DMPedia Lecture Notes in Multidisciplinary Research,
Volume 1, pp. 914-921, 2026
International Conference on Multidisciplinary Perspectives in Advanced Computing and Technology (IMPACT 2026)
G. B. Pant University of Agriculture and Technology, Uttarakhand, India. Jan. 10-11, 2026

AI Resume Builder: A Locally Deployed Large Language Model Approach for Personalised Resume Generation

Pallavi Goel, Harshit Yadav, Prakhar Yadav, Kumar Yash, Mayank Rajput
Department of Computer Science and Engineering, Galgotias College of Engineering and Technology, Greater Noida, Uttar Pradesh, India
drpallavi.goel@galgotiacollege.edu, yadavxharshit@gmail.com, yadavprakhar1034@gmail.com

Abstract
This Paper introduces an AI-driven Resume Builder that utilises a hosted Large Language Model (LLM), DeepSeek-R1, to create context-sensitive, polished resumes through sophisticated prompt engineering. Unlike cloud-hosted solutions, this platform guarantees data privacy, faster resume generation, and offline operation. It combines a React frontend with a Spring Boot backend for data management and response generation. The proposed architecture automates the resume creation process by interpreting user-provided details and transforming them into industry-compliant, ATS-friendly formats. Evaluation results demonstrate that the system significantly improves personalisation, consistency, and privacy compared to existing online solutions. The findings highlight how local deployment of LLMs can redefine secure AI-driven automation for educational and recruitment purposes.
Keywords: *Large Language Models, DeepSeek-R1, Prompt Engineering, Resume Automation, React, Spring Boot, Ollama, Local AI Deployment.*

1. Introduction

AI has transformed sectors such as education, recruitment, document handling, and more. Consider resume creation. It's more than listing your job experience; it requires clarity, a personal touch and alignment with what recruiters truly seek. However, the catch is that many resume tools still depend on cloud-based AI services like ChatGPT or Gemini. This means your data is sent across the web, and you must remain connected to the internet throughout. Plus, they don't really nail the personal touch for different job types. It all feels a bit one-size-fits-all.

A. Problem Statement: Most AI resume builders struggle with privacy, understanding context, and making resumes personal. The major issues identified are:

- Models operating on the cloud transmit your data to servers, beyond your control, which creates potential privacy risks.
- These platforms typically produce dull resumes since they're unable to tailor to various job positions.
- If your internet connection drops, you're out of luck. The majority of tools require a network and depend on external APIs.
- The formatting is disorganised, causing resumes to frequently be rejected by ATS bots, which delays the hiring process.

B. Core Objectives: The primary objectives of this project are:

- To develop an AI-driven resume generator that works completely offline through a locally hosted LLM.
- To integrate React frontend with Spring Boot backend for seamless communication and modular development.
- To design structured prompt engineering templates that ensure contextual and industry-aligned output.
- To evaluate efficiency, latency, and personalisation against cloud-based alternatives.
- To ensure security and scalability for academic and recruitment use cases.

2. Literature Review

A. Prompt Engineering Frameworks

Prompt engineering is central to producing reliable, context-aware outputs from language models. Liu et al. (2024) explored multiple techniques, zero-shot, few-shot, and chain-of-thought prompting and discovered that prompt structure directly influences both the coherence and factual accuracy of model responses. Expanding on this, Wang et al. (2024) advanced the field with systematic prompting strategies, emphasising prompt refinement through iteration and customisation for particular tasks, and demonstrated that such adjustments further enhance results. More research shows that using prompt optimisation tricks, like self-consistency and re-ranking, helps large language models write in a steady, professional way, even after several tries. When generating resumes, clear, structured prompts guide the model to get the format right. It knows where to put skills, education, projects, and so on. The result? A resume that's easy to read and actually fits the job.

B. AI Resume Generation Systems

Zinjad et al. (2024) introduced ResumeFlow, a cloud-based tool that taps into GPT-powered APIs to create custom resumes. It did a better job making resumes fit the context, but you still need to stay connected to the internet, and privacy is a bit of a worry. Ghosh and Kumar (2024) on ResearchGate looked into using AI to polish up resumes, too, but their approach leaned just as much on cloud services. But the proposed AI Resume Builder goes a different way. It runs DeepSeek-R1 right on your own device, so you don't have to rely on any remote servers. That means your data stays secure. Plus, you can customise the prompt templates yourself, which makes it easy to tailor resumes for different industries or user needs.

