Plagiarism Report

Your Input:

```
import os
from flask import Flask, render_template, request, send_file
from werkzeug.utils import secure_filename

from utils.pdf_reader import extract_text_from_pdf
from utils.preprocess import preprocess_text
from utils.similarity import compute_similarities, highlight_matches

# PDF generation
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer
from reportlab.lib.styles import getSampleStyleSheet
from reportlab.lib.pagesizes import letter

BASE_DIR = os.path.dirname(os.path.abspath(__file__))
UPLOAD_FOLDER = os.path.join(BASE_DIR, 'uploads')
DOCUMENTS_FOLDER = os.path.join(BASE_DIR, 'documents')
ALLOWED_EXTENSIONS = {'txt', 'pdf'}

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024

os.makedirs(UPLOAD_FOLDER, exist_ok=True)
os.makedirs(DOCUMENTS_FOLDER, exist_ok=True)

# Global variables for PDF report
input_text_global = ""
similarities_global = []


def allowed_file(filename):
return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


@app.route('/', methods=['GET', 'POST'])
def index():
global input_text_global, similarities_global
```

```
if request.method == 'POST':
input_text = request.form.get('input_text', '').strip()
uploaded_file = request.files.get('file')

# File Upload
if uploaded_file and uploaded_file.filename != "" and allowed_file(uploaded_file.filename):
filename = secure_filename(uploaded_file.filename)
filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
uploaded_file.save(filepath)

if filename.lower().endswith(".pdf"):
input_text = extract_text_from_pdf(filepath)
else:
with open(filepath, "r", encoding="utf-8", errors="ignore") as f:
input_text = f.read()

if not input_text:
return render_template("index.html", error="Please enter text or upload a TXT/PDF file.")

input_text_global = input_text

# Preprocess input
clean_input = preprocess_text(input_text)

# Load reference docs
doc_texts = []
doc_names = []
for fname in os.listdir(DOCUMENTS_FOLDER):
path = os.path.join(DOCUMENTS_FOLDER, fname)
if os.path.isfile(path) and fname.endswith(".txt"):
with open(path, "r", encoding="utf-8", errors="ignore") as f:
raw = f.read()
doc_texts.append(preprocess_text(raw))
doc_names.append(fname)

# Similarity scores
similarities, top_matches = compute_similarities(clean_input, doc_texts, doc_names)
similarities_global = similarities

# Highlight matches
highlights = []
```

```
for match in top_matches:
idx = match["index"]
raw_path = os.path.join(DOCUMENTS_FOLDER, doc_names[idx])

with open(raw_path, "r", encoding="utf-8", errors="ignore") as f:
raw_doc = f.read()

highlighted = highlight_matches(input_text, raw_doc)

highlights.append({
"doc": doc_names[idx],
"score": match["score"],
"highlighted": highlighted
})

return render_template(
"result.html",
input_text=input_text,
similarities=similarities,
highlights=highlights
)

return render_template("index.html")


@app.route("/download_pdf")
def download_pdf():
file_path = "report.pdf"

styles = getSampleStyleSheet()
story = []

# Title
story.append(Paragraph("Plagiarism Report", styles["Title"]))
story.append(Spacer(1, 14))

# Input Section
story.append(Paragraph("Your Input:", styles["Heading2"]))
for line in input_text_global.split("\n"):
story.append(Paragraph(line, styles["Normal"]))
story.append(Spacer(1, 6))
story.append(Spacer(1, 14))

# Similarity Scores Section
story.append(Paragraph("Similarity Scores:", styles["Heading2"]))
for s in similarities_global:
story.append(Paragraph(f"{s['doc']} — {round(s['score'] * 100, 2)}%", styles["Normal"]))
story.append(Spacer(1, 6))

story.append(Spacer(1, 14))

# Build PDF
doc = SimpleDocTemplate(file_path, pagesize=letter)
doc.build(story)

return send_file(file_path, as_attachment=True)


if __name__ == "__main__":
app.run(debug=True)
```

Similarity Scores:

doc2.txt — 2.01%

doc3.txt — 0.0%

doc1.txt — 0.0%