# **Abstract**

The DS PDF App is a modular Python backend designed to perform various PDF manipulations in one integrated script. It supports three primary operations:

* **Collaging:** Combines two PDF pages into one output page by rotating, scaling, and positioning them side by side (or top/bottom) on a letter-sized canvas. This is particularly useful for printing multiple pages on a single sheet to save on paper and printing costs.
* **Extraction:** Allows the user to extract a specific range of pages from a PDF document and save them as a new PDF. This is ideal for isolating chapters or sections from larger documents.
* **Merging:** Concatenates multiple PDF files into a single PDF. The order of selection directly determines the output sequence, making it straightforward to merge documents chronologically.

The code is built as an extendable class (DSPDF) that groups related operations into static methods, facilitating future enhancements and integration—such as transitioning to a Flask-based web application or cloud deployment. It leverages the PyPDF2 library for PDF manipulation and Tkinter for simple graphical file dialogs. Note that parameters like the horizontal and vertical adjustments (referred to as RX and RY factors) play a crucial role in aligning PDF content. Incorrect adjustments might result in output pages with content that falls outside the visual canvas.

# **User Manual**

## **Potential Use Cases**

* **Selective Printing:** If you want to print specific chapters from a book (for example, *Blink* by Malcolm Gladwell) where the printed version restricts page count, you can extract just the needed pages.
* **Cost-Effective Printing:** When you need to print documents economically, the collage feature allows you to combine multiple pages onto a single sheet, thereby reducing paper usage and printing expenses.

## **Requirements**

* **Python 3.x:** The code is compatible with Python 3.
* **PyPDF2:** Install via pip (pip install PyPDF2) for PDF manipulation.
* **Tkinter:** Typically comes pre-installed with Python for creating simple GUI dialogs.

## **How the Code Works**

1. **Modular Design:** The DSPDF class encapsulates all PDF-related operations:  
   * collage\_pdf: Combines two pages into one output page using rotation, scaling, and positional adjustments.
   * extract\_pages: Extracts a user-defined range of pages from a PDF.
   * merge\_pdfs: Merges multiple PDFs into a single document.
2. **File Selection:** The application uses Tkinter dialogs to select input files and designate output paths. For example, the extraction module may trigger a file-saving dialog that might appear in the background—so be sure to check all open windows if you don't see it immediately.
3. **Customization with RX, RY Factors:** In the collage operation, the parameters horizontal\_offset (RX factor) and vertical\_adjust (RY factor) determine the precise placement of PDF content on the output canvas. Adjust these values to suit different PDF dimensions. Note that inappropriate values can cause the output to be misaligned or fall outside the printable area.
4. **User Interaction:** A simple command-line menu guides the user through the available operations. Depending on the choice:  
   * **Collage:** The user is prompted to select a PDF, specify an output location, and the collage is created with two pages per sheet.
   * **Extract:** The user inputs the start and end page numbers. The extraction process saves the selected pages as a new PDF.
   * **Merge:** The user selects multiple PDF files. **Important:** To achieve the desired order in the merged document, select the PDFs chronologically (the selection order determines the final order in the output).

## **Step-by-Step Guide**

**Launching the Application:** Run the script from your command line:  
  
 python dspdf\_app.py

1. You will see a menu with three options.
2. **Collage Operation:**
   * **Select Option 1** for collage.
   * Choose the input PDF file using the file dialog.
   * Choose the output file location via the save dialog.
   * The script rotates, scales, and places two pages on one sheet.  
      *Tip:* If the output seems misaligned, adjust the horizontal\_offset and vertical\_adjust values in the script.
3. **Extraction Operation:**
   * **Select Option 2** for extraction.
   * Choose the input PDF file.
   * Enter the start and end page numbers when prompted.
   * A file dialog will appear (possibly in the background) for specifying the output file name. Ensure you locate this window if it doesn’t pop up in the foreground.
   * The selected pages are then saved as a new PDF.
4. **Merge Operation:**
   * **Select Option 3** for merging.
   * Use the file dialog to select the PDFs. **Important:** Select the files in the exact order you wish them to appear in the merged document.
   * Choose the output file location.
   * The script merges the PDFs in the order selected.
5. **Extensibility:** The structure of the code allows for easy addition of new features. You can add more static methods to the DSPDF class or integrate these functions into a web framework like Flask for online use.

This comprehensive overview should help you understand, use, and extend the DS PDF App according to your needs. Enjoy the flexibility and efficiency offered by the tool!