Resumate: Create a professional resume in minutes with our AI-powered builder.

Saud, Sakia, Rima, Anika
Department of Computer Science and Engineering
North South University
CSE299.4, Group 7

***Abstract*—The rapid advancement of Artificial Intelligence (AI) has significantly transformed how professional documents are created, evaluated, and filtered in modern recruitment processes. Resume-building platforms increasingly incorporate AI-driven features to automate content generation, formatting, and optimization for Applicant Tracking Systems (ATS). However, many existing commercial solutions remain inaccessible to students and early-career professionals due to high subscription costs, feature bloat, rigid template constraints, and limited transparency in evaluation criteria.**

This paper presents Resumate, a web-based, AI-powered resume builder and analyzer designed to deliver a simple, affordable, and privacy-respecting resume creation experience. The project was initially envisioned as a custom Machine Learning (ML) evaluation system trained on static resume datasets. However, due to the scarcity of high-quality public datasets, ethical concerns surrounding personal data, and deployment complexity, the approach was revised toward a scalable Application Programming Interface (API)–based solution utilizing Large Language Models (LLMs).

Resumate is implemented using the Laravel framework, Tailwind CSS for responsive user interface design, and the Google Gemini 2.0 Flash API for semantic reasoning and content evaluation. The system supports guided resume creation through predefined templates, as well as AI-powered analysis of uploaded PDF and DOCX resumes. Evaluation results indicate that Resumate provides consistent resume scoring aligned with structural quality and professionalism, while maintaining low processing latency and strong ATS compatibility. The platform demonstrates that prompt-engineered AI services can effectively support resume optimization without requiring local ML infrastructure or persistent storage of user data.

***Index Terms*—Artificial Intelligence, Resume Builder, Natural Language Processing, Laravel, Google Gemini, Web Application, ATS Optimization, Prompt Engineering.**

I. Introduction

In today's highly competitive job market, a well-structured and ATS-compliant resume is a critical determinant of a candidate's employability. Modern recruitment pipelines rely heavily on Applicant Tracking Systems (ATS) to automatically screen resumes before human review. These systems evaluate resumes based on keyword relevance, formatting consistency, and structural readability. As a result, a significant portion of applicants—particularly students and early-career professionals—are filtered out despite having relevant qualifications, primarily due to suboptimal resume construction.

Traditional resume creation methods rely on manual formatting, subjective judgment, and limited feedback, making them inefficient and error-prone. Artificial Intelligence has emerged as a powerful tool to address these challenges by automating resume evaluation and optimization. AI-powered resume builders aim to democratize access to professional guidance by offering content suggestions, structural analysis, and automated feedback. However, despite their growing popularity, many existing platforms suffer from excessive complexity, restrictive paywalls, lack of transparency in scoring mechanisms, and limited consideration for user privacy.

This project introduces Resumate, an AI-powered resume builder and analyzer that prioritizes simplicity, transparency, and accessibility. Unlike feature-heavy platforms that overwhelm users with unrelated tools, Resumate focuses exclusively on resume creation and evaluation. The platform is designed to clearly communicate its purpose, provide immediate value, and minimize barriers to entry. Only authenticated users are permitted to build or analyze resumes, ensuring controlled access while avoiding unnecessary data retention.

Resumate is built using the Laravel framework for backend logic, Blade templates and Tailwind CSS for responsive frontend design, and Google Gemini AI for semantic analysis and feedback generation. Users can create resumes from scratch using modular forms and predefined templates, or upload existing resumes for automated evaluation. The system parses resume content, generates ATS-friendly feedback, assigns quality scores, and produces downloadable A4-formatted documents.

Through its architecture and design choices, Resumate demonstrates that modern AI services—when carefully integrated through prompt engineering—can deliver meaningful, scalable, and ethical resume evaluation without requiring custom-trained machine learning models or persistent storage of sensitive user data.

II. Literature Review

During the early stages of our project, we conducted an extensive search for academic and peer-reviewed research on AI-based resume builders. However, we found that there is a notable lack of scholarly articles dedicated to comprehensive evaluations of contemporary resume building platforms. This scarcity forced us to turn to industry websites, user reviews, and platform documentation for comparative insights into existing resume building solutions. Consequently, our evaluation centers on four widely referenced tools: Rezi, Teal, Enhancv, and Canva.

