A PRELIMINARY PROJECT REPORT

ON

CAREER LENS- SMART CV ANALYZER AND BUILDER

FOR THE DEGREE OF

BACHELOR OF ENGINEERING

IN

COMPUTER ENGINEERING

SUBMITTED BY

Jayesh Bapurao Tupere
Shrutika Shahaji Desai
Pallavi Raosaheb Kharade
Sagar Sudam Pathare

UNDER THE GUIDANCE OF

Prof.VIKRAM CHAVAN

DEPARTMENT OF COMPUTER ENGINEERING

Sinhgad Technical Education Society′s

SINHGAD INSTITUTE OF TECHNOLOGY

Gat no. 309/310, Off Mumbai-Pune Expressway, Kusgaon (Bk.) Tal-Maval, Dist-Pune, Lonavala 410401

AFFILIATED TO

SAVITRIBAI PHULE PUNE UNIVERSITY

YEAR 2025-26

Sinhgad Technical Education Society′s

SINHGAD INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

THIS IS TO CERTIFY THAT THE PROJECT REPORT ENTITLED

CAREER LENS- SMART CV ANALYZER AND BUILDER

WHICH IS BEING SUBMITTED BY

1.Jayesh Bapurao Tupere
2.Shrutika Shahaji Desai
3.Pallavi Raosaheb Kharade
4.Sagar Sudam Pathare

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of Prof. A. B. C and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of Bachelor of Engineering (Computer Engineering)

Prof. Vikram Chavan
Project Guide

Dr. Shubhangi R. Patil
Head of Dept of Computer Engg.

Dr. S. D. Babar
Principal
SINHGAD INSTITUTE OF TECHNOLOGY

Place : Lonavala
Date :

Contents

List of Figures

List of Tables

Abstract

The increasing demand for efficient and intelligent recruitment systems necessitates tools capable of evaluating and improving resumes with precision and personalization. This project proposes **ResuGen**, a Generative AIâbased framework designed for automated resume analysis and creation. Leveraging Large Language Models (LLMs) and contextual parsing, ResuGen provides semantic understanding of both resumes and job descriptions, enabling accurate and explainable evaluation while maintaining low computational complexity. The system introduces a two-part architecture consisting of **Resume Analyzer** and **Resume Builder**.

The **Resume Analyzer** module allows users to upload their resume and job description, after which the backend parsing pipeline extracts structured data and processes it through an LLM-driven inference layer. The model generates comprehensive metrics such as ATS Score, Missing Skill Analysis, and Resume Summary, with future expansion toward chatbot-based improvement suggestions. The **Resume Builder** module enables registered users to generate resumes interactively through a structured form interface, dynamically formatting their inputs into professional templates.

Unlike traditional keyword-matching systems, ResuGen incorporates a GenAI-driven semantic layer that understands context, role requirements, and skill alignment, providing deeper and more adaptive insights. Implemented using the MERN stackâReact.js, TypeScript, Tailwind CSS for the frontend, and Node.js, Express.js, MongoDB for the backendâthe system ensures scalability, modularity, and efficiency. Designed for real-world applicability in digital recruitment and career development platforms, ResuGen demonstrates how GenAI and LLM-based automation can redefine resume evaluation and generation through intelligent, explainable, and user-centered innovation.

List of Abbreviations

LLM :	Large Language Model
GenAI :	Generative Artificial Intelligence
ATS :	Applicant Tracking System
XAI :	Explainable Artificial Intelligence
NLP :	Natural Language Processing
API :	Application Programming Interface
UI/UX :	User Interface / User Experience
JSON :	JavaScript Object Notation
MERN :	MongoDB, Express.js, React.js, Node.js
HTML :	HyperText Markup Language
CSS :	Cascading Style Sheets
CRUD :	Create, Read, Update, Delete
IDE :	Integrated Development Environment
DBMS :	Database Management System
CPU :	Central Processing Unit
UI :	User Interface
JSX :	JavaScript XML
SDK :	Software Development Kit

Chapter 1

INTRODUCTION

1.1 Overview

In today's competitive job market, both recruiters and job seekers face major challenges in resume management. Recruiters must filter through hundreds of resumes for a single position, while applicants often struggle to tailor their resumes effectively according to the job description. Most resumes fail to pass through Applicant Tracking Systems (ATS) due to missing keywords, poor formatting, and lack of alignment with job requirements.

The proposed system, **Resume Analyzer and Builder using GenAI and LLM**, aims to automate and simplify this process. The project is divided into two modules:

- **Resume Analyzer:** This module allows the user to upload a resume and a job description. The backend parses the resume, extracts structured data, and utilizes a **Large Language Model (LLM)** to evaluate the resume against the given job description. The system generates an *ATS score*, identifies *missing skills or keywords*, and provides a detailed *summary and suggestions* to improve the resume content.

- **Resume Builder:** This module provides a login-based form system for users to enter their personal, educational, and professional details. Once filled, the system automatically generates a formatted, professional-looking resume that can be downloaded in PDF format.

