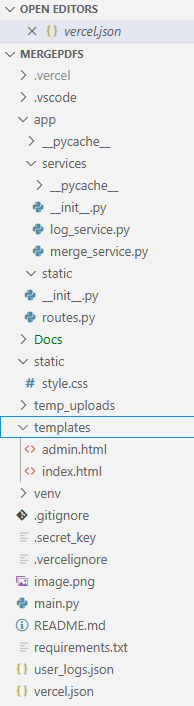
**Directory Structure for Merge-PDFs inside VS Code workspace**



**Inside app/Services**

**\_\_init\_\_.py (file)**

Blank (No Code here inside it)

**log\_service.py (file)**

from user\_agents import parse as ua\_parse

import json

import os

def log\_user\_info(req):

    ip = req.remote\_addr

    user\_agent\_str = req.headers.get('User-Agent', '')

    ua = ua\_parse(user\_agent\_str)

    log = {

        'ip': ip,

        'device': ua.device.family,

        'os': ua.os.family,

        'browser': ua.browser.family

    }

    log\_path = os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), '..', '..', 'user\_logs.json'))

    logs = []

    if os.path.exists(log\_path):

        with open(log\_path, 'r') as f:

            try:

                logs = json.load(f)

            except json.JSONDecodeError:

                logs = []

    logs.append(log)

    with open(log\_path, 'w') as f:

        json.dump(logs, f, indent=2)

**merge\_service.py (file)**

from pypdf import PdfReader, PdfWriter

from io import BytesIO

def merge\_pdfs(files):

    writer = PdfWriter()

    for file in files:

        if not file.filename.endswith('.pdf') or file.content\_length == 0:

            raise ValueError(f"Invalid or empty file: {file.filename}")

        reader = PdfReader(file.stream)  # ✅ Use .stream here

        for page in reader.pages:

            writer.add\_page(page)

    output\_stream = BytesIO()

    writer.write(output\_stream)

    output\_stream.seek(0)

    return output\_stream

**Inside Project root/app (directory)**

**\_\_init\_\_.py (file)**

import os

from flask import Flask

from app.routes import register\_routes

def create\_app():

    template\_dir = os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), '..', 'templates'))

    app = Flask(\_\_name\_\_, template\_folder=template\_dir)

    # Load secret key from environment

    app.secret\_key = os.environ.get('SECRET\_KEY', 'fallback-key-for-local-dev')

    register\_routes(app)

    return app

**routes.py (file)**

from flask import render\_template, request, send\_file, flash, redirect, url\_for

from app.services.merge\_service import merge\_pdfs

from app.services.log\_service import log\_user\_info

import traceback

def register\_routes(app):

    @app.route('/', methods=['GET', 'POST'])

    def home():

        if request.method == 'GET':

            return render\_template('index.html')

        try:

            log\_user\_info(request)

            pdfs = request.files.getlist('pdfs')

            if not pdfs or not all(f.filename.endswith('.pdf') and f.content\_length > 0 for f in pdfs):

                flash("Please upload valid PDF files only.")

                return redirect(url\_for('home'))

            output\_stream = merge\_pdfs(pdfs)

            return send\_file(

                output\_stream,

                as\_attachment=True,

                download\_name='merged.pdf',

                mimetype='application/pdf'

            )

        except Exception as e:

            print("Error:", e)

            traceback.print\_exc()

            flash("Something went wrong while merging. Please try again.")

            return redirect(url\_for('home'))

    @app.route('/admin', methods=['GET'])

    def admin():

        return render\_template('admin.html')

**Inside Project Root / static (Directory)**

style.css (file)

blank (no stylesheet code inside this file)

**Inside Project Root / templates (Directory)**

**admin.html (file)**

<!DOCTYPE html>

<html>

<head><title>Admin Dashboard</title></head>

<body>

  <h1>User Log</h1>

  <ul>

    {% for log in logs %}

      <li>{{ log.timestamp }} - {{ log.ip }} - {{ log.device }} - {{ log.os }} - {{ log.browser }}</li>

    {% endfor %}

  </ul>

</body>

</html>

**index.html (file)**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Merge PDFs</title>

</head>

<body>

  <h2>Merge PDFs</h2>

  {% with messages = get\_flashed\_messages() %}

  {% if messages %}

    <ul style="color: red;">

      {% for msg in messages %}

        <li>{{ msg }}</li>

      {% endfor %}

    </ul>

  {% endif %}

{% endwith %}

  <form action="/" method="POST" enctype="multipart/form-data">

    <input type="file" name="pdfs" multiple accept="application/pdf" required>

    <button type="submit">Merge & Download</button>

  </form>

</body>

</html>

**.verelignore (file inside project root)**

venv/

.vscode/

\*.pyc

\_\_pycache\_\_/

user\_logs.json

image.png

\*.log

temp\_uploads/

**main.py (file inside project root)**

from app import create\_app

app = create\_app()

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

**requirements.txt (file inside project roo)**

Flask==3.1.1

pypdf==5.8.0

user-agents==2.2.0

**user\_logs.json (file inside project roo)**

[

{

"ip": "127.0.0.1",

"device": "Other",

"os": "Windows",

"browser": "Edge"

},

{

"ip": "127.0.0.1",

"device": "Other",

"os": "Windows",

"browser": "Edge"

}

**vercel.json (file inside project root)**

{

  "builds": [

    {

      "src": "main.py",

      "use": "@vercel/python"

    }

  ],

  "routes": [

    {

      "src": "/(.\*)",

      "dest": "main.py"

    }

  ]

}