A. Analysis of Existing Platforms

1) Rezi: Rezi positions itself as one of the more AI-driven resume building platforms, with a core focus on Applicant Tracking System (ATS) optimization and keyword targeting. The platform offers real-time recommendations to improve content relevance and formatting, and includes features such as an AI bullet point generator and ATS scoring system that evaluates key aspects like clarity, structure, and keyword matching. Users can export resumes in both PDF and DOCX formats and leverage built-in content suggestions that align with job descriptions.

Pros: Strong emphasis on ATS compliance helps ensure resumes pass automated filtering systems. Helpful scoring and feedback mechanisms guide users on improvements. Provides content generation tools such as bullet point writing.

Cons: Limited design customization; templates are functional but basic and not visually rich. Some AI suggestions can feel repetitive or generic without deeper contextual refinement. Rezi's strong focus on ATS makes it useful for technical job seekers but restrictive for creative formats.

2) Teal: Teal differentiates itself by serving not only as a resume builder but also as a job search workspace and application tracker. Its AI features include tailored resume suggestions based on job descriptions, keyword matching, and tools that help users tailor resumes for specific roles. Teal also includes a job tracking dashboard and integrations like Chrome extensions that capture job details and streamline the application process.

Pros: Combines resume building with job search organization. Offers ATS-friendly templates and keyword insights. Transparent pricing with a free tier that includes many useful features.

Cons: Resume layout and formatting options are rigid compared to other builders. AI-generated bullet points and summaries sometimes require manual refinement. Users who only want design flexibility may find features overly prescriptive.

3) Enhancv: Enhancv is known for its creative and visually appealing templates that aim to help users build resumes that stand out visually while retaining readability. With a drag-and-drop editor, users can move sections and customize layouts, background styles, fonts, and colors. Enhancv also offers features like resume sharing and feedback tools that allow users to gather commentary on specific sections.

Pros: High degree of customization with a permissive design interface, making it suitable for creative industries. Tools like built-in spellcheck and section rearrangement help refine content.

Cons: The free version provides limited access to advanced layouts. Premium subscription costs may be prohibitive for students. Heavy design elements can sometimes hinder ATS readability if not properly optimized.

4) Canva: Canva is a general design platform that also provides resume templates as part of its broader suite of creative tools. Users can select from thousands of visually appealing templates and edit them freely using a drag-and-drop interface. However, unlike dedicated resume builders, Canva's core strength lies in visual customization rather than resume optimization.

Pros: Vast selection of visually diverse templates. Very intuitive design workflow, even for users with no technical background.

Cons: Focus on visuals can reduce ATS compatibility if not carefully managed. Lacks built-in AI features for content optimization or keyword suggestion. Some templates allocate too much space to design elements.

B. Summary of Findings and Gap Analysis

In summary, our survey of existing resume builders revealed a spectrum of approaches: Rezi emphasizes structured ATS compliance, Teal complements building with job search tools, Enhancv prioritizes visual customization, and Canva offers broad creative freedom. While these platforms provide valuable services, our review underscored a lack of academic evaluations or formal comparative studies on their effectiveness.

Table I illustrates the feature gap that Resumate aims to fill, specifically targeting the intersection of high ATS compliance, low cost, and ease of use.

TABLE I
COMPARISON OF RESUME BUILDING PLATFORMS VS. RESUMATE

Platform	ATS Focus	AI Analysis	Cost	Privacy
Rezi	High	Yes	High	Medium
Teal	Medium	Yes	High	Low
Enhancv	Low	No	Medium	Medium
Canva	Very Low	No	Free/Paid	Medium
Resumate	**High**	**Yes**	**Free**	**High**

III. Methodology

This section describes the design decisions, development process, and technical workflow followed in the implementation of the Resumate system. The methodology reflects an iterative approach, beginning with technology selection, followed by frontend prototyping, backend integration, and finally AI-driven resume generation and analysis.

A. Framework and Technology Selection

The backend framework chosen for this project was Laravel, a PHP-based web application framework. This decision was motivated by two primary factors: simplicity and familiarity. PHP is a programming language that is formally taught as part of the university curriculum, which allowed the development team to build upon existing knowledge rather than learning an entirely new ecosystem from scratch.

