IJARSCT

International Journal of Advanced Research in Science, Communication and Technology

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

ISSN: 2581-9429

Volume 6, Issue 1, March 2026

Impact Factor: 8.2

ResumeGen – Online Resume Generator Using Flask

Prof. Rahul Lilhare[1], Anjali B. Gajbhiye[2], Vaishnavi Salve[3]

Assistant Professor, MCA, KDK College of Engineering Nagpur, India [1]

PG Scholar, MCA, KDK College of Engineering Nagpur, India[2,3]

rahul.lilhare@kdkce.edu.in, anjalibgajbhiye.mca24f@kdkce.edu.in,

salvevvijayrao.mca24f@kdkce.edu.in

Abstract: *In today's competitive job market, a well-structured and professional resume plays a crucial role in securing employment opportunities. However, many candidates face challenges in designing resumes due to lack of technical knowledge and formatting skills. This project proposes ResumeGen – an Online Resume Generator using Flask, which enables users to create professional resumes easily through an interactive web interface. The system allows users to input personal, educational, and professional details, select customizable templates, preview the resume in real-time, and download it in PDF format. The application is developed using Python Flask framework with HTML, CSS, and JavaScript for front-end development. The system aims to simplify resume creation by providing automation, user-friendly templates, and multilingual support. This tool significantly reduces time and effort required in resume building while ensuring accuracy and professional presentation.*

Keywords*:* Resume Generator, Flask Framework, Web Application, PDF Conversion, Templates

I. INTRODUCTION

A resume is a formal document summarizing an individual's educational background, professional experience, technical skills, and achievements. It serves as the first impression for recruiters and plays a crucial role in job selection processes. Traditional resume creation involves manual formatting using word processors, which can timeconsuming and prone to formatting inconsistencies.

Many job seekers lack knowledge of professional resume design standards such as alignment, spacing, font selection, and section structuring. As a result, resumes may appear unorganized and fail to meet recruiter expectations. ResumeGen is designed to simplify this process by offering a structured and automated web-based platform. The application enables users to generate standardized resumes by entering their details into predefined fields and selecting suitable templates. By automating the formatting process, ResumeGen enhances efficiency and ensures professional presentation standards.

II. LITERATURE REVIEW

Several online resume builders exist in the market that allow users to generate resumes digitally. However, many of these tools operate on subscription-based models and restrict access to premium templates unless payment is made. Research on web-based automation systems indicates that interactive user interfaces and real-time preview mechanisms significantly enhance user experience and engagement. Existing systems often lack multilingual support, offline functionality, and template customization options.

Some resume-building platforms also add watermarks to downloaded resumes, making them unsuitable for professional use. Therefore, there is a need for an efficient and cost-effective system that provides customization, flexibility, and watermarkfree resume downloads.

ResumeGen addresses these limitations by providing a customizable and automated solution using a lightweight backend architecture developed with Flask.

III. PROPOSED SYSTEM

The development of ResumeGen follows a structured Software Development Life Cycle (SDLC) approach to ensure systematic implementation and reliability. The proposed methodology consists of the following phases:

1. Requirement Analysis

In this phase, user requirements were collected by analyzing common resume components used in professional documents. The identified modules include:

Personal Information
Career Objective
Educational Qualifications
Technical Skills
Work Experience
Projects
Certifications
Achievements
The system was designed to accommodate both students and working professionals.

2. System Design

The system follows a modular three-tier architecture:
Presentation Layer (Frontend) – Responsible for user interaction using HTML, CSS, and JavaScript.
Application Layer (Backend) – Implemented using Flask framework to process business logic.
Data Layer (Database) – SQLite database used for temporary data storage.
The design ensures separation of concerns, making the system scalable and maintainable.

3. Data Collection and Form Handling

Structured web forms were created to collect user inputs. Each input field corresponds to a resume section. Flask routes handle form submission and pass the data to backend functions for processing. Input fields include validation constraints such as: Required fields
Email format validation
Date formatting
Character limits
This ensures data accuracy and consistency.

4. Input Validation and Error Handling

Validation mechanisms were implemented to prevent incomplete or incorrect submissions. Server-side validation ensures data integrity even if client-side validation fails.
Error handling mechanisms provide user-friendly messages in case of:
Missing mandatory fields
Invalid data formats
System processing errors

5. Template Rendering Using Jinja Engine

The Jinja2 templating engine dynamically injects user data into predefined resume templates. Each template follows structured formatting standards including:
Consistent font usage
Proper alignment
Section headings

Bullet-point formatting
Dynamic rendering ensures that user input automatically updates the resume preview.

