```
# Installing essential libraries

!pip install PyPDF2 pdfplumber pymupdf reportlab
```

```
# Merging multiple PDF Files

import PyPDF2

def merge_pdfs(pdf_list, output_path):
    pdf_writer = PyPDF2.PdfWriter()
    for pdf in pdf_list:
        pdf_reader = PyPDF2.PdfReader(pdf)
        for page_num in range(len(pdf_reader.pages)):
            pdf_writer.add_page(pdf_reader.pages[page_num])

    with open(output_path, 'wb') as out:
        pdf_writer.write(out)
    print(f"Merged PDF saved as {output_path}")

# Usage
merge_pdfs(['Page 1.pdf', 'Page 2.pdf', 'Page 3.pdf'], 'merged.pdf')
```

```
Merged PDF saved as merged.pdf
```

```
# Splitting PDF file to multiple PDF files/Pages.

import PyPDF2

def split_pdf(pdf_path, output_dir):
    pdf_reader = PyPDF2.PdfReader(pdf_path)
    for page_num in range(len(pdf_reader.pages)):
        pdf_writer = PyPDF2.PdfWriter()
        pdf_writer.add_page(pdf_reader.pages[page_num])
        output_path = f"{output_dir}/page_{page_num + 1}.pdf"
        with open(output_path, 'wb') as out:
            pdf_writer.write(out)
        print(f"Saved {output_path}")

# Usage
split_pdf('merged.pdf', 'pdf_files')
```

```
Saved pdf_files/page_1.pdf
Saved pdf_files/page_2.pdf
Saved pdf_files/page_3.pdf
```

```
# Extract Text from the PDF File.

import pdfplumber

def extract_text(pdf_path, output_txt_path):
    with pdfplumber.open(pdf_path) as pdf:
        full_text = ''
        for page in pdf.pages:
            full_text += page.extract_text() + '\n'
        with open(output_txt_path, 'w') as f:
            f.write(full_text)
        print(f"Extracted text saved as {output_txt_path}")

# Usage
extract_text('merged.pdf', 'output.txt')
```

```
Extracted text saved as output.txt
```

```
# Extract images from the PDF File.

import fitz  # PyMuPDF

def extract_images(pdf_path, output_dir):
    pdf_document = fitz.open(pdf_path)
    for page_index in range(len(pdf_document)):
        page = pdf_document.load_page(page_index)
        image_list = page.get_images(full=True)
        for img_index, img in enumerate(image_list):
            xref = img[0]
            base_image = pdf_document.extract_image(xref)
            image_bytes = base_image["image"]
            image_ext = base_image["ext"]
            image_filename = f"{output_dir}/image_{page_index + 1}_{img_index + 1}.{image_ex
            with open(image_filename, "wb") as image_file:
                image_file.write(image_bytes)
            print(f"Saved {image_filename}")

# Usage
extract_images('Page 4.pdf', 'folder')
```

```
Saved folder/image_1_1.jpeg
Saved folder/image_1_2.jpeg
Saved folder/image_1_3.jpeg
Saved folder/image_1_4.jpeg
```

```
# Creating Password Protected PDFs (Encrypted PDFs)

import PyPDF2

def encrypt_pdf(input_pdf, output_pdf, password):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        pdf_writer.add_page(pdf_reader.pages[page_num])

    pdf_writer.encrypt(password)

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"Encrypted PDF saved as {output_pdf}")

# Removing password from the PDF file (Decryption)

def decrypt_pdf(input_pdf, output_pdf, password):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_reader.decrypt(password)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        pdf_writer.add_page(pdf_reader.pages[page_num])

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"Decrypted PDF saved as {output_pdf}")

# Usage
encrypt_pdf('merged.pdf', 'encrypted.pdf', 'pass123')
decrypt_pdf('encrypted.pdf', 'decrypted.pdf', 'pass123')
```

```
Encrypted PDF saved as encrypted.pdf
Decrypted PDF saved as decrypted.pdf
```

```
# Rotating Pages - Landscape to Portrait

import PyPDF2

def rotate_pages(input_pdf, output_pdf, rotation):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        page = pdf_reader.pages[page_num]
        page.rotate(rotation)
        pdf_writer.add_page(page)

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"New PDF saved as {output_pdf}")

# Usage - Portait to Landscape
rotate_pages('merged.pdf', 'Landscape.pdf', 90)
```

```
New PDF saved as Landscape.pdf
```

```
# Rotating Pages - Potrait to Landscape

import PyPDF2

def rotate_pages(input_pdf, output_pdf, rotation):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        page = pdf_reader.pages[page_num]
        page.rotate(rotation)
        pdf_writer.add_page(page)

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"New PDF saved as {output_pdf}")

# Usage - Landscape to Portait (use angle either as -90 or 270)
rotate_pages('Landscape.pdf', 'Portrait.pdf', -90)
```

```
New PDF saved as Portrait.pdf
```

```
# Add Metadata to the PDF files (e.g. Title, Author, etc.)

import PyPDF2

def add_metadata(input_pdf, output_pdf, title, author):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in range(len(pdf_reader.pages)):
        pdf_writer.add_page(pdf_reader.pages[page_num])

    # Set the metadata directly using add_metadata
    metadata = {
        '/Title': title,
        '/Author': author,
        '/Producer': ''
    }
    pdf_writer.add_metadata(metadata)

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"PDF with added metadata saved as {output_pdf}")

# Usage
add_metadata('Portrait.pdf', 'metadata_added.pdf', 'Sample File for Python Coding', 'Dr. Raj

# Reading PDF Metadata

import PyPDF2

def read_metadata(pdf_file):
    pdf_reader = PyPDF2.PdfReader(pdf_file)
    metadata = pdf_reader.metadata

    print("Metadata of the PDF:")
    for key, value in metadata.items():
        print(f"{key}: {value}")

# Usage
read_metadata('metadata_added.pdf')
```

```
PDF with added metadata saved as metadata_added.pdf
Metadata of the PDF:
/Producer:
/Title: Sample File for Python Coding
/Author: Dr. Raj Gaurav Mishra
```

```
# Create a PDF directly from the text.

from reportlab.lib.pagesizes import letter
from reportlab.pdfgen import canvas

def create_pdf(output_pdf):
    c = canvas.Canvas(output_pdf, pagesize=letter)
    c.drawString(100, 750, "Hello, World!")
```

```
# Re-arranging pages in the PDF file.

import PyPDF2

def rearrange_pages(input_pdf, output_pdf, page_order):
    pdf_reader = PyPDF2.PdfReader(input_pdf)
    pdf_writer = PyPDF2.PdfWriter()

    for page_num in page_order:
        pdf_writer.add_page(pdf_reader.pages[page_num])

    with open(output_pdf, 'wb') as out:
        pdf_writer.write(out)
    print(f"Rearranged PDF saved as {output_pdf}")
```