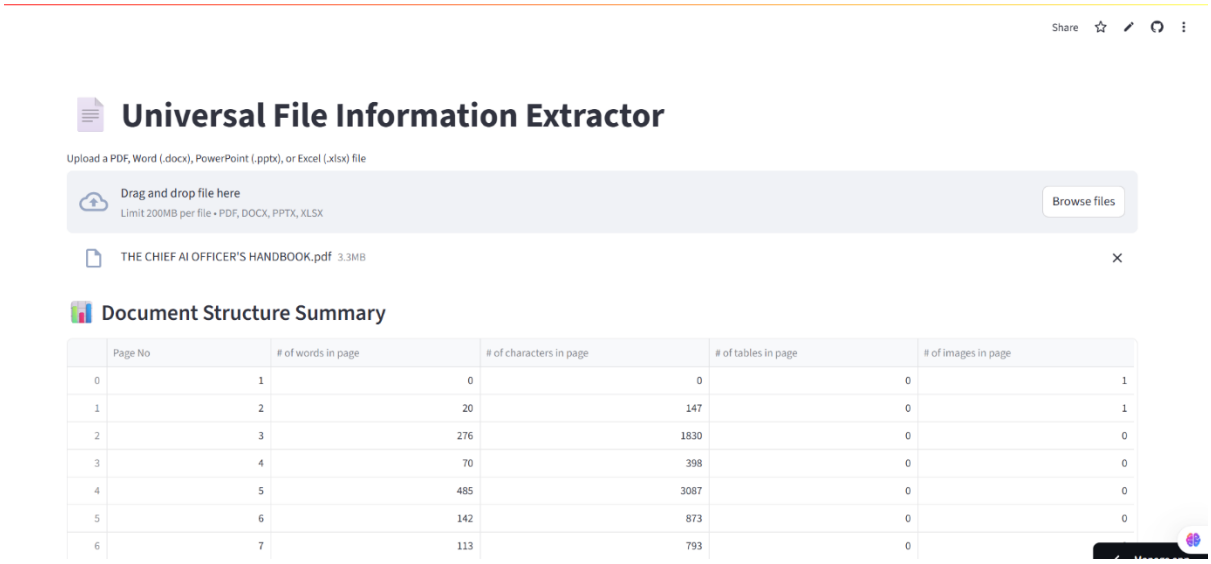
https://pdf-extract-ximtkohqgldzujrvlnjtmy.streamlit.app/

5. Data Analysis and Results

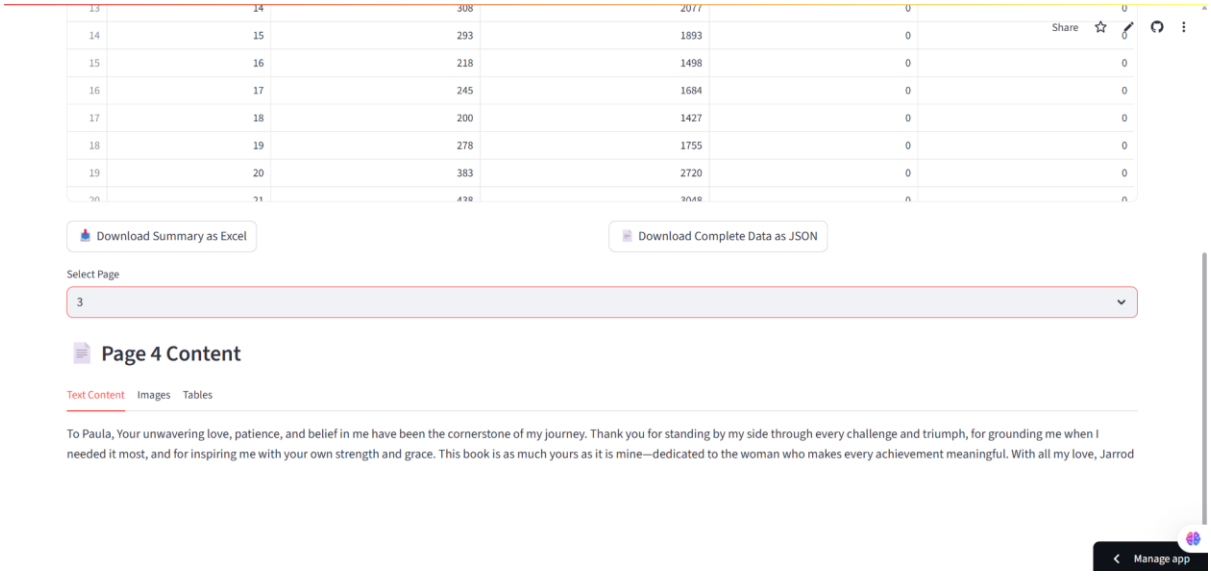
Descriptive Analysis:

