**JobFit Resume**

**Objective**

JobFit Resume aims to address this issue of users manually tailoring the resume for every job application which is prone to human errors, by creating a resume and cover letter that aligns with the job description. The primary goal of the JobFit Resume project is to save time for job seekers, enhance the quality of their applications, and ultimately improve their chances of securing interviews in a challenging job market.

**Methodology**

*User Profile*

Firstly, we aimed to extract user details from the resume provided by the user. The process started with parsing the resume and converting its content into a readable text format. Once we had the text, we utilized a large language model (LLM), specifically Llama 3.2 70B, to analyse the user profile and extract specific information. The extraction process was guided by a predefined format, where we clearly outlined the expected structure of the output, such as a JSON dictionary containing keys like name, contact details, and other relevant fields. Despite specifying the exact output format in the 1-shot prompt, the LLM often added extra details like "Here is the resume" or included additional commentary, deviating from the strict JSON structure we required. To address this, we reprocessed the output using the LLM itself, focusing on isolating and extracting the JSON content accurately. This iterative approach involved multiple attempts, and we observed that after 2 to 3 trials, the LLM was generally able to produce the correct content in the desired format.

To ensure reliability, we implemented a mechanism to retry up to a maximum of 5 attempts, which allowed us to consistently achieve the correct JSON output. Once the final JSON structure was obtained, we stored it in a JSON file for further processing or integration. This iterative refinement process helped us overcome the initial inconsistencies and ensured accurate extraction of user details from resumes.

*Job Description*

The user inputs the JD, which is then analysed using the LLM. The goal is to extract key details such as company name, role, required skills, preferred qualifications, experiences mentioned in the job role. Similar to the method used for resumes, we leverage the LLM to identify and structure this information into a predefined JSON format. The format includes keys like keywords, experience, and requirements, ensuring the output is structured and easy to use.

**Customized Resume**

*Skills*

The main task is to produce a customized resume tailored to the job description (JD), starting with the skills section. To achieve this, we first extract the user’s skills from the original resume JSON and provide only these skills along with the job description to the LLM. The prompt for the LLM is carefully crafted to emphasize its role as an expert in optimizing resumes. The instructions specify the following:

Highlight Relevant Skills: The LLM must prioritize and highlight skills present in the JD. These are the skills that align with the job requirements and increase the candidate's chances of selection.

Remove Irrelevant Skills: Any skills not pertinent to the JD are considered unnecessary noise and should be omitted. This ensures the resume remains focused and readable, presenting only what matters to the recruiter.

Avoid Adding Unfamiliar or Misleading Skills: The LLM is explicitly instructed not to include skills that the candidate does not possess or cannot reasonably demonstrate. This avoids creating an inaccurate representation of the candidate's abilities.

Flexibility with Similar Skills: If the user has skills that are similar to those mentioned in the JD, the LLM has the flexibility to adapt and include them to align better with the role’s requirements.

Soft and Miscellaneous Skills: Any relevant soft skills or miscellaneous skills mentioned in the JD, but not explicitly listed in the original resume, can be added if they fit naturally into the skills section. This ensures the skills section is comprehensive and aligned with the JD.

With these instructions, the LLM processes the input and generates an updated, polished skills section for the resume. This approach ensures the skills section is highly targeted, concise, and relevant, effectively tailored to the specific job description.

*Experiences and Projects*

A similar methodology was followed for optimizing the experience and project sections of the resume. Using the JSON file as the source, we extracted the values of the respective keys (experience and projects). Along with the extracted content and the job description (JD), we crafted a specific prompt to guide the LLM in generating optimized content. The prompt was structured to position the LLM as an expert in writing professional resumes. The key instructions provided were:

Professional Tone with Action Verbs: The LLM was instructed to rewrite the content starting with impactful action verbs, ensuring the descriptions are professional and compelling.

Follow the STAR Method: The content for each project and experience was to follow the STAR method (Situation, Task, Action, Result). This approach ensures clarity and effectively communicates the candidate's accomplishments and contributions.

