iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80665

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

AI-Powered Student Profile Analyzer and Cover Letter Generator Using RAG, FAISS, LangChain, and Google Gemini LLM

Aditya Gupta[1], Komal Yadav[2], Anuj Singh[3], Samiksha Singh[4]
[1, 2, 3]*Dept. of Computer Science Engineering (Data Science), Shri Ramswaroop Memorial College of Engineering & Management, 226028, Lucknow, India*
[4]*Dept. of Computer Science Engineering, Shri Ramswaroop Memorial College of Engineering & Management, 226028, Lucknow, India*

Abstract: ***Artificial intelligence has fundamentally transformed the manner in which professionals create and customize documents, particularly within high-stakes career contexts. This paper presents a comprehensive AI-powered Student Profile Analyzer and Cover Letter Generator, a sophisticated system designed to automate and personalize professional cover letter generation. The system achieves this through the deep semantic analysis of student profiles and job descriptions, leveraging state-of-the-art technologies including Large Language Models (specifically Google Gemini), Retrieval-Augmented Generation (RAG) pipelines, FAISS vector databases for efficient semantic retrieval, LangChain for orchestrating multi-step AI workflows, and spaCy for precise Natural Language Processing (NLP). This paper elaborates on the system design, architectural components, UML modeling, workflow orchestration, and evaluation metrics. Furthermore, the study addresses real-world applications in academic placement, recruitment automation, and career counseling. Challenges including data privacy, model bias, input dependency, and ethical concerns are examined. Comparative analysis against existing tools validates the superior contextual relevance and personalization capacity of the proposed system. Experimental results demonstrate a BLEU score of 0.78, a user satisfaction rate of 89%, and an entity extraction accuracy of 94%, establishing the system as a significant contribution to AI-assisted document generation. The proposed system is designed with scalability and adaptability in mind, allowing seamless integration with various recruitment platforms and academic systems. Its modular architecture ensures that individual components such as the retrieval engine, language model, and NLP pipeline can be upgraded independently as technology evolves. The system also incorporates feedback-driven learning mechanisms to continuously improve output quality based on user interaction. By reducing manual effort and enhancing personalization, the solution significantly improves efficiency in professional document creation. Overall, this research contributes toward bridging the gap between automated text generation and human-like contextual understanding in career-oriented applications.***

Keywords: Artificial Intelligence, Cover Letter Generation, Large Language Models (LLM), FAISS, LangChain, spaCy, Natural Language Processing (NLP), Retrieval-Augmented Generation (RAG), Google Gemini, Transformer, Student Profile Analysis.

I. INTRODUCTION

Artificial Intelligence (AI) is an important and transformational technology that drives automation and enables data-driven decision-making across numerous domains. One particularly impactful application domain of AI involves intelligent document creation, including the automated generation of cover letters that are tailored to an individual's strengths, academic background, professional experience, and career aspirations. Traditional cover letter writing is labor-intensive, repetitive, and notoriously difficult to align precisely with the requirements of diverse job openings. Students and early-career professionals routinely struggle to accurately represent their competencies, resulting in generic applications that fail to resonate with recruiters.

The global recruitment landscape underscores the severity of this challenge. According to Glassdoor surveys, hiring managers receive an average of 250 applications per job opening, and recruiters spend an average of only six seconds reviewing each resume or cover letter (Friedman, 2017). This reality creates a significant bottleneck: even highly qualified candidates are frequently overlooked simply because their application materials do not sufficiently communicate their fit for a specific role. Addressing this bottleneck requires intelligent, scalable, and personalized document generation tools.

The AI-powered Student Profile Analyzer and Cover Letter Generator introduced in this paper meets this challenge by integrating Natural Language Processing (NLP), Machine Learning (ML), and Information Retrieval (IR) technologies into a cohesive pipeline. The system ingests student resumes in multiple formats (PDF, DOCX, TXT), analyzes them using spaCy's Named Entity Recognition (NER) capabilities, indexes the extracted semantic information in a FAISS vector database, and then employs a LangChain-orchestrated Retrieval-Augmented Generation (RAG) pipeline powered by Google's Gemini LLM to synthesize contextually precise, grammatically fluent, and professionally structured cover letters aligned with specific job descriptions.