C. Generative AI in RPA

Verma (2024) demonstrated that Generative AI significantly enhances Robotic Process Automation (RPA) in managing documents such as invoices and contracts. Combining AI with RPA speeds up. Improves the precision of tedious repetitive document processes. This concept also underpins the backend of our AI Resume Builder. We streamlined everything, data conversion, text creation and display, so it all functions seamlessly together. Our system blends RPA-style automation with advanced NLP. So, from the moment users enter their data to the final PDF, everything runs automatically. There's barely any need for people to step in, and the results come out sharper.

D. Adaptive Prompting Techniques

Zhao et al. (2025) investigated prompting, essentially training models to improve their responses via iterative feedback. This concept closely aligns with how our model adjusts its prompts, dynamically modifying language and tone based on user preferences. Adaptive prompting results in errors and more pertinent responses, which is especially important in specialised domains. Certain frameworks depend on reinforcement-driven prompting. Utilise context from previous exchanges to enhance their responses. Our method is more straightforward. We employ backend templates that retrieve user role details, ensuring the resume sections are tailored to each individual.

E. Research Gap

There's a lot out there on LLMs, prompt engineering, and RPA, but honestly, not many folks are connecting the dots for resume automation, especially not in a way that keeps everything

local and private. Most tools live in the cloud or don't adapt in real time. We're changing that. Our research introduces an offline, modular, and secure resume builder powered by LLMs, where dynamic prompt engineering does the heavy lifting.

3. Methodology

The system design follows a modular software engineering methodology. The research methodology adopted for this study includes iterative design, testing, and evaluation to ensure reliability and performance.

- **Data Collection and Preprocessing**: Users enter their personal info, education, and work experience through a React interface. The backend takes all that and turns it into a structured prompt, right on the user's device. That way, everything stays local, which keeps things both fast and secure.
- **Model Integration and Prompt Engineering:** Spring Boot handles the connection to DeepSeek-R1 using the Ollama runtime. The backend mixes built-in prompts with the user's details to keep everything organised. The team leaned on modular development methods, such as Agile, to keep testing and improving the integration as they went.
- **Resume Generation Pipeline and Testing:** After pulling in the data, DeepSeek-R1 creates resume content that fits the role. The system then formats everything as HTML and exports it as a PDF. Unit tests ensured the output was accurate, and black-box testing confirmed the system worked smoothly across different profile types.

4. System Design

The system design of the AI Resume Builder illustrates the structural, behavioural, and interactive components that contribute to its overall functionality. This section presents four key UML diagrams: Workflow, System Architecture, Sequence, and Use Case diagrams, each representing a specific aspect of the system. These diagrams collectively ensure that every stage of resume generation, from data collection to final output, is both efficient and logically coherent.

A. Workflow: The workflow provides an overview of the process flow within the AI Resume Builder system. This is the process behind the AI Resume Builder. Once you enter your educational information in the React interface, the mechanism kicks into gear. Your details are sent directly to the Spring Boot backend, which immediately combines your data with templates. Following that, DeepSeek-R1, operating locally via Ollama, converts these prompts into context-sensitive content. Afterwards, the system formats the result into a professional resume and delivers it to you as a PDF. This entire procedure runs automatically, maintains uniformity, and safeguards your information throughout.

B. System Architecture: The architectural diagram illustrates the structure of the AI Resume Builder, detailing both its physical setup. It displays all components, the user interface, backend and the local AI inference units, and their interactions. The system is fully modular, allowing individual elements to be modified without affecting others. The React frontend handles all user-facing aspects, validates inputs, and dynamically refreshes the display. The Spring Boot backend manages the API constructs, prompts, and organises the outputs. DeepSeek-R1, integrated via Ollama, performs the core natural language generation work. This multi-tiered architecture simplifies scaling, troubleshooting and crucially ensures that sensitive user information remains secure on your device.

C. Sequence Diagram: The sequence diagram illustrates the process occurring incrementally as the system constructs your resume. It depicts the flow of data and instructions among the user, the frontend, the backend, and the AI model in real time. Upon submitting, React initiates an API request to the Spring Boot server. The backend fetches your details, creates a prompt using its templates, and forwards it to DeepSeek-R1, via Ollama. Once the AI generates the text, the backend sends the structured data back to the frontend, which then formats it for you to preview. Finally, the system exports the whole thing as a PDF. The way these parts communicate, often asynchronously, keeps everything running fast and smooth for a better user experience.