Quantify Achievements: The LLM was asked to quantify achievements wherever possible by incorporating metrics, percentages, or numbers. This helps to add credibility and impact to the content.

Maintain Integrity: A strict instruction was given to avoid hallucination or the addition of any new projects, roles, or metrics not present in the candidate’s original profile. The LLM was allowed only to enhance or modify the existing content, not to make up new details.

Emphasize JD-Relevant Details: From the given content, the LLM was tasked to reframe and emphasize elements that resonate more strongly with the JD, making the experience and projects sections as relevant and impactful as possible.

By applying this methodology, we successfully refined the skills, experience, and projects sections of the resume. These three critical sections were optimized to create a customized and professional resume tailored specifically to the job description, ensuring both relevance and authenticity.

*Generating latex file*

To finalize the process, we ensured the updated resume in JSON format could be converted into a PDF using LaTeX. A customized LaTeX .cls file was designed to define the structure and style of the resume. This file serves as a template applicable to all resumes and includes the placeholders for all major sections (e.g., Skills, Experience, Projects) and their respective subsections. We also predefined formatting rules, such as font styles, sizes, alignment, spacing, and bullet-point styles. The placeholders are dynamically populated with content extracted from the JSON file, ensuring consistency and readability.

A Python script was developed to automate the conversion of optimized resume details from a JSON file into a professionally styled PDF. The script reads the updated JSON file, extracts the content from various sections (e.g., experiences, projects, skills), and processes it for compatibility with LaTeX. Special characters and syntax that could cause errors in LaTeX (e.g., &, %, #, \_) are identified and escaped during processing. The cleaned and formatted content is then written into a .tex file using pre-defined functions from a LaTeX .cls template file. Finally, the script invokes the LaTeX compiler ( pdflatex) to generate a polished PDF resume, ensuring a professional appearance suitable for use.

By following this approach, we automated the process of transforming a JSON-formatted resume into a visually appealing PDF document, leveraging the power of LaTeX for precise formatting and presentation.

**Cover Letter**

To generate the cover letter, user profile and JD is provided to the LLM and prompted it to create a three-paragraph structure. In the first paragraph, the LLM was instructed to describe the candidate's background and skills. This includes introducing the candidate, their qualifications, and key abilities that make them suitable for the role. In the second paragraph, the LLM highlights the most relevant sections of the candidate’s profile that align with the JD. This involves emphasizing specific experiences, projects, and achievements, connecting them to the company's requirements. The goal is to demonstrate how the candidate's work and skills can add value to the organization.

In the third paragraph, the LLM expresses the candidate's interest in the company and the role. It includes reasons for wanting to join the organization, mentioning specific aspects of the company that resonate with the candidate. This paragraph concludes by expressing enthusiasm for discussing their profile further and thanking the employer for considering their application. The output text is covered to pdf format, which can be later downloaded

**ChatBot Integration**

When filling out job applications, candidates are often required to answer additional company-specific questions, such as "Why are you interested in this role?" or "Describe a project where you worked in a team." Customizing these responses for each application can be time-consuming and tedious. To streamline this process, we developed a chatbot that acts as a job application assistant. This chatbot leverages the user's profile, including their resume and key experiences, to craft tailored responses to these questions efficiently. It analyses the question, aligns the answers with the candidate's profile, and ensures relevance to the specific job or company to address the challenge of creating a seamless and dynamic interaction experience, we implemented a chatbot that uses Langraph to manage conversational state effectively. Since the LLM operates in a stateless mode and does not retain the memory of previous responses, we configured the chatbot with Langraph to save the state of each conversation. With this setup, after every interaction, the chatbot stores the context and state of the conversation. When the user asks a follow-up question, Langraph helps the chatbot identify the most recent or relevant state of the previous interaction and continues. This ensures the responses remain consistent and contextually accurate, allowing for a more dynamic and coherent conversational flow.