The system combines the power of **Generative AI (GenAI)** for intelligent text evaluation with modern web technologies such as **ReactJS, TypeScript, Tailwind CSS, NodeJS, ExpressJS, and MongoDB**. By integrating these components, the system delivers an intelligent, user-friendly, and scalable solution for modern resume creation and evaluation.

1.2 Motivation of the project

In the current competitive job market, an applicantâs resume often determines the first impression they make on recruiters. However, most students and job seekers lack proper guidance on creating effective resumes that align with job requirements. At the same time, recruiters face the challenge of reviewing thousands of resumes for a single job opening, making manual screening inefficient and prone to bias.

Recent advancements in **Generative AI (GenAI)** and **Large Language Models (LLMs)** have made it possible to understand and compare textual information with high accuracy. This capability can be leveraged to automatically evaluate resumes and provide meaningful feedback. The motivation behind this project is to use GenAI to bridge the gap between candidates and recruiters by offering intelligent, automated resume analysis and guided resume creation.

By combining resume parsing, LLM-driven evaluation, and real-time feedback, this project aims to help students, job seekers, and professionals build stronger resumes that are both human-readable and ATS-friendly. It also motivates candidates to understand where their resumes lack certain skills or experiences, allowing them to improve before applying.

Traditional resume screening methods are slow and prone to inconsistencies, while existing resume builders offer only static templates without personalized analysis. With the advent of **GenAI** and **LLMs**, it is now possible to build systems that understand the semantic meaning of job descriptions and resumes, providing detailed insights and recommendations.

This project was motivated by the idea of leveraging these modern technologies to build a smart system that:

- Helps candidates understand their resume strengths and weaknesses.
- Offers real-time feedback using LLM-based text evaluation.
- Simplifies the process of building a polished, ATS-friendly resume.

1.3 Problem definition

Recruitment processes involve significant manual effort for both applicants and recruiters. Candidates often struggle to tailor their resumes to specific job descriptions, while recruiters face the challenge of screening large numbers of resumes accurately. Many applicants remain unaware of how their resumes perform under ATS evaluation, leading to missed opportunities despite relevant skills.

The objective is to develop an intelligent web-based system that analyzes and evaluates resumes against job descriptions using **GenAI** and **LLM**-based models, while also allowing users to build professional resumes dynamically.

The system focuses on two core modules:

1. **Resume Analyzer:** Automatically parses and evaluates uploaded resumes, generates ATS scores, and identifies missing keywords or skills through LLM-powered text comparison.

2. **Resume Builder:** Enables users to log in, fill out structured resume forms, and generate visually appealing, ATS-compliant resumes.

The backend utilizes *NodeJS* and *ExpressJS* with data stored in *MongoDB*, while the frontend is developed using *ReactJS*, *TypeScript*, and *Tailwind CSS*. This architecture ensures scalability, security, and an efficient, user-friendly experience.

Chapter 2

LITERATURE SURVEY

Automated resume analysis and intelligent resume generation have gained significant attention in recent years due to the increasing digitalization of recruitment processes. Traditional hiring systems rely on manual screening and keyword-based filtering, which are often inconsistent and time-consuming. With the emergence of Generative AI and Large Language Models (LLMs), researchers are now focusing on context-aware evaluation systems that can understand the semantics and structure of resumes and job descriptions to improve candidate-job alignment.

AI-Based Resume Ranking and Skill Matching Systems

According to IEEE Xplore (2022) [1], early research introduced AI-driven ranking models that compare resumes with job descriptions using keyword extraction and vector similarity techniques. These models improved the accuracy of candidate shortlisting and reduced recruiter effort. However, the systems lacked contextual understanding and provided limited feedback to users, which restricted their effectiveness in modern, dynamic hiring environments.

NLP Techniques for Resume Parsing

Choudhary et al. (2023) [2], in Elsevier Expert Systems with Applications, implemented Natural Language Processing (NLP) techniques to parse and segment resumes into structured components such as education, experience, and skills. The study demonstrated improved information extraction from unformatted resumes. However, it did not employ contextual or LLM-based evaluation, which limited its adaptability to varied resume formats and job-specific nuances.

LLM-Based Resume and Job Matching

Singh and Deshmukh (2024) [3], in Springer Journal of Applied Computing, explored the use of Large Language Models for semantic text similarity between resumes and job descriptions. Their model outperformed keyword-based methods by understanding contextual relationships and improving candidate-job alignment. This research highlighted the potential of GenAI-based systems to transform traditional resume filtering into a more intelligent, meaning-driven process.

Automated Resume Generation through Web-Based Systems

Sharma and Patel (2023) [4], in IJIRSET, proposed an automated resume generation tool that allows users to input their personal and professional details through a web-based form. The system automatically formatted the input into a professional resume template, enhancing usability and accessibility. However, it lacked intelligent feedback mechanisms such as ATS scoring or suggestions for improvement, which modern job seekers often require.

Transformer-Based Contextual Evaluation Models

Verma and Rao (2024) [5], in ACM Digital Library, introduced a transformer-based contextual evaluation model for candidate-job fit analysis. The model utilized deep contextual embeddings to assess semantic relationships between job descriptions and resumes. This research demonstrated the potential of transformer architectures for deeper text understanding, forming a foundation for integrating LLMs into resume analysis systems.