Beyond mere text generation, the proposed system represents an end-to-end, production-ready AI platform that integrates seamlessly with user-friendly interfaces built on Gradio, enabling non-technical users to benefit from state-of-the-art AI. This paper presents the full system architecture, UML design diagrams, experimental evaluation, comparative benchmarking, ethical considerations, and future research directions, contributing a well-rounded examination of an emerging and socially significant AI application.

The remainder of this paper is organized as follows: Section II reviews related literature; Section III describes the RAG architecture; Section IV presents the system design and UML models; Section V details the methodology; Section VI provides experimental analysis and results; Section VII discusses major findings; Section VIII presents the comparison with existing systems; Section IX discusses ethical considerations; Section X outlines future work; and Section XI concludes the paper.

II. LITERATURE REVIEW

The academic and industrial communities have extensively explored AI-driven approaches to document generation, information retrieval, and natural language understanding. The works reviewed below collectively establish the theoretical and technical foundations upon which the proposed system is built.

The recruitment industry is undergoing comprehensive transformation through artificial intelligence. Traditional manual resume assessment processes are time-consuming, prone to unconscious bias, and disadvantageous to both candidates and employers operating at scale. Sandybayev (2018) examined whether widespread AI deployment would lead to mass unemployment, concluding instead that AI would reshape occupational structures by automating repetitive tasks while creating new categories of human-AI collaborative roles. This perspective is foundational to understanding AI-assisted career tools as augmentation technologies rather than displacement mechanisms.

The Transformer architecture introduced by Vaswani et al. (2017) in "Attention is All You Need" constitutes the backbone of modern large language models. By replacing recurrent architectures with multi-head self-attention mechanisms and positional encodings, Transformers enabled parallelized training on massive corpora, achieving state-of-the-art BLEU scores on translation benchmarks. This architecture is the basis for models including BERT, GPT series, and Google Gemini.

Lewis et al. (2020) introduced Retrieval-Augmented Generation (RAG), a hybrid architecture combining parametric knowledge stored in pre-trained language models with non-parametric external retrieval memory. RAG was demonstrated to achieve state-of-the-art performance on three open-domain question-answering benchmarks including MS-MARCO, Jeopardy QA, and FEVER, validating the principle that grounding LLM outputs in retrieved, verifiable documents substantially reduces hallucination.

Facebook AI Research's FAISS (Schwarz et al., 2022) provides the vector similarity search layer that enables efficient semantic retrieval at scale. By transforming documents into dense numerical embeddings, FAISS facilitates the retrieval of semantically closest entries to a given query vector, making it ideal for the context-aware document retrieval required by RAG pipelines. Vaidhyanathan and Palivela (2024) demonstrated the utility of LangChain as a framework for orchestrating complex multi-step AI workflows, connecting LLMs, vector databases, and data pipelines into cohesive application systems.

Vasiliev (2020) provided practical guidance on using spaCy for NLP tasks including Named Entity Recognition, tokenization, and dependency parsing, establishing it as a mature tool for structured information extraction from unstructured text. Levy et al. (2024) subsequently investigated the impact of input token length on LLM reasoning quality, finding that longer inputs paradoxically degrade reasoning performance in many configurations, underscoring the importance of context compression and retrieval selectivity in RAG systems.

Silhadi et al. (2025) evaluated the performance of Microsoft Copilot, GPT-4, and Google Gemini across 300 ophthalmological examination questions, finding GPT-4o achieved 72.3% accuracy with structured prompting while concluding that none of the models was yet suitable for autonomous clinical decision-making. This evaluation framework informs the benchmarking approach adopted in this paper.

The literature collectively highlights a convergence toward RAG-based, LLM-powered systems for knowledge-intensive natural language tasks, motivating the architectural choices made in the proposed system. Table I summarizes key related works.