By providing personalized and contextually appropriate answers, the chatbot saves significant time while maintaining quality and coherence in responses. It assists users with any profile-related queries, making the application process smoother and more effective, while reducing the burden of repetitive customization. The code and steps to run it are given <https://github.com/JanvitaReddy11/JobFit-Resume>

**Steps**

1] The present working directory should be the folder name LLM\_Project

2] Install dependencies: pip install -r requirements.txt

3] Create a dot env file that contains the groq api key

4] Run the command: streamlit run app.py

5] Upload the resume and give the job description as input

**Note**

To compile the latex file to pdf, we should have latex distribution installed on our desktop

*For Mac Users:*

Install MacTex (<https://www.tug.org/mactex/>)

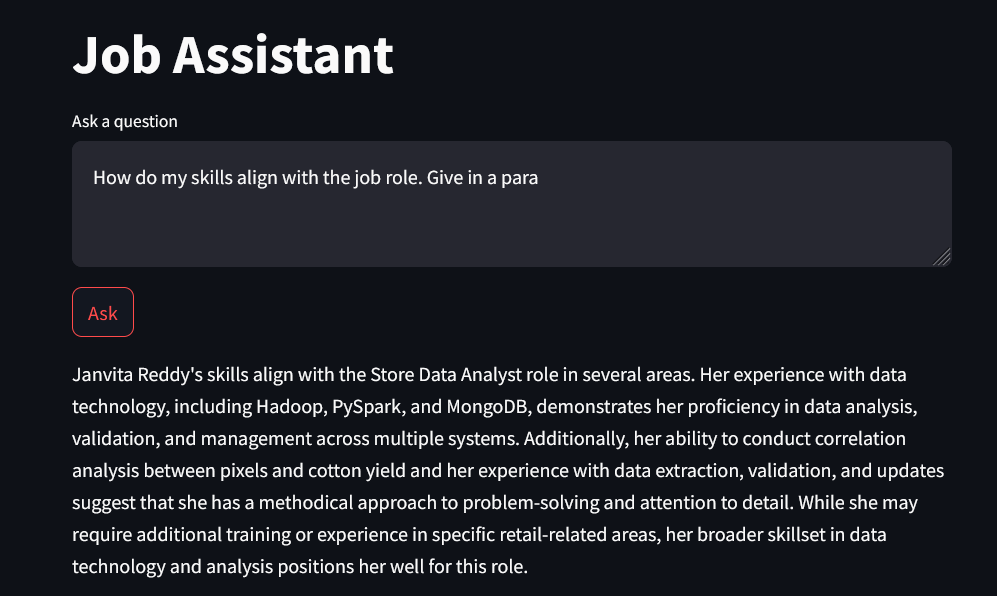
Install a text editor like TeXShop

*For window users:*

Install MiKTeX (<https://miktex.org/download>)

Install a Text Editor like TeXstudio





**Challenges**

1] The resume format was predefined to extract only specific sections such as contact details, skills, work experience, and projects. However, it failed to dynamically recognize and include other sections like publications, certifications, or achievements when they were present in the resume. This limitation required additional manual intervention in the extraction logic.

2] When the uploaded resume lacked information required by the predefined format, the algorithm returned None for those fields. This led to incomplete updated resumes, as these cases were not handled effectively. Developing a fallback mechanism or default handling for such scenarios remains a key area for improvement.

3] Extracting job descriptions from URLs proved challenging due to inconsistent webpage structures or access issues. As a result, the algorithm currently accepts job descriptions only in text format.

4] Converting the updated JSON into a LaTeX format initially posed issues, particularly with understanding the LaTeX workflow and ensuring compliance with its syntax rules. Special characters like &, %, and \_ needed to be properly escaped to avoid compilation errors. Additionally, installing and configuring the necessary tools to compile the .tex file into a PDF required effort and troubleshooting.

5] Processing large resumes, and job descriptions, or handling iterative LLM queries introduced noticeable delays. Improvements are still required to optimize the workflow to reduce response time