Resume Classification using Large Language Models

Wang and Chen (2024) [6], in their arXiv publication âResumeAtlas,â presented a large-scale dataset and classification framework using LLMs for automated resume categorization and ranking. Their approach achieved state-of-the-art performance and provided explainability for decision-making through semantic embeddings, proving the scalability of LLM-powered resume intelligence.

Collectively, these studies demonstrate a technological shift from rule-based and keyword-dependent screening to context-aware, explainable, and adaptive GenAI-powered systems. The proposed project, ResuGen, builds upon these advancements by combining resume parsing, ATS scoring, missing skill detection, and resume generation into a single unified platform. Through LLM-driven evaluation and prompt-engineered GenAI pipelines, it aims to deliver a scalable, transparent, and intelligent solution for both candidates and recruiters in modern digital recruitment systems.

Chapter 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 Introduction

The **Resume Analyzer and Builder** system, titled **ResuGen**, is a Generative AIâpowered application designed to evaluate and generate resumes intelligently. The system automates resume analysis using Large Language Models (LLMs) to compare uploaded resumes against job descriptions, providing ATS score, missing skill analysis, and contextual insights. It also allows users to create their own resumes interactively through a dynamic web-based form. This chapter outlines the detailed software requirements specification for the project.

3.1.1 Project Scope

The project aims to simplify and improve the resume evaluation and creation process through automation, semantic analysis, and LLM-driven intelligence. ResuGen offers two major modules:

- **Resume Analyzer:** Parses uploaded resumes, evaluates them against job descriptions using LLM inference, and provides scores such as ATS Match Score, Missing Keywords, and Resume Summary.
- **Resume Builder:** Enables users to log in and create professional resumes interactively using structured form inputs that generate standardized templates automatically.

3.1.2 User Classes and Characteristics

- **Job Seekers:** Individuals who wish to evaluate or build resumes aligned with specific job profiles.

- **Recruiters / HR Professionals:** Users who can use the analyzer module for candidate screening and skill-matching.
- **System Administrator:** Responsible for maintaining the database, monitoring system performance, and managing user accounts.

3.1.3 Assumptions and Dependencies

- The system assumes the availability of a stable internet connection for API-based LLM interactions.
- It depends on a reliable backend database (MongoDB) for data storage and session management.
- Frontend and backend modules communicate through RESTful APIs.
- LLM models (e.g., OpenAI API or similar) will handle resume evaluation and natural language understanding.

3.2 Functional Requirements

3.2.1 System Feature 1 (Resume Analyzer Module)

- Users can upload a resume (PDF/DOCX) and a job description.
- The system parses the resume into structured text using a resume parser.
- The parsed content and job description are processed through an LLM for analysis.
- The system outputs:
  1. ATS Score
  2. Missing Skills and Keywords
  3. âTell Me About My Resumeâ Summary
  4. Interactive Chatbot (Future enhancement)

3.2.2 System Feature 2 (Resume Builder Module)

- Allows registered users to input details through a guided form (personal details, education, skills, experience, etc.).
- Dynamically generates a downloadable professional resume template.

- Provides options to edit or update resume sections anytime.
- Stores user data securely in the backend for future edits.

3.3 External Interface Requirements (If Any)

3.3.1 User Interfaces

- Web-based interface developed using React.js, TypeScript, and Tailwind CSS.
- Intuitive dashboard for uploading resumes and viewing analytics.
- Interactive form-based interface for building new resumes.

3.3.2 Hardware Interfaces

The system runs on standard computing devices (laptops/desktops) with internet connectivity. No additional hardware is required.

3.3.3 Software Interfaces

- **Frontend:** React.js, TypeScript, Tailwind CSS
- **Backend:** Node.js, Express.js
- **Database:** MongoDB
- **External APIs:** Generative AI/LLM model API for resume analysis

3.3.4 Communication Interfaces

- Frontend-backend communication via REST APIs.
- Secure data transfer using HTTPS protocol.
- JSON format for request-response payloads.

3.4 Nonfunctional Requirements

3.4.1 Performance Requirements

- The system should process a resume and return results within 10â15 seconds.
- Database queries and API calls should execute efficiently with minimal latency.

3.4.2 Safety Requirements

- Data backup mechanisms should be in place to prevent loss of user information.
- Uploaded files should be automatically removed after processing.

3.4.3 Security Requirements

- User authentication via secure login (JWT-based or OAuth).
- Resume data must not be shared or stored beyond the userâs consent.

3.4.4 Software Quality Attributes

- Reliability, Scalability, and Maintainability.
- Modular design for easy updates and future integration of new AI models.
- User-friendly interface with responsive design.

3.5 System Requirements

3.5.1 Database Requirements

- MongoDB database to store user credentials, resume data, and analytics.
- Efficient schema design for fast retrieval and updates.

3.5.2 Software Requirements (Platform Choice)

- Frontend: React.js, TypeScript, Tailwind CSS
- Backend: Node.js, Express.js
- Database: MongoDB
- Development Environment: Visual Studio Code

3.5.3 Hardware Requirements

- Minimum: Intel i5 processor, 8GB RAM
- Recommended: GPU-enabled system for faster LLM-based inference

3.6 Analysis Models: SDLC Model to be Applied

The project follows the **Incremental SDLC Model**. Each module â Resume Analyzer and Resume Builder â is developed, tested, and integrated in iterations. This model allows early feedback, reduces risks, and supports modular development with scalable upgrades.

