A PROJECT REPORT ON

SMART RESUME BUILDER WITH AI SUGGESTIONS

Introduction

The modern job market is highly competitive, demanding resumes that are not only well-structured but also strategically written to capture the attention of recruiters and pass through automated screening systems. Many job seekers, however, lack the expertise to optimize their resume content effectively. The "Smart Resume Builder" project was initiated to address this gap by creating a user-friendly, web-based tool that empowers users to build professional resumes while receiving real-time, AI-driven feedback for improvement.

The primary objective was to develop a full-stack application that provides a seamless user experience from data entry to final output. The final deliverable is an interactive web application capable of generating polished resumes and exporting them to a universally accepted format like PDF.

Abstract

The application features an interactive, dual-panel interface where users can input their professional information into a structured form on one side and see a live, formatted preview of their resume on the other.

The core innovation of this project is the integration of a powerful generative AI (Google's Gemini Pro). By making a direct API call, the application can analyze user-provided text—such as a professional summary—and return actionable, expert-level suggestions for improvement.

The system is built on modern web technologies, utilizing React.js and Vite for a highly efficient frontend. Styling is handled with Tailwind CSS, and functionality is extended with Lucide React for icons and Axios for API communication.

Tools and Technologies Used

Frontend (Client-Side):

- **React.js:** A popular JavaScript library for building dynamic and responsive user interfaces.
- **Tailwind CSS:** A utility-first CSS framework for rapidly building custom, modern designs. It was integrated using the official @tailwindcss/vite plugin for a streamlined setup.
- **Axios:** A promise-based HTTP client for making API requests from the browser to the backend service.

Backend & Third-Party Services:

- **Node.js:** A JavaScript runtime environment used as the foundation for the server-side logic.

- **Google Gemini API:** The core AI service used to generate content suggestions. Direct REST API calls were made to the gemini-2.5-flash model.
- **MongoDB:** A NoSQL database used for storing resume data in a flexible, JSON-like format.

Export Functionality:

- **Browser's Native Print-to-PDF:** The PDF export feature was implemented by leveraging the browser's built-in "Print" functionality. Custom print-specific CSS styles were used to ensure only the resume preview is visible and properly formatted during the print/save process, offering a dependency-free solution.

Steps Involved in Building the Project

- **Environment Setup:** Initiated the project with separate client (React/Vite) and server (Node.js) directories. Installed essential packages including axios for API calls, lucide-react for icons, tailwindcss with the @tailwindcss/vite plugin for styling, mongoose for database interaction, and dotenv for environment variables.
- **Backend Foundation:** Set up a Node.js server to manage data persistence using MongoDB. Configured endpoints to handle resume data and integrated the AI suggestion logic by making direct HTTP requests to the Google Gemini API.
- **Frontend Structure:** Built a React application featuring a two-column layout (App.jsx) displaying the editor form and a live resume preview. Implemented central state management to dynamically update the preview based on user input in the form (ResumeForm.jsx).
- **Feature Integration:** Added the AI suggestion capability within the form, sending user text to the backend for analysis. Implemented the PDF export feature by utilizing the browser's native print functionality combined with print-specific CSS to isolate and format the resume preview.
- **Styling & Refinement:** Applied Tailwind CSS utility classes and incorporated Lucide React icons throughout the interface to achieve a clean, modern, and user-friendly design.

Conclusion

The "**Smart Resume Builder**" project successfully delivered a functional, AI-enhanced tool for creating resumes. Utilizing a modern React/Vite frontend with Tailwind CSS and a Node.js backend for AI integration, the application offers an intuitive user experience and valuable AI suggestions. Key successes included a robust backend architecture making direct calls to the Gemini API and leveraging the browser's native print function for dependency-free PDF export. The primary challenges centered on navigating Google Cloud API authentication, which was resolved by correctly enabling services and verifying API key permissions. Overall, the project serves as a successful proof-of-concept, demonstrating a streamlined approach to building intelligent web applications. Future work could extend functionality with user accounts and template options.