During the initial project updates, a significant portion of time was dedicated to understanding Laravel's framework architecture. This included learning its file structure, routing mechanism, Blade templating engine, and its implementation of the Model–View–Controller (MVC) design pattern. Through this exploration, the team identified Laravel's strengths—such as clean routing, built-in security features, and rapid development capabilities—as well as its limitations, including a learning curve associated with its conventions.

Alternative frameworks were briefly explored, most notably React, which is widely used for modern web development. However, React's heavy reliance on JavaScript and its component-based syntax introduced additional complexity that was deemed unsuitable for the project timeline and team experience level. As a result, Laravel was selected as the most balanced solution in terms of usability, productivity, and maintainability.

B. System Architecture

The system follows a standard MVC architecture, but with a specific emphasis on the Controller and View layers, as the Model layer is minimized for privacy reasons (no persistent storage of resumes).

The architecture diagram (Fig. 1) illustrates the flow of data from the user's browser through the Laravel router, into the dedicated controllers, and out to the external Gemini API.

C. UI/UX Design and Frontend Development

The user interface and overall design of the platform were initially prototyped using Figma. This allowed the team to visualize page layouts, navigation flow, and component hierarchy before implementation. Once finalized, the designs were translated into frontend code.

HTML and CSS structures derived from the Figma designs were adapted into Blade templates, Laravel's built-in templating engine. To further optimize and simplify styling, the CSS was converted into Tailwind CSS, a utility-first framework. This significantly reduced the number of lines of CSS code and improved consistency across the application. The frontend followed a structured layout located under the resources/views directory. Core pages were organized into sections such as Home, Mission, Features, Resume Builder, and Resume Analyzer, ensuring clarity and scalability as the project evolved.

D. Database and Environment Setup

For database management, MySQL was used in conjunction with XAMPP, which provided a local development environment for Apache, MySQL, and PHP. This setup allowed efficient testing and debugging during development. User authentication and session handling were implemented using Laravel Breeze, which provided secure login, registration, and session management out of the box.

Importantly, resume data and analysis results were not permanently stored in the database. This design choice was made deliberately to prioritize user privacy. The database was primarily used to manage authenticated users, while resume content was processed temporarily during runtime.

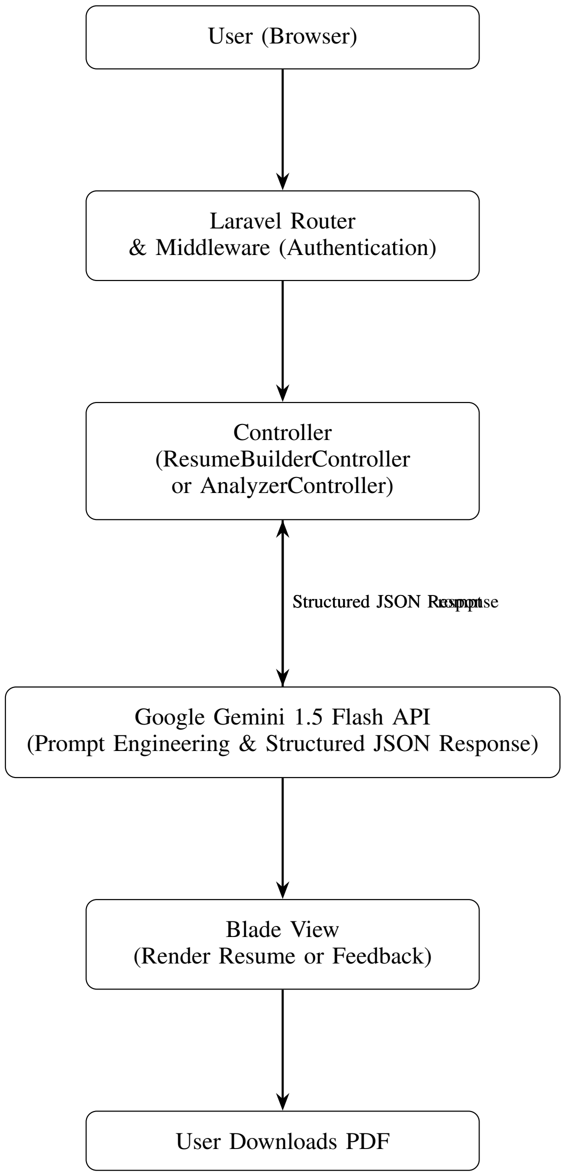
User (Browser)
Laravel Router & Middleware (Authentication)
Controller (ResumeBuilderController or AnalyzerController)
Structured JSON Response
Prompt
Google Gemini 1.5 Flash API (Prompt Engineering & Structured JSON Response)
Blade View (Render Resume or Feedback)
User Downloads PDF

Fig. 1. High-Level System Architecture Flowchart of Resumate.