3.7 System Implementation Plan

3.7.1 4.1 Project Estimates

The project estimates outline the anticipated effort, cost, time, and resources required for the successful implementation of the proposed ResuGen system. Since this project involves LLM integration, frontend-backend communication, and user interface development, estimates are calculated considering typical academic and small-scale development constraints.

Reconciled Estimates

Time Estimates: 12â14 weeks of total development and testing.
Cost Estimates: Minimal, as open-source technologies and free-tier GenAI APIs are used.

Project Resources

People: 3 developers (Frontend, Backend, Integration)
Hardware: Mid-range computing system with stable internet
Software/Tools: VS Code, MongoDB Atlas, Postman, Node.js, React.js, LLM API key (OpenAI/Anthropic)

3.7.2 4.2 Risk Management

This section discusses potential project risks and management approaches.

Risk Identification

- Delay in LLM API integration due to rate limits or model changes.
- Misalignment between resume parser output and LLM prompts.
- Performance lag in response time during peak usage.
- Security risks from improper file handling.

Risk Analysis

Each identified risk is monitored through regular testing, prompt validation, and efficient error handling. Periodic review meetings ensure mitigation measures are taken promptly.

3.7.3 4.3 Project Schedule

Project Task Set

- **Task 1: Data Parsing and Preprocessing** â Implement resume parsing logic and test multiple resume formats.
- **Task 2: LLM Integration and Scoring Mechanism** â Connect API endpoints and develop prompt templates for ATS and missing skill analysis.
- **Task 3: Resume Builder Implementation** â Create a multi-step resume builder form and PDF export feature.
- **Task 4: Testing and Optimization** â Validate results, improve accuracy, and ensure security compliance.

Task Network

Tasks are sequentially dependent, with front-end and back-end modules developed in parallel and integrated in later stages.

Timeline Chart

Project development spans approximately 12â14 weeks. The Gantt chart and detailed planner are attached in **Annex C (Project Planner)**.

3.7.4 4.4 Team Organization

Team Structure

- Project Lead â oversees planning, testing, and integration.
- Frontend Developer â builds UI/UX using React.js.
- Backend Developer â handles Node.js, Express.js, and database logic.

Management Reporting and Communication

Weekly progress reports are shared with the project guide. Team discussions are conducted twice a week to track updates and resolve blockers. Tools such as Google Meet and WhatsApp are used for collaboration.

Chapter 4

PROJECT PLAN

4.1 PROJECT ESTIMATES

The project estimates outline the anticipated effort, cost, time, and resources required for the successful implementation of the proposed **ResuGen** system. Since this project involves Generative AI integration, resume parsing, and full-stack web application development, the estimates are calculated considering typical academic-level development environments.

4.1.1 Reconciled Estimates

- **Time Estimates:** The total project duration is approximately 12â14 weeks, divided into phasesârequirement gathering, design, development, testing, and documentation.
- **Cost Estimates:** The project uses open-source technologies such as React.js, Node.js, and MongoDB, minimizing cost. The only expense involves temporary access to GenAI APIs (e.g., OpenAI) under free or student-tier plans.

4.1.2 Project Resources

Project Resources: The system utilizes academic-level hardware and software resources suitable for web-based AI applications.

- **People:** Three members â Project Lead, Frontend Developer, Backend Developer.
- **Hardware:** System with Intel i5 processor or higher, 8 GB RAM, and stable internet connection.
- **Software and Tools:** Node.js, React.js, Tailwind CSS, MongoDB Atlas, Visual Studio Code, Postman, GitHub, and an LLM API key for resume evaluation.

4.2 RISK MANAGEMENT

This section discusses the potential project risks and the approach to managing them, emphasizing Generative AI integration, system accuracy, and data handling.

4.2.1 Risk Identification

Risks are identified through review of scope documents, requirement specifications, and development schedules. The following questions guide risk evaluation:

1. Are all stakeholders and developers committed to the project timeline?
2. Are requirements and objectives clearly defined?
3. Are GenAI/LLM APIs reliable and accessible throughout the development?
4. Does the development team possess the required technical skills?
5. Are parsed resumes compatible with the LLM prompt format?
6. Are there sufficient safeguards for user data and privacy?
7. Is the system architecture scalable and maintainable?

4.2.2 Risk Analysis

Overview of Risk Mitigation, Monitoring, and Management:

- Continuous monitoring of LLM API reliability and fallback endpoints.
- Validation of resume parsing before AI processing.
- Weekly reviews of milestones to minimize schedule delays.
- Secure handling of uploaded resumes using temporary file storage.

4.3 PROJECT SCHEDULE

4.3.1 Project Task Set

Major project tasks under the Waterfall model are as follows:

- **Task 1: Data Parsing and Preprocessing** â Implement resume parser for PDF/DOCX files and convert into structured text.

