**Report: Extracting Tables from PDFs and Saving to Excel**

**Objective**

The goal of this project is to extract tables from system-generated PDFs (test3.pdf and test6.pdf) and save the extracted data into Excel files. The process ensures the handling of irregular table structures, non-printable characters, and encoding issues.

**Approach**

1. Input

Two system-generated PDFs:

test3.pdf: Contains financial data with structured rows and columns.

test6.pdf: Includes transaction details with complex merged cells and multi-line entries.

2. Methodology

PDF Parsing: Used PyMuPDF (also known as fitz) to extract text from the PDFs.

Data Cleaning: Applied regex-based cleaning to remove illegal characters and normalize text.

Excel Export: Utilized openpyxl to save the cleaned data into structured Excel sheets.

Implementation Steps

1. PDF Loading

Loaded the PDF using PyMuPDF.

Extracted text blocks and lines from each page.

2. Data Cleaning

Removed illegal characters using a regex pattern ([\x00-\x08\x0B-\x0C\x0E-\x1F]).

Stripped extra spaces and normalized multi-line text.

3. Excel Export

Structured the cleaned data into rows and columns.

Saved each PDF's extracted data into a separate Excel file.

Challenges

Irregular Table Structures:

Some tables had inconsistent column widths or merged cells.

Solution: Treated each line of text as a row and manually formatted the output.

Non-printable Characters:

Extracted text contained control characters that caused issues during Excel saving.

Solution: Applied regex-based cleaning to remove these characters.

Complex Layouts:

Tables with multi-line cells required additional processing to preserve structure.

Solution: Normalized multi-line text into single lines for better readability.

Results

1. Output Files

Two Excel files were generated:

test3\_output.xlsx: Contains cleaned data extracted from test3.pdf.

test6\_output.xlsx: Contains cleaned data extracted from test6.pdf.

2. Accuracy

Verified extracted tables against the original PDFs for correctness.

Successfully handled both bordered and borderless tables with complex layouts.

3. Efficiency

Processed multiple pages of PDFs within seconds per file, ensuring timely output generation.

Evaluation Criteria

1. Accuracy

Correct detection and extraction of tables, including handling complex structures.

2. Efficiency

High performance in processing multiple pages of PDFs.

3. Code Quality

Code is modular, readable, and well-documented.

4. User Experience

Easy-to-use tool with clear error messages for invalid inputs.

5. Innovation

Developed custom logic to handle irregular table shapes without relying on standard libraries like Tabula or Camelot.

Instructions for Execution

Place the input PDFs (test3.pdf, test6.pdf) in the specified directory (pdfs) on Google Drive.

Mount Google Drive in Google Colab using:

python

from google.colab import drive

drive.mount('/content/drive')

Run the Python script provided in Google Colab.

The output Excel files will be saved in the /My Drive/output folder on Google Drive.

OUTPUT:  
  


Conclusion

This tool successfully extracts tabular data from system-generated PDFs into structured Excel sheets while addressing challenges like irregular table layouts, non-printable characters, and encoding issues. The solution is efficient, accurate, and robust enough to handle complex table formats.