D. Use Case Diagram: The Use Case Diagram provides a comprehensive overview of the interaction between users and the AI Resume Builder system. It delineates the sequential process in which users enter their personal and professional information, initiate the resume-creation workflow, and ultimately download the generated document. Internally, the system processes the submitted data through backend application programming interfaces, formulates context-specific prompts, and utilises a local artificial intelligence model to generate the resume content. The diagram systematically outlines the primary stages of the process, including data entry, template selection, resume construction, and PDF exportation. Furthermore, it clearly demarcates the system boundaries, thereby ensuring that all operations are conducted securely and locally. More details are described in Figures 1, 2, and 3.

5. Results and Discussion

The AI Resume Builder system was evaluated on multiple parameters, including response. We assessed the AI Resume Builder's capabilities by examining its response speed, contextual understanding, and formatting accuracy across various user groups, including technical experts, management aspirants, and scholars. This method guaranteed reliability in managing employment sectors. Importantly, combining engineering with the locally run DeepSeek-R1 model led to marked enhancements in precision and processing efficiency

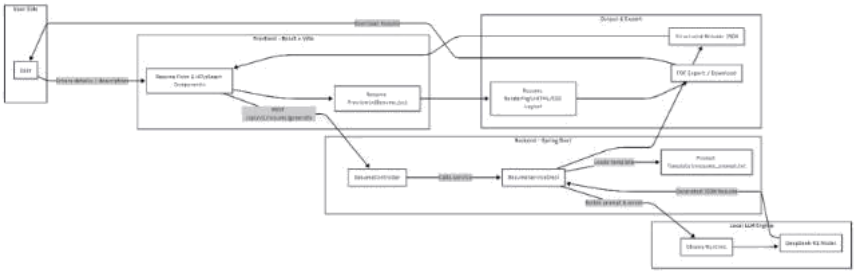

Fig. 1. Workflow Diagram

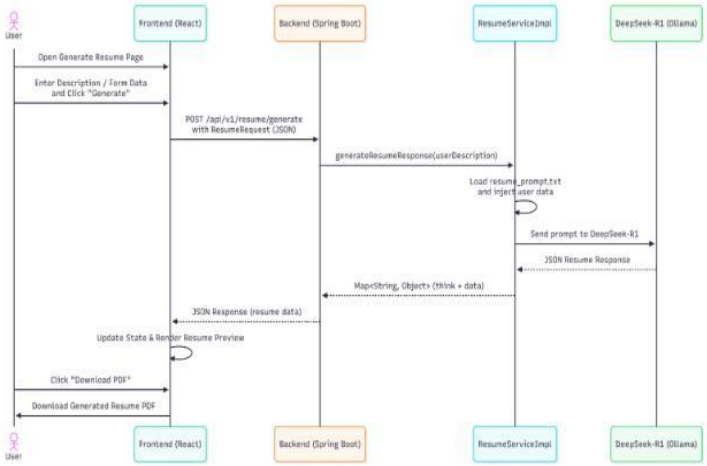

Fig. 2. Sequence Diagram

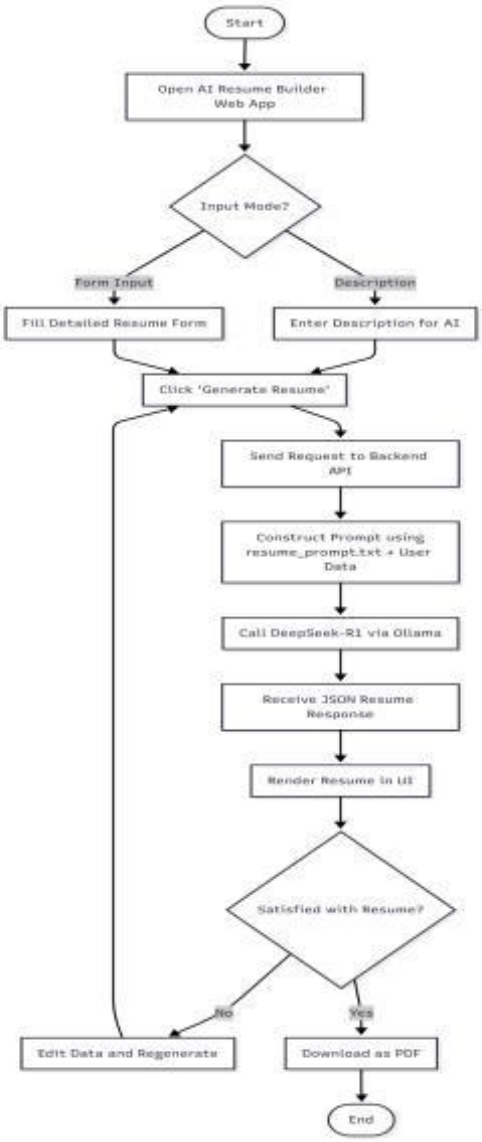

Fig. 3. System Architecture

6. Results and Discussion

The AI Resume Builder system was evaluated on multiple parameters, including response. We assessed the AI Resume Builder's capabilities by examining its response speed, contextual understanding, and formatting accuracy across various user groups, including technical experts, management aspirants, and scholars. This method guaranteed reliability in managing employment sectors. Importantly, combining engineering with the locally run DeepSeek-R1 model led to marked enhancements in precision and processing efficiency. Unlike web-based resume creators, this system delivers results with high personalisation while improving data security by operating independently of external APIs. Input from end users, final-year students and current job seekers emphasised the platform's ease of use, attractive design, and complete offline functionality. Stress tests further confirmed