- **Task 2: Model Integration and Implementation** â Integrate parsed text with LLM API for generating ATS Score, Missing Skills, and Resume Summary.
- **Task 3: Resume Builder Module** â Develop an interactive form-based resume builder using React.js with export functionality.
- **Task 4: Testing and Optimization** â Validate accuracy, ensure security, and finalize deployment.

4.3.2 Task Network

Project tasks and dependencies are sequential. Frontend and backend modules are developed in parallel, integrated during the later phase, and tested together for performance and stability.

4.3.3 Timeline Chart

A detailed timeline chart for the entire project is provided in **Annex C (Project Planner)**. It outlines all major milestones and task durations across 14 weeks.

4.4 TEAM ORGANIZATION

This section defines the team structure and reporting mechanisms used during the project.

4.4.1 Team Structure

- **Project Lead:** Oversees architecture, design validation, and testing.
- **Frontend Developer:** Builds the user interface using React.js and Tailwind CSS.
- **Backend Developer:** Handles API endpoints, MongoDB integration, and LLM communication.

4.4.2 Management Reporting and Communication

Weekly progress meetings are held with the project guide. Team members communicate via WhatsApp, Google Meet, and GitHub commits for code tracking. Documentation and weekly progress reports are maintained according to the lab assessment schedule.