6. Real-Time Preview Mechanism
The system provides real-time resume preview functionality before final download. This improves usability and allows users to verify formatting and content accuracy.

7. PDF Generation Module
Once the resume is finalized, the system converts the formatted HTML content into a downloadable PDF file using backend rendering libraries. The PDF generation module ensures:
Proper page alignment
Margin consistency
Cross-device compatibility

8. Testing and Evaluation
The system underwent functional testing and usability testing. Testing scenarios included:
Data entry validation
Template switching
PDF download verification
Cross-browser compatibility
User feedback was collected to improve interface responsiveness and template structure.

9. Performance Optimization
To enhance performance, lightweight libraries were used and unnecessary dependencies were avoided. Flask's routing efficiency ensures quick request-response cycles.
Summary of Methodology Flow:
User Input → Validation → Template Selection → Dynamic Rendering → Preview → PDF Generation → Download

IV. SYSTEM ARCHITECTURE

The ResumeGen system is designed using a three-tier architecture model to ensure modularity, scalability, and efficient data handling. The architecture separates the user interface, application logic, and data management components, thereby improving maintainability and performance of the system.
The architecture consists of the following layers:

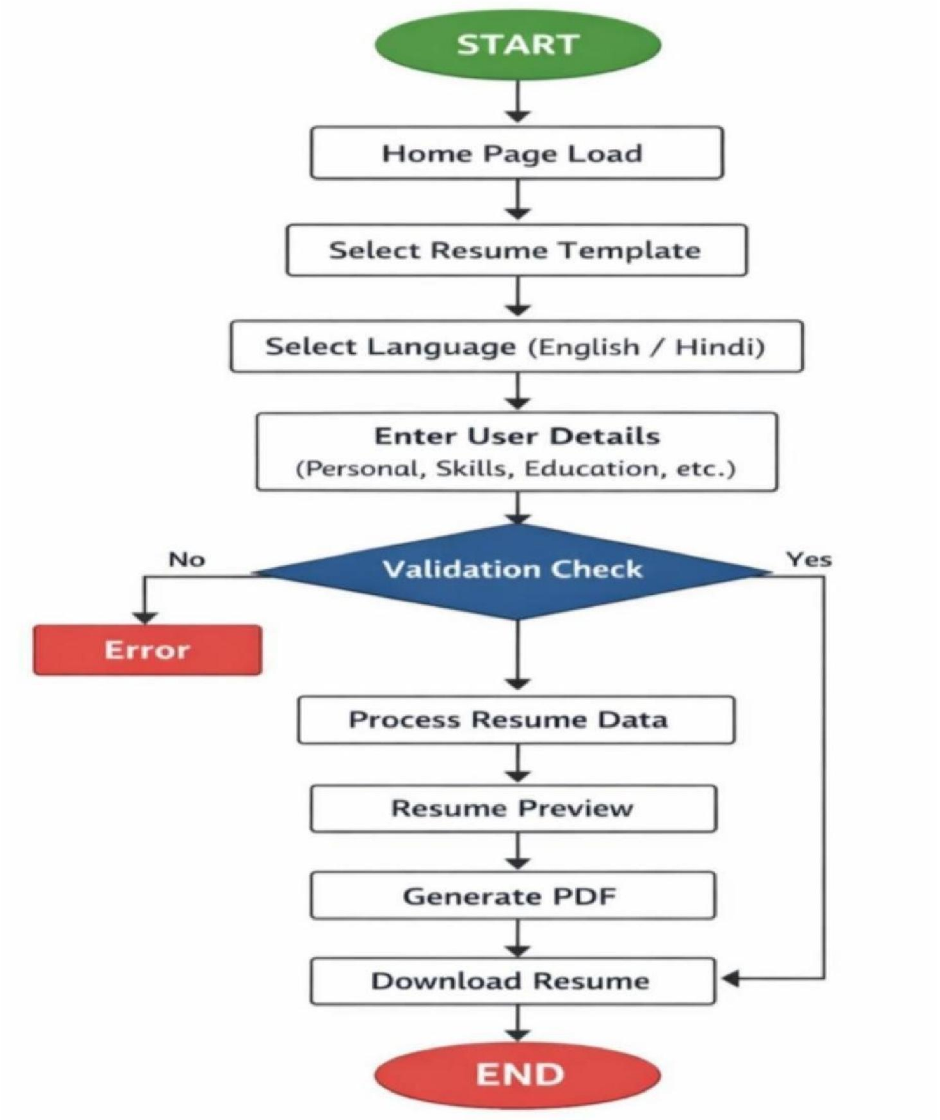
1. Presentation Layer (Frontend Layer)
The presentation layer is responsible for handling user interaction with the system. This layer provides a graphical user interface through which users can input their personal and professional details required for resume generation. The frontend is developed using:
HTML5 for structuring web pages
CSS3 for styling and layout design
JavaScript for client-side interactivity
Users interact with structured input forms to enter details such as:
Personal Information
Educational Background
Technical Skills
Work Experience
Certifications