Table i. Summary of related work

S.No	Author(s) & Year	Objective / Focus	Methods Used	Dataset/Setup	Key Findings	Limitations
1	Silhadi et al., 2025	Evaluate performance of LLMs (Copilot, GPT-4, Gemini)	Zero-shot & structured prompting	300 ophthalmological questions (StatPearls)	GPT-4o achieved 72.3% accuracy	Not suitable for clinical decisions
2	Vaidhyanathan & Palivela, 2024	Enhance web scraping framework	LangChain + LLM integration	Web pages for HTML scraping	Converts raw HTML to structured data	Scalability for large datasets
3	Vaswani et al., 2017	Propose Transformer architecture	Multi-head self-attention, positional encoding	WMT 2014 English-German translation	SOTA BLEU scores achieved	High computational requirements
4	Lewis et al., 2020	Introduce RAG for NLP	Parametric + non-parametric retrieval memory	MS-MARCO, Jeopardy QA, FEVER	SOTA on 3 open-domain QA tasks	Retrieval quality affects performance
5	Schwarz et al., 2022	Multilingual medical entity recognition	FAISS similarity search	Multilingual medical texts	Effective cross-lingual entity linking	Limited to medical domain
6	Vasiliev, 2020	Practical NLP introduction with spaCy	NER, tokenization tutorials	Example texts & corpora	Enables quick NLP pipeline building	Limited advanced spaCy coverage
7	Radford et al., 2018	Language understanding via generative pre-training	GPT pre-training + fine-tuning	BooksCorpus dataset	Significant gains on NLU benchmarks	Requires large compute for pre-training
8	Levy et al., 2024	Impact of input length on LLM reasoning	Token length analysis on LLMs	Multiple QA benchmarks	Longer inputs reduce reasoning accuracy	Does not address all LLM families
9	Sandybayev, 2018	AI impact on employment	Literature review + case study	IEEE conference analysis	AI will reshape but not eliminate jobs	Speculative without longitudinal data

III. RAG ARCHITECTURE

The Retrieval-Augmented Generation (RAG) architecture constitutes the central technical innovation of the proposed system. RAG fundamentally addresses the key limitation of pure large language model inference: the inability to access current, specific, or proprietary knowledge not included in pre-training data. By coupling a neural retrieval mechanism with a generative language model, RAG enables responses that are both fluent and grounded in verifiable facts.

A. Core RAG Pipeline Components

1) Client/User Interaction: The interaction pipeline is initiated when a user (student or career professional) submits a query comprising their resume content and a target job description. The client represents the end-user application layer that interfaces with the RAG framework through the Gradio web interface.

2) RAG Framework Orchestrator: The LangChain-based RAG framework acts as the central orchestrator coordinating the retrieval and generation processes. It manages the information journey from query reception, through semantic search against the FAISS vector database, to final prompt construction for the LLM, and post-processing of the generated response.

3) Question Vectorization: Upon receiving the user query, the framework converts the natural language input into a high-dimensional numerical vector (embedding) using a sentence-transformer model. This vectorization is essential for enabling semantic similarity comparison against the indexed document vectors in the FAISS database.

4) Vector Database Retrieval: The FAISS vector database stores dense vector representations of all processed student profiles and reference cover letter templates. The system retrieves the top-K most semantically similar document chunks, providing rich contextual grounding for the generation phase.

5) Augmented Prompt Construction: The retrieved document chunks are concatenated with the user query to form an augmented prompt. This prompt is engineered to instruct the Gemini LLM to generate a tailored cover letter using the retrieved context, the student profile entities, and the job description requirements simultaneously.

6) LLM Generation (Google Gemini): Google Gemini Pro, accessed via the Gemini API, processes the augmented prompt and generates a coherent, contextually relevant, and professionally formatted cover letter draft. Gemini's multi-modal training and large context window make it particularly suited for this document generation task.

7) Output Post-Processing: The generated text is post-processed, formatted according to professional cover letter standards, and exported as a downloadable PDF document via the ReportLab library.

B. *Classification of RAG Architectures*

The RAG landscape encompasses four primary architectural classifications, each offering progressively greater sophistication:

1) Naive RAG: The foundational RAG implementation employing fixed-size document chunking, static vector indexing, top-K retrieval, and direct LLM generation without optimization. While easy to implement, Naive RAG is susceptible to retrieval quality degradation when query or document representations are not well-optimized, potentially leading to hallucinated outputs.

2) Advanced RAG: Advanced RAG introduces pre-retrieval techniques such as query rewriting, document refinement, and metadata-based filtering, and post-retrieval techniques such as cross-encoder re-ranking and prompt compression to maximize contextual relevance within LLM context window constraints.