Chapter 5

System Design

5.1 System Architecture

5.2 Data Flow Diagrams

5.3 Entity Relationship Diagrams

5.4 UML Diagrams

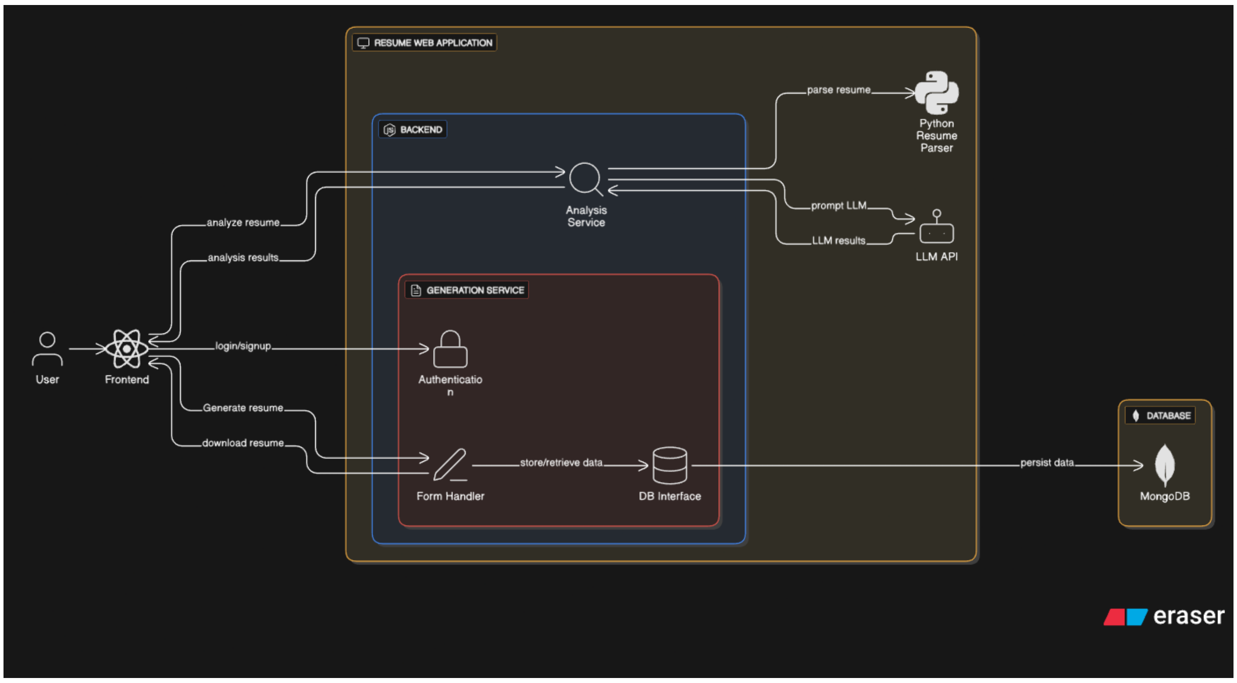

Figure 5.1: System Architecture

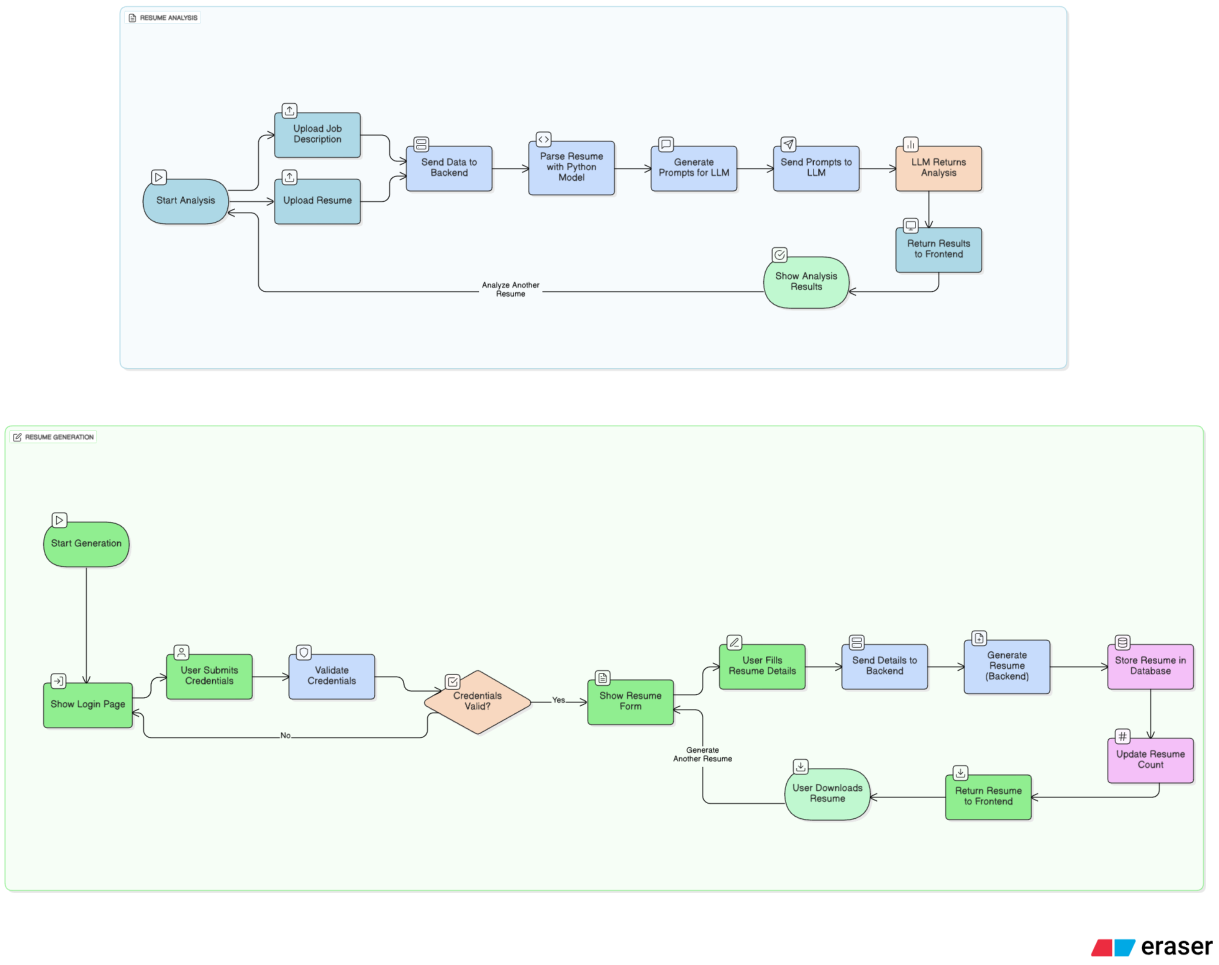

Figure 5.2: Data Flow Diagrams

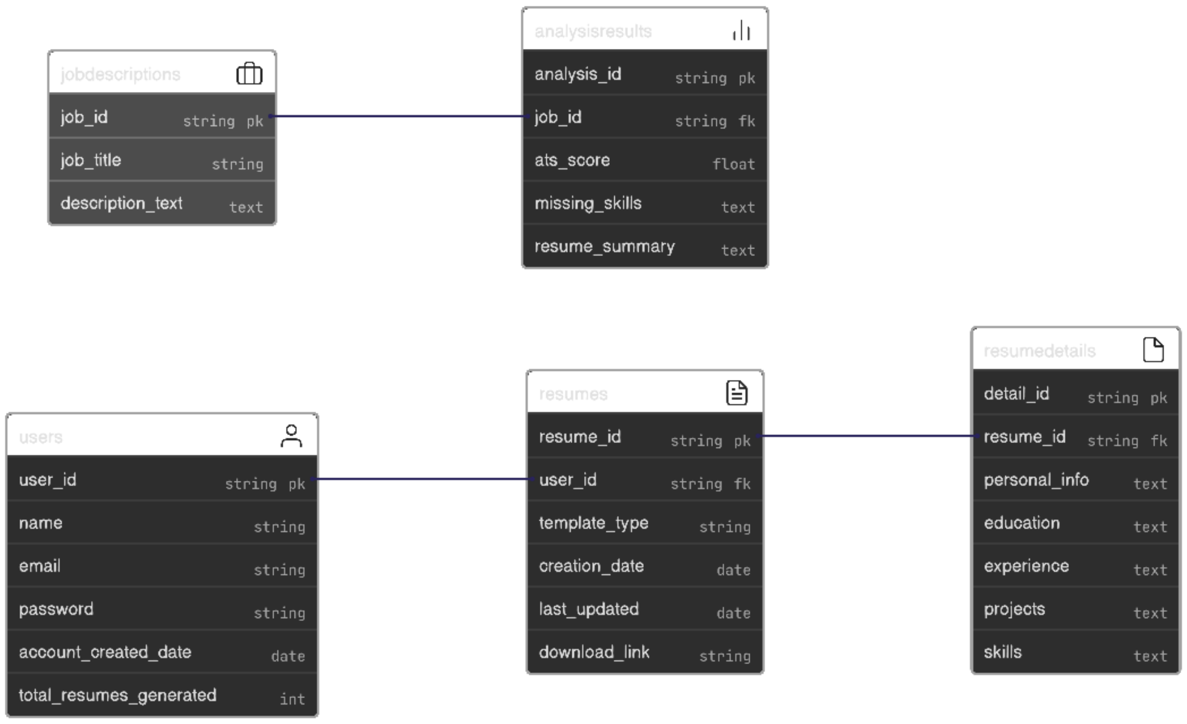

Figure 5.3: Entity Relationship Diagrams

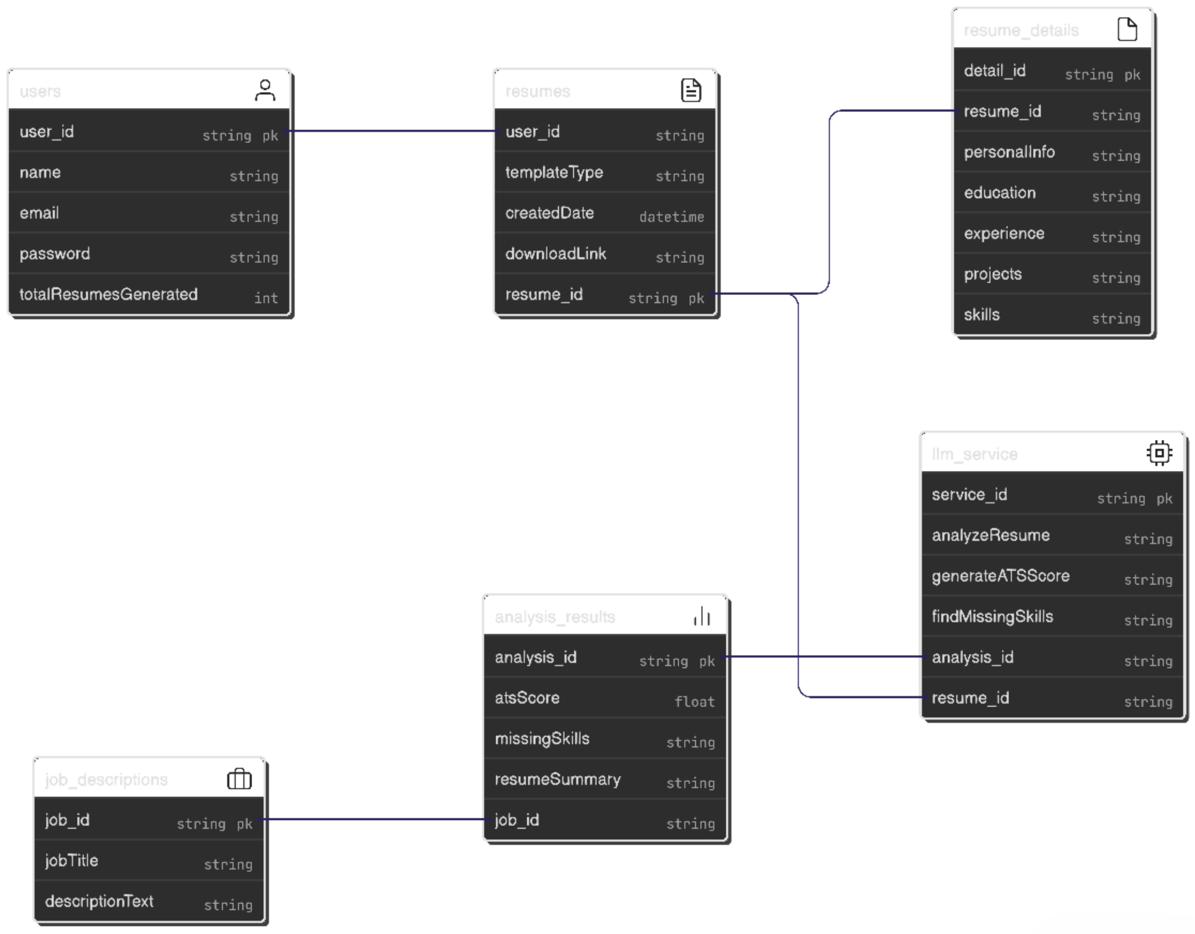

Figure 5.4: Enter Caption

Chapter 6

OTHER SPECIFICATION

6.1 Advantages

The proposed system **ResuGen** provides a modern, automated, and intelligent solution to resume evaluation and creation. It leverages Generative AI and Large Language Models to enhance the recruitment and resume-building experience for users. The key advantages include:

- **Automated Resume Evaluation:** Users can instantly analyze their resumes and receive detailed insights such as ATS Score, Missing Skills, and Resume Summary.
- **LLM-Powered Accuracy:** Unlike keyword-based systems, ResuGen uses contextual understanding through LLMs, providing more relevant and accurate evaluations.
- **User-Friendly Interface:** Built with React.js and Tailwind CSS, the interface is simple, fast, and responsive across all devices.
- **Time-Saving Process:** Users can both analyze and generate professional resumes quickly, eliminating the need for third-party resume services.
- **Secure and Scalable:** The system only stores necessary user data for resume building, ensuring privacy and scalability for future enhancements.
- **Customizable Resume Generation:** Users can select templates and modify sections easily before downloading.
- **Minimal Human Intervention:** The use of LLMs and automated parsing reduces manual screening and resume editing effort.