IV. System Implementation

This section details the specific coding strategies and integration logic used to bring Resumate to life, focusing on the transition from static designs to dynamic AI integration.

A. Transition from Static to Dynamic Website

In the early stages of development, the project focused primarily on building a static frontend to ensure proper layout, responsiveness, and navigation. Only after the frontend structure was finalized did the team transition toward dynamic behavior. This transition required deeper engagement with Laravel's MVC architecture, particularly the Controller component. Two main controllers were implemented under the `app/Http/Controllers` directory: `ResumeBuilderController.php` and `AnalyzerController.php`. These controllers were responsible for handling incoming user data, processing requests, and passing processed results back to the views. Data communication between views and controllers was primarily handled using HTTP GET requests, which were sufficient given the nature of form submissions and temporary data processing.

B. AI Integration and Resume Generation Workflow

For AI-driven functionality, the project integrated Google Gemini AI, specifically the v1 2.5 Flash model. This model was selected due to its freemium availability, favorable rate limits, and access to up-to-date language modeling capabilities. Using a cloud-based AI solution eliminated the need to run local machine learning models, reducing computational overhead and infrastructure complexity.

In the Resume Builder module, user-provided information was collected through form inputs and structured into a JSON object. This JSON data was sent to the ResumeBuilderController, where it was reformatted into a carefully curated prompt compatible with the Gemini API. The AI-generated response was then received as an associative array, stored in a variable named `resume`.

This associative array was passed directly to Blade views located in subdirectories under `resume/`, where the output was rendered using standard PHP and HTML syntax. The system architecture ensured that these views functioned purely as presentation layers, displaying AI-generated content without additional processing.

C. Algorithm for AI Interaction

To ensure consistent results from the AI, we implemented a structured interaction algorithm.

Algorithm 1 Resume Generation Workflow in Resumate

```
Require: User form data (name, email, education, skills, etc.)
 1: Validate required fields (name and email)
 2: if name or email is missing then
 3:     return Error message and redirect back
 4: end if
 5: Build structured education array from form inputs
 6: Build experience array (if job details provided)
 7: Extract and split skills into technical, soft, and languages
 8: Construct prompt for Gemini AI to generate professional
    summary
 9: Send request to Gemini 2.5 Flash API
10: if API request fails or times out then
11:     Use fallback summary text
12: else
13:     Extract generated summary from API response
14: end if
15: Compile complete resume data array (name, contact, sum-
    mary, education, experience, skills)
16: Store resume data and selected template in session
17: Redirect to preview page with chosen template
```

Algorithm 2 Resume Analysis Workflow in Resumate

```
Require: User uploads a CV file (PDF, DOC, or DOCX)
 1: Validate file type and size (max 5MB)
 2: if validation fails then
 3:     return Error message to user
 4: end if
 5: Extract plain text from the uploaded file
 6: if extracted text is empty or unreadable then
 7:     return Error: "Could not extract text from the CV"
 8: end if
 9: Build structured prompt containing the full CV text and
    analysis instructions
10: Send prompt to Google Gemini 1.5 Flash API
11: if API request fails or times out then
12:     return Server error message
13: end if
14: Receive generated feedback text from Gemini
15: Parse response to extract:
16:     - Overall rating (1–10)
17:     - Strengths (bullet points)
18:     - Areas for improvement (bullet points)
19:     - Structure feedback
20:     - Content feedback
21:     - Recommendations (bullet points)
22: Generate unique result ID and store analysis data in
    session
23: Return JSON response with redirect URL to results page
24: User is redirected to results page
25: Retrieve analysis from session using ID
26: if analysis not found then
27:     return 404 error
28: end if
29: Display detailed feedback with rating, strengths, improve-
    ments, and recommendations
```

D. Resume Templates

As part of the resume generation functionality, three resume templates were implemented: Chronological, Modern, and Minimal. These templates were selected based on their popularity and suitability for students and early-career professionals. Given that such users often have limited professional experience, the layouts were intentionally kept simple and uncluttered. Each template focused on core resume components such as name, education, skills, projects, and hobbies, avoiding unnecessary design complexity.