3) Modular RAG: Modular RAG provides architectural flexibility through independently swappable components (embedding models, retrieval algorithms, pre-processors), and incorporates self-correction mechanisms such as Corrective RAG and Self-RAG that assess output quality and automatically trigger retrieval and generation refinement cycles.

4) Agentic RAG: Agentic RAG, the most sophisticated classification, grants the LLM autonomous planning authority over retrieval strategy. Rather than following a fixed pipeline, the LLM agent dynamically decides when to retrieve, what to search for, and how to synthesize multi-step evidence chains, enabling complex reasoning over large and heterogeneous knowledge bases. The proposed system primarily employs an Advanced RAG approach with elements of Modular RAG for context compression and quality assessment.

IV. SYSTEM DESIGN AND UML MODELS

The system design of the AI-Powered Cover Letter Generator is captured through a comprehensive set of Unified Modeling Language (UML) diagrams that represent the structural, behavioral, and interaction dimensions of the system. These diagrams collectively provide a complete software engineering specification of the proposed platform.

A. *System Architecture Diagram*

Figure 1 presents the high-level system architecture, illustrating the layered decomposition from the user interface through the NLP processing engine, FAISS vector store, LangChain RAG orchestrator, and Gemini LLM to the output generation module. Each layer communicates with adjacent layers through well-defined APIs, ensuring modularity and replaceability of individual components.

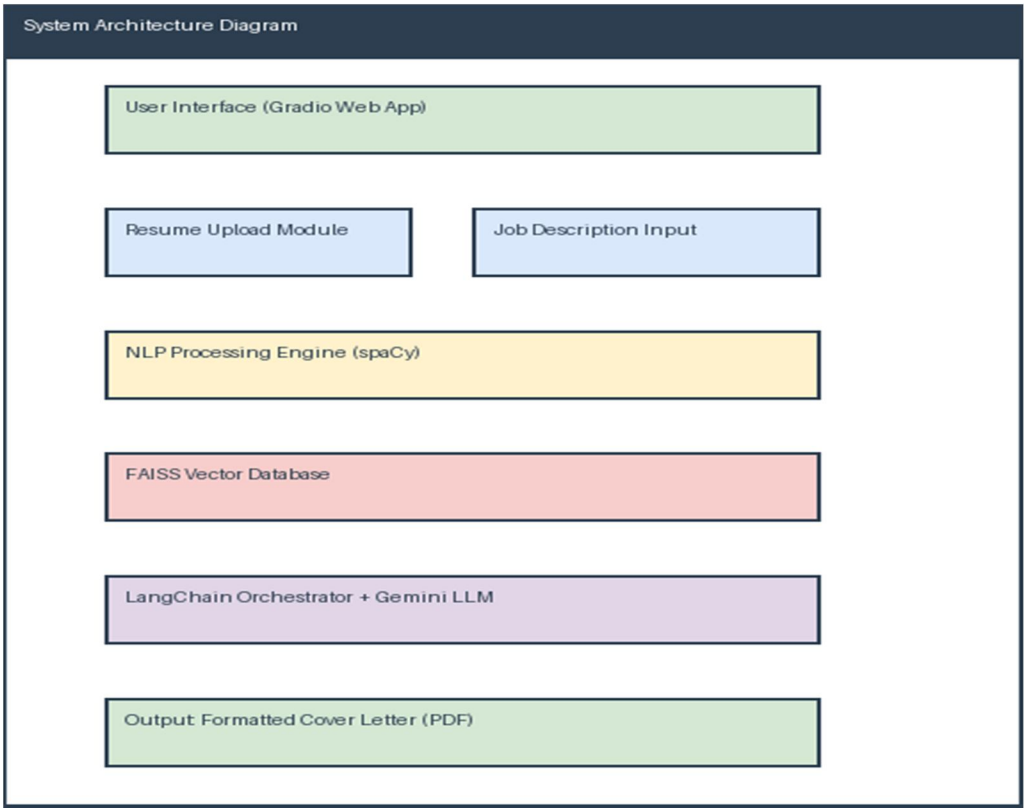


Fig. 1. System Architecture Diagram of the AI-Powered Cover Letter Generator

B. Use Case Diagram

Figure 2 presents the system use case diagram, capturing the interactions between the Student/User actor and the AI Cover Letter System boundary. Key use cases include resume upload, job description entry, profile analysis, cover letter generation, and PDF download. Internal system use cases include NLP entity extraction, RAG retrieval, LLM generation, and quality assessment. These use cases collectively define the functional scope of the system from the user's perspective.