Projects and Achievements
The presentation layer ensures data readability and provides a real-time preview of resume content before final submission.

2. Application Layer (Backend Layer)
The application layer acts as the core processing unit of the system. This layer is implemented using the Flask framework, which is a lightweight Python-based web development framework.
The backend performs the following operations:
Handling user requests via HTTP methods
Managing form submissions
Validating input data
Processing resume content
Applying selected templates
Communicating with the database
Generating formatted output
Flask routes are defined to manage navigation between different pages such as:
Home Page
Data Entry Page
Template Selection Page
Resume Preview Page
Download Page
The Jinja2 templating engine is used to dynamically integrate user-provided data into predefined HTML resume templates.

3. Data Layer (Database Layer)
The data layer is responsible for storing user-provided information temporarily during the resume creation process. The system uses SQLite database, which is lightweight and suitable for small-scale web applications. The database stores structured information such as:
User personal details
Educational qualifications
Technical skills
Project descriptions Work experience
This layer ensures efficient data retrieval and processing during template rendering and resume generation.

4. Output Generation Module
After processing user input and applying the selected template, the output generation module converts the formatted HTML resume into a downloadable PDF document.
The module ensures:
Proper alignment of content
Consistent margin settings
Font uniformity
Page layout compatibility
The generated resume can be downloaded and used for professional purposes.
Overall Architectural Workflow
User Interface → Flask Backend Processing → SQLite Database → Template Rendering → PDF Generation → Resume Download



V. IMPLEMENTATION AND TESTING

The implementation of the ResumeGen system involves the integration of frontend and backend technologies to develop a responsive and automated resume generation platform. The system is implemented using a modular approach, ensuring flexibility, scalability, and ease of maintenance.

1. Frontend Implementation

The frontend interface of the system is developed using HTML, CSS, and JavaScript to provide a user-friendly environment for data entry. Structured web forms are designed to collect resume-related information from users.

Key features of the frontend implementation include:

Section-wise data entry for resume components

Responsive user interface design

Template selection functionality
Real-time resume preview
Form validation using JavaScript
CSS styling ensures professional formatting of resume sections such as headings, bullet points, alignment, and spacing.

2. Backend Implementation Using Flask

The backend of the system is implemented using the Flask web framework. Flask handles user requests and processes resume data through server-side logic.
Backend functionalities include:
URL routing for navigation between modules
Handling form submissions
Input validation and error checking
Communication with the database
Dynamic template rendering
Resume file generation
Flask routes are defined to manage system pages such as:
Home Page
Resume Data Entry Page
Template Selection Page
Resume Preview Page
Download Page
The Jinja2 templating engine is integrated to dynamically generate resume templates by inserting user-provided data into predefined HTML layouts.

3. Database Integration

SQLite database is used for storing user data temporarily during the resume generation process. The database maintains structured records for:
Personal Information
Educational Qualifications
Technical Skills
Work Experience Certifications
Efficient data retrieval and storage enable seamless template rendering and resume generation.

4. PDF Conversion Module

After rendering the resume template dynamically, the system converts the HTML content into a downloadable PDF file using backend rendering tools.
The PDF generation module ensures: Proper page alignment
Margin consistency
Font uniformity
Layout compatibility
The generated document is suitable for professional job applications.

