■ Documentação Técnica do Projeto: PDF_SUMMARIZER

■ Objetivo do Projeto

O PDF_SUMMARIZER é um aplicativo em Python que permite que o usuário envie um arquivo PDF, tenha o texto extraído, resumido e exibido em tela, e também baixe um novo PDF contendo o resumo formatado. A interface gráfica é construída com a biblioteca Gradio.

■ Estrutura de Arquivos

- app.py : Arquivo principal com interface Gradio - summarizer.py : Lógica para extrair e resumir texto - PDF_Downloader.py : Exportação do resumo para PDF formatado - preprocess.py : Limpeza e tokenização do texto - extractor.py : Alternativas para extração de texto (OCR, pdfplumber, etc.) - test_basic.py : Testes básicos

■■ Arquivo: app.py

Define a interface web com Gradio. Recebe o PDF, chama o resumo e oferece o download do novo arquivo.

```
import gradio as gr
from summarizer import summarize_pdf
from PDF_Downloader import salvar_pdf

def process_pdf(pdf_path: str):
    resumo = summarize_pdf(pdf_path)              # chama função de resumo
    out_name = "resumo.pdf"
    pdf_file = salvar_pdf(resumo, nome_arquivo=out_name)  # gera PDF resumido
    return resumo, pdf_file

with gr.Blocks() as demo:
    pdf_input = gr.File(file_types=[".pdf"], label="Escolha o PDF", type="filepath")
    resumo_output = gr.Markdown(label="Resumo")
    pdf_download = gr.File(label="Baixar PDF")
    btn = gr.Button("Resumir")
    btn.click(fn=process_pdf, inputs=pdf_input, outputs=[resumo_output, pdf_download])

demo.launch()
```

■ Arquivo: summarizer.py

Extrai o texto de um PDF usando a biblioteca pypdf e aplica um resumo simples.

```
from pypdf import PdfReader

def summarize_pdf(pdf_path: str) -> str:
    reader = PdfReader(pdf_path)  # abre o PDF
    text = ""
    for page in reader.pages:     # percorre cada página
        text += page.extract_text() + "\n"
    return text[:1000] + "..."    # retorna resumo simples
```

■ Arquivo: PDF_Downloader.py

Gera um PDF formatado com o resumo usando a biblioteca reportlab.

```
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer
from reportlab.lib.pagesizes import A4
from reportlab.lib.styles import getSampleStyleSheet

def salvar_pdf(texto: str, nome_arquivo: str = "resumo.pdf") -> str:
    doc = SimpleDocTemplate(nome_arquivo, pagesize=A4)
    styles = getSampleStyleSheet()
```

```
    story = []
    for para in texto.split("\n\n"):     # divide em parágrafos
        story.append(Paragraph(para, styles["Normal"]))
        story.append(Spacer(1, 12))
    doc.build(story)
    return nome_arquivo
```

■ Arquivo: preprocess.py

Limpa o texto extraído antes de resumir.

```
def clean_text(text: str) -> str:
    text = text.replace("\n", " ")     # remove quebras de linha
    text = " ".join(text.split())       # remove espaços extras
    return text
```

■ Arquivo: test_basic.py

Valida que o resumo não retorna vazio.

```
from summarizer import summarize_pdf

def test_summarizer():
    resumo = summarize_pdf("exemplo.pdf")
    assert resumo != ""
```

■ Fluxo Geral do Projeto

1. Usuário acessa a interface (app.py). 2. Faz upload de um PDF. 3. O summarizer.py extrai e resume o texto. 4. O preprocess.py limpa o conteúdo. 5. O resumo é exibido na tela. 6. O PDF_Downloader.py gera o arquivo resumido para download.