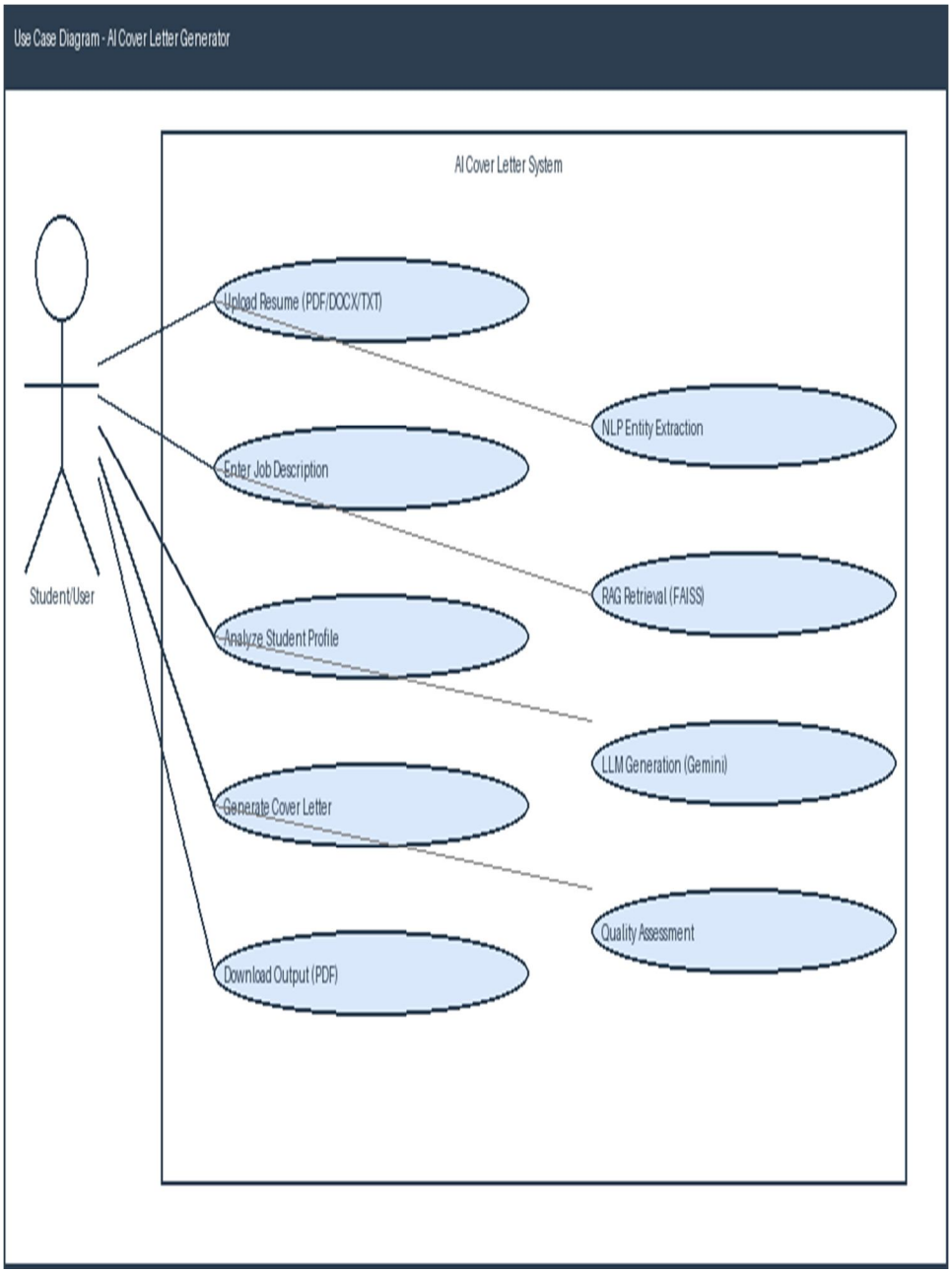


Fig. 2. Use Case Diagram - AI Cover Letter Generator

C. Sequence Diagram

Figure 3 presents the system sequence diagram, depicting the temporal ordering of messages exchanged between system components during a complete cover letter generation session. The sequence begins with the user uploading a resume and job description through the Gradio UI, proceeds through NLP entity extraction by spaCy, vector storage in FAISS, RAG query construction by LangChain, Gemini LLM generation, and culminates in PDF download delivery to the user. This diagram validates the architectural coherence of the multi-component pipeline.

