Lightweight PDF Tool — Python Desktop Version

Overview

Goal: Build a small, local app for merging, annotating, and digitally signing PDFs — no server required.

Stack:
- Python ≥ 3.10
- PDF processing: pikepdf, PyMuPDF, pyHanko
- UI: PySide6 (Qt for Python)
- Packaging: PyInstaller

Phase 1 — Set up your environment

1. Install Python (≥ 3.10)
2. Create a project folder and virtual environment:

```
mkdir pdf_tool && cd pdf_tool
python -m venv venv
source venv/bin/activate (Windows: venv\Scripts\activate)
```

3. Install dependencies:

```
pip install pikepdf pymupdf pyHanko PySide6
```

4. Optional dev tools:

```
pip install black pylint pytest pyinstaller
```

Phase 2 — Build core functions

Merge PDFs (pikepdf):

```
from pikepdf import Pdf
def merge_pdfs(input_files, output_path):
pdf = Pdf.new()
for file in input_files:
src = Pdf.open(file)
pdf.pages.extend(src.pages)
pdf.save(output_path)
```

Annotate PDFs (PyMuPDF):

```
import fitz
def add_text_annotation(pdf_path, page_num, x, y, text, output_path):
doc = fitz.open(pdf_path)
page = doc[page_num]
page.add_text_annot((x, y), text)
doc.save(output_path)
```

Sign PDFs (pyHanko CLI):

```
pyhanko sign addsig --p12-file mycert.p12 --p12-password secret input.pdf -o signed.pdf
```

Phase 3 — Build GUI (PySide6)

Basic window example:

```
import sys
from PySide6.QtWidgets import QApplication, QMainWindow, QPushButton, QFileDialog,
QVBoxLayout, QWidget, QLabel
from merge_pdfs import merge_pdfs
class PDFTool(QMainWindow):
def __init__(self):
super().__init__()
self.setWindowTitle("PDF Tool")
layout = QVBoxLayout()
self.label = QLabel("Select PDFs to merge:")
```

```
layout.addWidget(self.label)
merge_button = QPushButton("Merge PDFs")
merge_button.clicked.connect(self.merge_pdfs)
layout.addWidget(merge_button)
container = QWidget()
container.setLayout(layout)
self.setCentralWidget(container)
def merge_pdfs(self):
files, _ = QFileDialog.getOpenFileNames(self, "Select PDF files", "", "PDF Files (*.pdf)")
if files:
out, _ = QFileDialog.getSaveFileName(self, "Save Merged PDF", "", "PDF Files (*.pdf)")
if out:
merge_pdfs(files, out)
self.label.setText(f"Merged {len(files)} files → {out}")
if __name__ == "__main__":
app = QApplication(sys.argv)
window = PDFTool()
window.show()
sys.exit(app.exec())
```

Phase 4 — Add Annotation Tools

Use PyMuPDF methods like:

```
page.add_highlight_annot(fitz.Rect(100, 100, 200, 120))
page.add_text_annot((150, 150), "Check this section")
```

Phase 5 — Add Digital Signing

Call pyHanko from Python:

```
import subprocess
def sign_pdf(input_pdf, cert_path, cert_password, output_pdf):
subprocess.run([
"pyhanko", "sign", "addsig",
"--p12-file", cert_path,
"--p12-password", cert_password,
input_pdf, "-o", output_pdf
])
```

Phase 6 — Package the App

Build a single-file executable:

```
pyinstaller --onefile --windowed main.py
```

Phase 7 — Testing

Test with various PDFs (multi-page, encrypted, forms).
Ensure compatibility with Acrobat and Preview.

Phase 8 — Security & Polish

- Keep all signing local (no server).
- Validate signatures with pyHanko.
- Add autosave, thumbnails, page reorder, custom signature image.

Optional Enhancements

- Page reorder UI (thumbnails with PyMuPDF)

- Annotation flattening
- Custom signature appearances
- Config file for settings