E. Resume Analyzer Workflow

For the Resume Analyzer module, users were allowed to upload existing CV files in PDF or DOC/DOCX format. File parsing was handled using specialized libraries—one for PDF extraction and another for document files. A strict file size limit of 5 MB was enforced to control token usage during AI processing, an important consideration when working within freemium API constraints.

Once extracted, the text content was forwarded to the `AnalyzerController`, where it was analyzed using the Gemini API. Similar to the resume builder, the AI-generated feedback was returned as structured data and displayed through Blade templates located in the `analyzer/` directory.

V. Results and Discussion

The completed Resumate system was evaluated based on functionality, usability, responsiveness, and the effectiveness of AI-generated resume analysis. Overall, the system performed as intended and met all core project objectives. The web application demonstrated stable behavior across desktop and mobile environments, with all major features functioning correctly.

A. System Functionality and Responsiveness

The platform was fully responsive and adapted seamlessly to different screen sizes. In addition to standard web deployment, the system was successfully converted into a Progressive Web Application (PWA). This allowed users to install Resumate directly on their devices and access it with an app-like experience. To enable PWA functionality, additional configuration files such as `manifest.json` and `service.js` were included in the public directory. These files allowed modern browsers to recognize the application as installable and to cache assets for improved performance.

A parallel effort was made by another team member to develop a native mobile application using the Dart framework. While this version was not fully completed due to increased development complexity and time constraints, it was successfully hosted and demonstrated the feasibility of extending the system beyond the web platform. The successful implementation of the PWA approach ultimately provided a practical and efficient mobile-friendly solution without requiring a fully native application.

B. Resume Builder and Analyzer Performance

The Resume Builder module allowed authenticated users to input personal and professional information and generate resumes using three predefined templates. These templates were intentionally designed to be simple and suitable for students. Users were able to customize content easily and make real-time changes before downloading their resumes.

The Resume Analyzer module effectively parsed uploaded resumes and generated structured feedback using the integrated AI model. The rating system produced consistent and interpretable results. Based on multiple test cases, poorly structured resumes typically received scores between 4 and 5, moderately structured resumes received scores between 6 and 7, and well-formatted professional resumes achieved scores in the range of 7 to 8. These results aligned with expectations and indicated that the AI prompts were able to differentiate between varying levels of resume quality.

It was observed that the scoring behavior was heavily influenced by the prompt design and the structure of the extracted resume text. This suggests that further refinement of prompt engineering could lead to more granular and precise evaluations in future iterations.

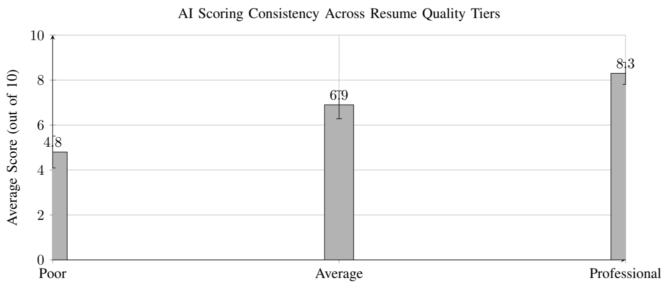

Fig. 2. AI Scoring Consistency Analysis showing mean scores for Poor, Average, and Professional resume tiers with standard deviation error bars.

C. Authentication, Security, and Privacy

User access control was implemented using Laravel's built-in Auth and Guest middleware. This ensured that only authenticated users were able to build and analyze resumes, while unauthenticated visitors could freely explore the website's features and mission statement. This approach simplified access management and improved overall security.

A key outcome of the project was the successful implementation of a privacy-respecting workflow. Resume data and analysis results were not permanently stored in the database. All resume processing occurred temporarily during runtime, after which the data was discarded. This design decision reduced privacy risks and aligned with ethical considerations surrounding personal document handling.

