JobFit – Development Diary

(Day 1: Project Setup & File Upload)

# Project Overview

An AI-based job matching tool that takes a resume and a job description as input, extracts text, compares them, and provides a match score.

# Day 1 Progress (21 July 2025)

## 1. Flask Setup & Basic Server

Installed Flask:

pip install flask

Basic server (app.py):

from flask import Flask, render\_template  
  
app = Flask(\_\_name\_\_)  
  
@app.route("/")  
def home():  
 return render\_template("index.html")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(debug=True)

## 2. Template Rendering (index.html)

• All HTML files are stored in a templates/ folder (Flask rule).  
• render\_template("index.html") loads the file into the browser.  
• By default, this uses the GET method.

## 3. GET vs POST (Understanding)

|  |  |  |
| --- | --- | --- |
| Method | When it Triggers | What Happened in Our Project |
| GET | When you just open the page | Only shows the HTML form (no processing) |
| POST | When you click Submit | Sends files to Flask → Flask processes & saves |

Allowed methods in route:

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

## 4. HTML Form Setup (File Upload)

We designed a Bootstrap-based form in index.html:

<form method="POST" enctype="multipart/form-data">  
 <input type="file" name="resume" required>  
 <input type="file" name="job" required>  
 <button type="submit">Submit</button>  
</form>

Important Points:  
• method="POST" → required to send files to the server.  
• enctype="multipart/form-data" → required for file uploads.  
• name="resume"/"job" → Flask uses these names in request.files.

## 5. File Upload Handling in Flask

Updated app.py:

from flask import Flask, render\_template, request  
import os  
  
app = Flask(\_\_name\_\_)  
UPLOAD\_FOLDER = "upload"  
os.makedirs(UPLOAD\_FOLDER, exist\_ok=True)  
  
@app.route("/", methods=["GET", "POST"])  
def home():  
 if request.method == "POST":  
 resume = request.files.get("resume")  
 job = request.files.get("job")  
  
 if resume:  
 resume\_path = os.path.join(UPLOAD\_FOLDER, resume.filename)  
 resume.save(resume\_path)  
 print("Resume saved:", resume\_path)  
  
 if job:  
 job\_path = os.path.join(UPLOAD\_FOLDER, job.filename)  
 job.save(job\_path)  
 print("Job Description saved:", job\_path)  
  
 return render\_template("index.html")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(debug=True)

## 6. Folder Structure (Current)

ai\_job\_maker/  
│  
├── app.py  
├── templates/  
│ └── index.html  
├── static/  
│ ├── css/style.css  
│ └── js/script.js  
└── upload/ (saved files here)

## 7. Current Status

✅ HTML Form Working  
✅ Flask GET & POST Working  
✅ File Uploads Saving Successfully

## 8. Next Steps (Phase 3)

• Write a function extract\_text(file\_path) to extract text from PDF/DOCX/TXT.  
• Print extracted text in the console for testing.  
• Required libraries:  
 pip install PyPDF2 docx2txt