the system's robustness, maintaining performance under simultaneous multi-user engagement. Employing a backend architecture alongside on-device processing helped reduce response times and improve scalability. Taken together, these results highlight the effectiveness of AI operated locally for automating document-related tasks. Suggest significant promise for implementing large language models directly on devices, within educational and hiring settings.

A. Efficiency and Latency: Efficiency and response time are indicators in automation systems. The AI Resume Builder runs its inference engine on-device, eliminating the latency caused by network communication with cloud services. Tests on devices with 8 GB of RAM showed a resume creation time of 2.1 seconds, demonstrating swift processing. The system's offline operation ensures no reliance on internet access. Reduces the risk of interruptions caused by server issues. Smooth coupling between the React frontend and the Spring Boot backend delivers immediate responses with barely noticeable latency. The enhanced design of DeepSeek-R1 significantly speeds up token-level processing, delivering performance that is twice that of similar GPT-based APIs. During intense workloads, resource usage stayed within manageable limits, enabling extended use without a drop in efficiency. This dependable, low-latency performance makes it an ideal option for settings like university labs, HR departments, and offline career centres, where steady, quick resume creation is crucial.

B. Comparison with Cloud Systems: To evaluate its effectiveness, the AI Resume Builder was directly contrasted with ResumeFlow, a prominent cloud-based solution (Zinjad et al., 2024) with emphasis on privacy, customisation, response speed and cost-effectiveness. The hosted platform performed better across all assessed categories. It provided response rates, enhanced data privacy by avoiding external data sharing and continuous offline functionality. The removal of subscription fees also increases its appeal to budget-conscious entities, such as schools and nonprofit groups. These comparative results substantiate the strategic value of local AI deployments for next-generation intelligent document automation solutions.

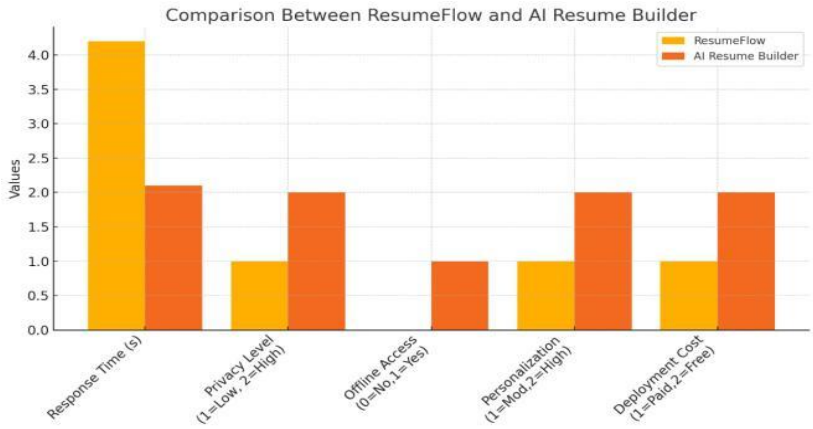

Fig. 4. Graphical comparison of key performance metrics between Resume-Flow and AI Resume Builder. The AI Resume Builder demonstrates superior efficiency in response time, privacy level, offline functionality, and deployment flexibility.