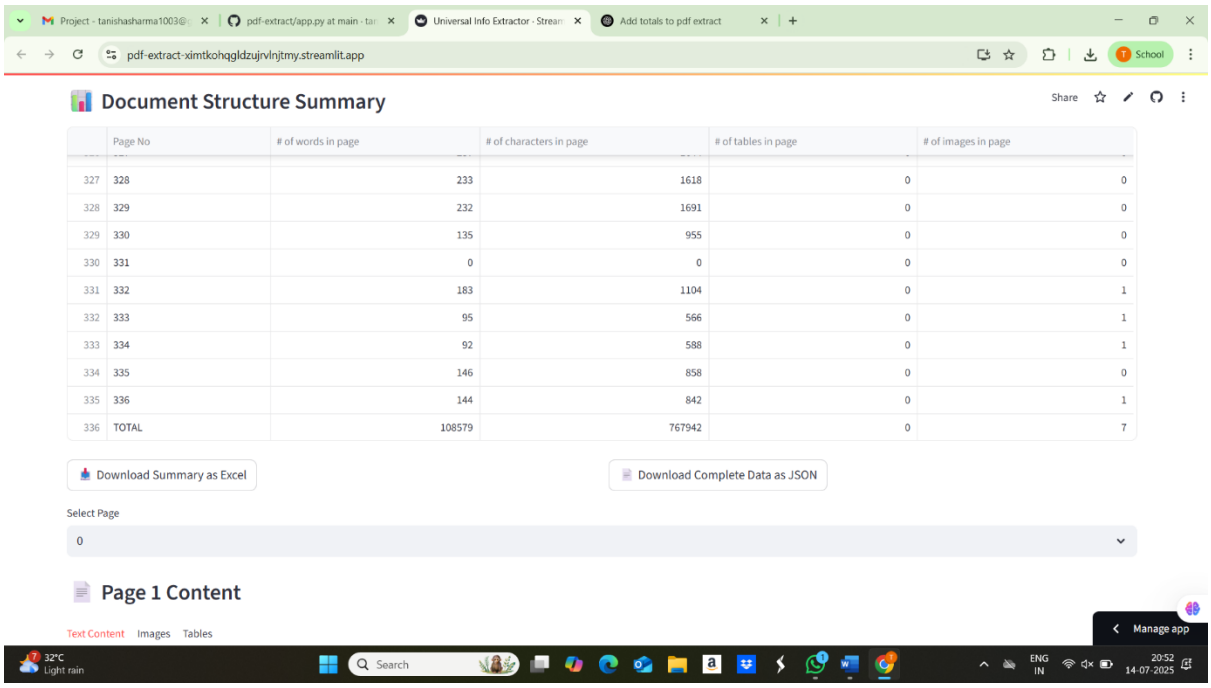
I took a sample file given by my mentor and I got the following sample result

- Maximum size of processed files: 200MB
- Cumulative pages processed: 335
- Total words extracted: 108579
- Total characters extracted: 767942
- Tables extracted: 0
- Images extracted: 7



	Page No	# of words in page	# of characters in page	# of tables in page	# of images in page
0	1	0	0	0	1
1	2	20	147	0	1
2	3	276	1830	0	0
3	4	70	398	0	0
4	5	485	3087	0	0
5	6	142	873	0	0
6	7	113	793	0	





6. Conclusion

The Document Extractor Pro project is well demonstrated to facilitate a scalable extraction of structured information from different document types. The use of several extraction libraries greatly enhanced accuracy and stability. Future work can involve OCR improvements to handle enhanced scanned PDF processing and support for more file formats (e.g., HTML). The tool is applicable to real-world document review, data gathering, and content archiving processes..

7. APPENDICES

1. PyMuPDF Documentation
   https://pymupdf.readthedocs.io/en/latest/
2. pdfplumber GitHub Repository
   https://github.com/jsvine/pdfplumber

3. tabula-py Documentation
https://tabula-py.readthedocs.io/en/latest/
4. Camelot Documentation
https://camelot-py.readthedocs.io/en/master/
5. Chatgpt
6. Claude.ai
7. Github repo
https://github.com/tanisha1030/pdf-extract
8. Streamlit link
https://pdf-extract-ximtkohqgldzujrvlnjtmy.streamlit.app/
9. https://pythonology.eu/what-is-the-best-python-pdf-library/
10. https://pymupdf.readthedocs.io/en/latest/about.html
11. https://github.com/opendatalab/PDF-Extract-Kit
12. https://github.com/docling-project/docling