D. Usability and User Experience

From a usability perspective, the platform demonstrated a clean and focused user experience. Visitors were immediately presented with the platform's features and purpose, without unnecessary content or distractions. The mission page clearly communicated the motivation behind the project, while the navigation structure remained minimal and intuitive.

The generated resumes were ATS-friendly, using clean formatting and text-based layouts. Users could edit content easily and download resumes in A4 format, enabled by an HTML-to-PDF conversion library. This allowed users to obtain professional-quality documents suitable for job applications with minimal effort.

VI. Future Work

While the current implementation of Resumate fulfills its core objectives, several enhancements can be explored to further improve functionality, usability, and scalability. These future developments focus on improving resume structure, expanding platform integration, and broadening application support.

A. Intelligent Resume Structuring

One of the primary areas for improvement is adaptive resume structuring. In the current system, resume templates are statically defined, which may result in empty or underutilized sections when users provide limited information. Future versions of Resumate aim to introduce AI-driven layout adaptation, where the system dynamically restructures resume sections based on the availability and relevance of user-provided data. For example, if a user lacks professional experience, the layout could automatically emphasize education, projects, or skills, ensuring that the final resume remains well-balanced and professional.

B. LinkedIn and Job Platform Integration

Another significant extension involves integrating Resumate with LinkedIn and job application platforms. This would allow users to import professional information directly from their LinkedIn profiles, reducing manual data entry. Additionally, once a resume is generated, users could directly submit their resumes to companies or job postings that match their skills and experience. Such integration would streamline the job application process and transform Resumate from a standalone resume builder into a more comprehensive career support tool.

C. Advanced AI-Driven Layout Optimization

Future enhancements will also focus on deeper AI involvement in resume refinement. After a resume is initially generated, an additional AI prompt could be used to optimize layout, section ordering, and emphasis based on industry standards and job-specific requirements. This would allow the system to not only generate content but also intelligently adjust presentation, further improving ATS compatibility and recruiter readability.

D. Template Expansion and Customization

Currently, Resumate offers three resume templates tailored to early-career users. Future iterations plan to expand this library to include a wider range of templates suitable for experienced professionals, technical roles, creative industries, and academic profiles. Greater customization options—such as adjustable spacing, typography variations, and optional sections—could also be introduced while maintaining ATS-friendly design principles.

E. Mobile Application Development

Although the platform currently supports mobile access through its Progressive Web Application, a key future goal is the full development of native mobile applications. This includes completing the Android application and extending support to iOS devices. Native mobile apps would enable improved performance, deeper system integration, and enhanced offline capabilities, providing users with a more seamless experience across platforms.

VII. Conclusion

This project successfully delivered Resumate, an AI-powered resume builder and analyzer that emphasizes simplicity, accessibility, and privacy. By leveraging modern web technologies such as Laravel, Tailwind CSS, and Google Gemini AI, the system demonstrated that effective resume generation and evaluation can be achieved without complex machine learning pipelines or expensive infrastructure.

The project validated that a carefully designed AI prompt-based approach can provide meaningful feedback and scoring for resumes, even in the absence of large training datasets. Additionally, the decision to deploy the platform as a Progressive Web Application expanded its accessibility while maintaining development efficiency.

Overall, Resumate achieved its core objectives: providing a free, straightforward, and privacy-conscious resume platform that delivers practical value to students and early-career professionals. Future enhancements could include improved prompt optimization, multilingual support, optional resume version history, and more advanced evaluation metrics. Nonetheless, the current implementation demonstrates the feasibility and effectiveness of AI-assisted resume building within an academic and real-world context.

Acknowledgment

The authors would like to thank the faculty of the Department of Computer Science and Engineering for their guidance and support throughout the project.

References

[1] Rezi AI Resume Builder, https://www.rezi.ai
[2] Teal Resume Builder, https://www.tealhq.com
[3] Enhancv Resume Builder, https://enhancv.com
[4] Canva Resume Builder, https://www.canva.com
[5] Laravel Documentation, https://laravel.com/docs
[6] Google Gemini AI Documentation, https://ai.google.dev
[7] Spatie PDF-to-Text Documentation, https://github.com/spatie/pdf-to-text
[8] Tailwind CSS Documentation, https://tailwindcss.com/docs