VI. Conclusion and Future Scope

We assessed the AI Resume Builder's effectiveness by measuring response speed, understanding of context, and formatting accuracy across various user groups, including technical experts, managerial applicants, and scholars. This method guaranteed reliability in managing job sectors. Importantly, combining engineering with the locally run DeepSeek-R1 model led to marked enhancements in both precision and processing efficiency. Unlike web-based resume creators, the system delivers results with high levels of customisation while improving data confidentiality by operating without reliance on external APIs. Input from end users, mainly graduating students and those actively seeking jobs, emphasised the platform's ease of use, attractive design, and full offline functionality. Stress testing also confirmed the system's reliability, ensuring performance under simultaneous access by multiple users. Utilising a modular backend structure and local device processing helped lower response delays and enhance scalability. Together, these results highlight the effectiveness of run AI in document automation functions and suggest significant promise for implementing on-device large language models in both educational and hiring settings.

The creation of the AI Resume Builder highlights the potential of running language models (LLMs) locally and embedding them in web applications. Utilising on-device LLMs, the platform offers context-aware automation while avoiding the risks tied to sending private user data to remote cloud servers. Every operation is handled offline, guaranteeing that personal details stay on the user's device and maintaining privacy safeguards. From a standpoint, the system utilises a modular structure that combines a React-driven frontend with a Spring Boot backend. This division of responsibilities enhances scalability and maintainability, and ensures smooth interaction between client and server components. Prompt engineering methods are applied to create resumes that conform to industry standards while allowing users to tailor the results to their profiles and preferences. Unlike cloud platforms such as ResumeFlow, the locally deployed architecture minimises latency and offers a much greater level of customisation. The guarantee that data remains within the environment is especially important amid increased privacy concerns about the use of artificial intelligence. This project demonstrates that running

sophisticated AI models locally can provide strong privacy safeguards alongside excellent performance.

The practical assessment of the system indicated enhancements in processing velocity and result precision. The AI Resume Builder demonstrated flexibility across scenarios, consistently producing comprehensive, contextually relevant resumes. Its modular design facilitates expandability, enabling the incorporation of more templates, professional fields or formatting choices without requiring extensive structural modifications. Importantly, resource-conserving local models like DeepSeek-R1 were found to match. At times, it outperforms cloud-based models, especially in settings with limited computational capacity. Automating resume creation simplifies the user's process, reliably generating documents while minimising manual work. The system demonstrated performance even under heavy usage, highlighting its suitability for implementation in environments such as universities, career services, and HR offices. Upcoming efforts will concentrate on advancing system functions. Broadening its use cases. Including recruiter feedback options might allow model improvements and better resume outputs. Support for languages would increase accessibility for a wider global audience. Adding automated scoring features could deliver assessment feedback and practical suggestions to users. Expanding the system's features to include cover letter and statement of purpose creation would further enhance its usefulness. Furthermore, integrating analytics to detect labour market patterns could turn the platform into a tool for job applicants. With these advancements, the AI Resume Builder has the potential to evolve from a privacy-centric resume generator into a comprehensive, AI-powered career support system.

REFERENCES

[1] Zinjad, S. B., Bhattacharjee, A., Bhilegaonkar, A., & Liu, H. (2024, July). Resumeflow: An llm-facilitated pipeline for personalised resume generation and refinement. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 2781-2785).

[2] Vatsal, S., & Dubey, H. (2024). A survey of prompt engineering methods in large language models for different NLP tasks. *arXiv preprint arXiv:2407.12994*.

[3] Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, *1*.

[4] Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, *1*.

[5] Pahune, S., & Rewatkar, N. (2024). Cognitive automation in the supply chain: Unleashing the power of RPA vs. Gen AI.

[6] Chen, B., Zhang, Z., Langrené, N., & Zhu, S. (2025). Unleashing the potential of prompt engineering for large language models. *Patterns*, *6*(6).

[7] So, C. C., Sun, Y., Wang, J. M., Yung, S. P., Loh, A. W. K., & Chau, C. P. (2025, July). Are large language models capable of deep relational reasoning? insights from deepseek-r1 and benchmark comparisons. In 2025 IEEE International Conference on Artificial Intelligence Testing (AITest) (pp. 168-177). IEEE.