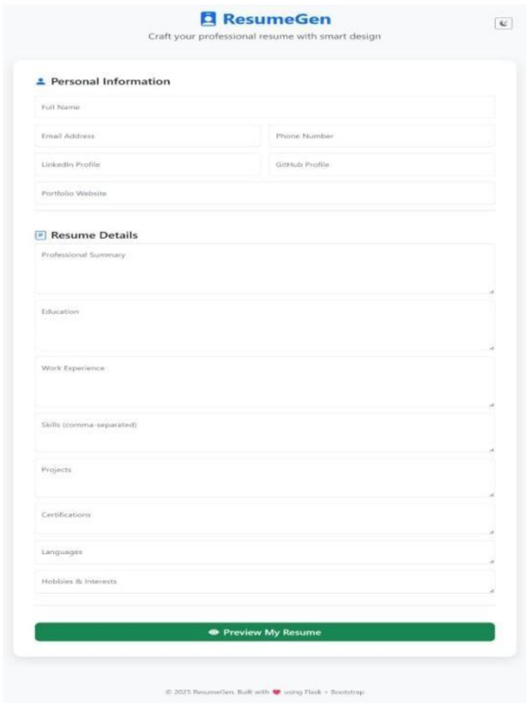
5. System Testing

To ensure system reliability and functionality, multiple testing techniques were employed: a) Functional Testing
Functional testing was performed to verify that each module of the system operates according to the specified requirements. Testing scenarios included:

Form data submission
Input validation
Template selection
Resume preview generation   PDF download functionality   b) Usability Testing
Usability testing was conducted to evaluate the user-friendliness of the system interface. Users were asked to generate resumes using different templates and provide feedback regarding:
Ease of navigation
Clarity of input forms
Template readability
Download process
Compatibility Testing
Compatibility testing ensured that the application functions correctly across different web browsers such as: Google Chrome
Mozilla Firefox Microsoft Edge
The generated PDF resumes were also tested for compatibility across various devices.   a. Performance Testing
Performance testing evaluated the response time of the system during data processing and resume generation. Results indicated efficient performance with minimal delay in PDF generation.
Testing Results
The testing process confirmed that the ResumeGen system:
Successfully captures user input
Generates structured resume templates
Provides real-time preview functionality
Produces downloadable PDF resumes
Maintains formatting consistency
Thus, the system meets its intended functional and usability requirements.

VI. RESULT AND ANALYSIS

The ResumeGen system was tested with multiple user scenarios including students, fresh graduates, and working professionals. The system successfully generated well-formatted resumes in different templates.
Real-time preview functionality improved usability by allowing users to verify their details before downloading the resume. The generated PDF output maintained formatting consistency across different devices.
User feedback indicated improved ease of use and time efficiency compared to traditional manual resume creation methods.



VII. CONCLUSION

ResumeGen provides an efficient and automated solution for professional resume creation. The system reduces manual effort, ensures formatting consistency, and improves accessibility for students and professionals.

With further enhancements such as AI-driven suggestions and cloud storage support, ResumeGen can evolve into a comprehensive career-support platform that assists users in preparing job-ready resumes quickly and accurately.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to our project guide, Prof. Rahul Lilhare, for his continuous guidance, valuable suggestions, and constant support throughout the development of this project. His technical insights and encouragement helped us in understanding the practical aspects of system design and implementation. We are also thankful to the Department of Computer Application, KDK College of Engineering, Nagpur, for providing the necessary resources and academic environment to successfully carry out this work. Finally, we extend our appreciation to our classmates and wellwishers who directly or indirectly contributed to the completion of this project.

REFERENCES

[1]. M. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed., O'Reilly Media, 2018.

[2]. Python Software Foundation, "Flask Documentation," Available: https://flask.palletsprojects.com

[3]. S. Sharma and R. Verma, "Design and Implementation of Online Resume Builder System," International Journal of Computer Applications, vol. 183, no. 21, pp. 15–20, 2021.

[4]. R. Singh and P. Kumar, "Web-Based Resume Generation Using Python Framework," in Proceedings of the International Conference on Computing Technologies, 2022, pp. 112–118.

[5]. Sommerville, Software Engineering, 10th ed., Pearson Education, 2016.

[6]. R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 8th ed., McGraw-Hill Education, 2015.
[7]. W3C, "HTML5 Specification," World Wide Web Consortium, 2018.
[8]. W3C, "Cascading Style Sheets (CSS) Level 3 Specification," World Wide Web Consortium, 2019.
[9]. M. Fowler, Patterns of Enterprise Application Architecture, Addison-Wesley, 2002.
[10]. A Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 6th ed., McGraw-Hill, 2011.