—

6.2 Limitations

Although ResuGen provides an efficient solution for resume management, certain limitations exist due to system constraints and dependencies:

- **LLM Dependency:** The accuracy and response quality depend on the performance and availability of external LLM APIs.
- **Internet Requirement:** The system requires a stable internet connection for backend-LLM communication and online authentication.
- **Limited Free Usage:** Extensive use of LLM APIs may require paid access if free-tier usage limits are exceeded.
- **No Offline Mode:** Resume analysis and generation functions are cloud-dependent and unavailable offline.
- **Parsing Variations:** Resume parsing accuracy may vary depending on document formatting or non-standard layouts.
- **Future Module Pending:** The chatbot/suggestion module is planned but not yet implemented in the current version.

—

6.3 Applications

The **ResuGen** system can be applied across multiple areas related to recruitment, job preparation, and academic use cases:

- **Placement Training:** Helps students evaluate and enhance resumes for campus placements and internships.
- **Corporate Recruitment:** Enables HR professionals to quickly assess candidate compatibility using ATS scoring and skill gap analysis.
- **Career Counseling:** As

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

The proposed system, **ResuGen â Resume Analyzer and Builder**, successfully demonstrates how Generative AI and Large Language Models (LLMs) can simplify and enhance the resume creation and evaluation process. It integrates two essential functionalities â intelligent resume analysis and dynamic resume generation â within a single, user-friendly platform.

The **Resume Analyzer** module enables users to upload their resumes and job descriptions for instant evaluation, generating valuable insights such as ATS Score, Missing Skills, and Resume Summary. By leveraging contextual understanding from LLMs, the system overcomes limitations of traditional keyword-based analysis, delivering more meaningful and personalized feedback.

The **Resume Builder** module allows users to create and download professional resumes through an interactive form-based interface. User data is securely stored and can be reused to track and manage multiple resume versions, offering flexibility and convenience.

Overall, ResuGen bridges the gap between automated resume screening tools and user-centric resume creation platforms. The system provides a strong foundation for AI-assisted job preparation tools and contributes to faster, smarter, and more effective recruitment workflows.

—

7.2 Future Work

While the current implementation of ResuGen achieves its primary objectives, several enhancements can further improve its scalability, performance, and user experience. Future improvements may include:

– **Chatbot Integration:** Implementing a conversational GenAI assistant to provide personalized career suggestions, resume tips, and real-time feedback.

– **Multi-Template Resume Generation:** Adding multiple professionally designed templates and layout customization options.

– **Enhanced Resume Parsing:** Incorporating advanced document understanding models for higher accuracy across diverse resume formats.

– **Offline or Hybrid Mode:** Allowing users to generate resumes without continuous internet dependency.

– **Role-Based Job Recommendation:** Using LLM insights to suggest suitable job roles based on resume content and missing skill analysis.

– **Analytics Dashboard:** Introducing a dashboard to visualize user performance metrics, resume quality trends, and improvement areas.

– **Enterprise Integration:** Expanding compatibility with HR management systems for large-scale recruitment automation.

—

Summary: The project demonstrates a practical use of GenAI and LLM technologies in solving real-world recruitment challenges. With planned improvements such as chatbot assistance, better parsing, and predictive job recommendations, **ResuGen** has the potential to evolve into a complete AI-driven career assistant platform â bridging the gap between candidates and opportunities.

REFERENCES

1. R. Mehta and P. Kulkarni, âAI-based Resume Ranking and Skill Matching System,â *IEEE Xplore*, 2022.
2. S. Choudhary, R. Das, and M. Banerjee, âNLP Techniques for Resume Parsing and Candidate Profiling,â *Expert Systems with Applications, Elsevier*, 2023.
3. V. Singh and A. Deshmukh, âIntelligent Job Recommendation using LLM-based Text Similarity,â *Springer Journal of Applied Computing*, 2024.
4. K. Sharma and N. Patel, âAutomated Resume Generation using Web-based Form Inputs,â *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, 2023.
5. A. Verma and D. Rao, âContext-Aware Candidate Evaluation using Transformer Models,â *ACM Digital Library*, 2024.
6. Y. Wang and M. Chen, âResumeAtlas: Revisiting Resume Classification with Large Language Models,â *arXiv Preprint*, 2024.
7. J. Devlin et al., âBERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,â *NAACL*, 2019.
8. OpenAI, âGPT-4 Technical Report,â *arXiv:2303.08774*, 2023.
9. J. Brownlee, âA Gentle Introduction to Transfer Learning for Deep Learning,â *Machine Learning Mastery*, 2020.
10. MongoDB Inc., âMongoDB Official Documentation,â [Online]. Available: https://www.mongodb
11. ReactJS Foundation, âReact Documentation,â [Online]. Available: https://react.dev/.