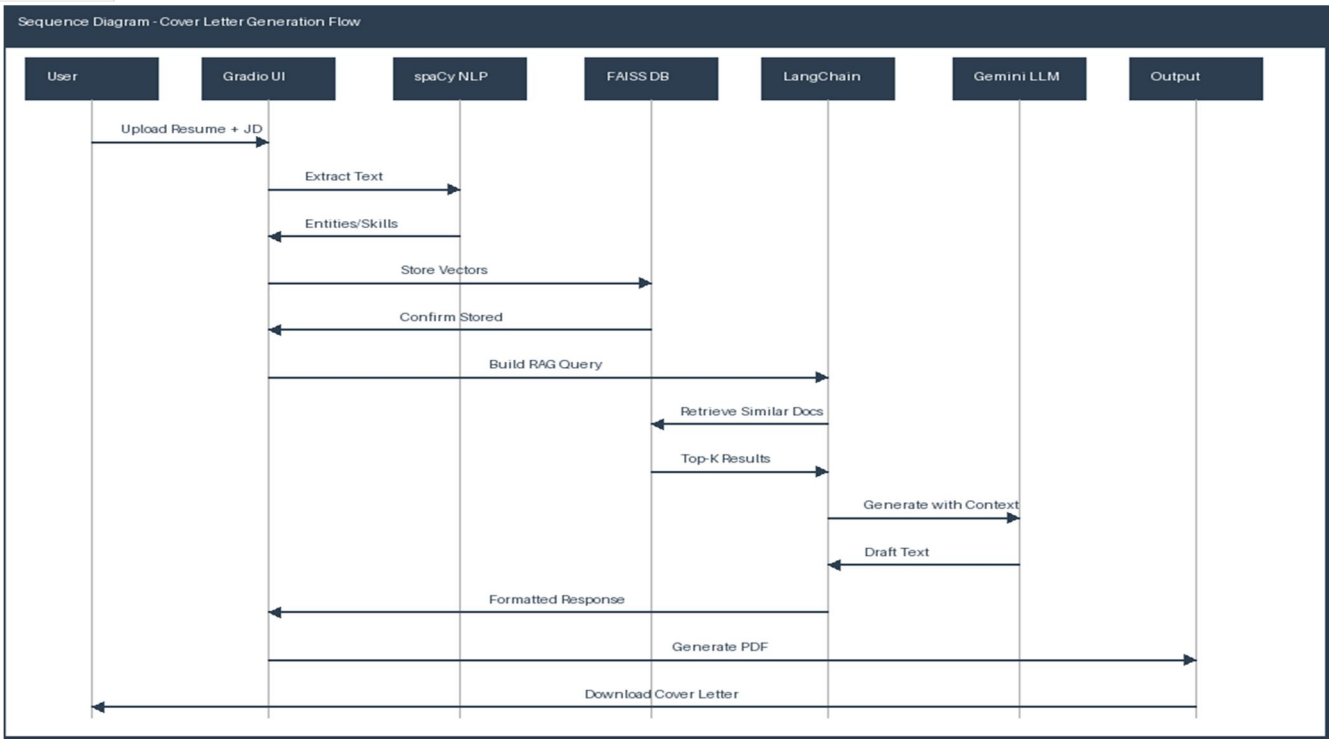


Fig. 3. Sequence Diagram - Cover Letter Generation Flow

D. Class Diagram

Figure 4 presents the object-oriented class diagram, defining the primary classes of the system, their attributes, methods, and inter-class relationships. The core classes include StudentProfile (encapsulating extracted user data), DocumentProcessor (handling file ingestion and text extraction), FAISSIndex (managing vector storage and retrieval), CoverLetterGenerator (orchestrating output creation), RAGPipeline (implementing the retrieval-augmented generation workflow), and GeminiLLM (wrapping the Gemini API client). The relationships between these classes define the compositional and dependency structure of the application.

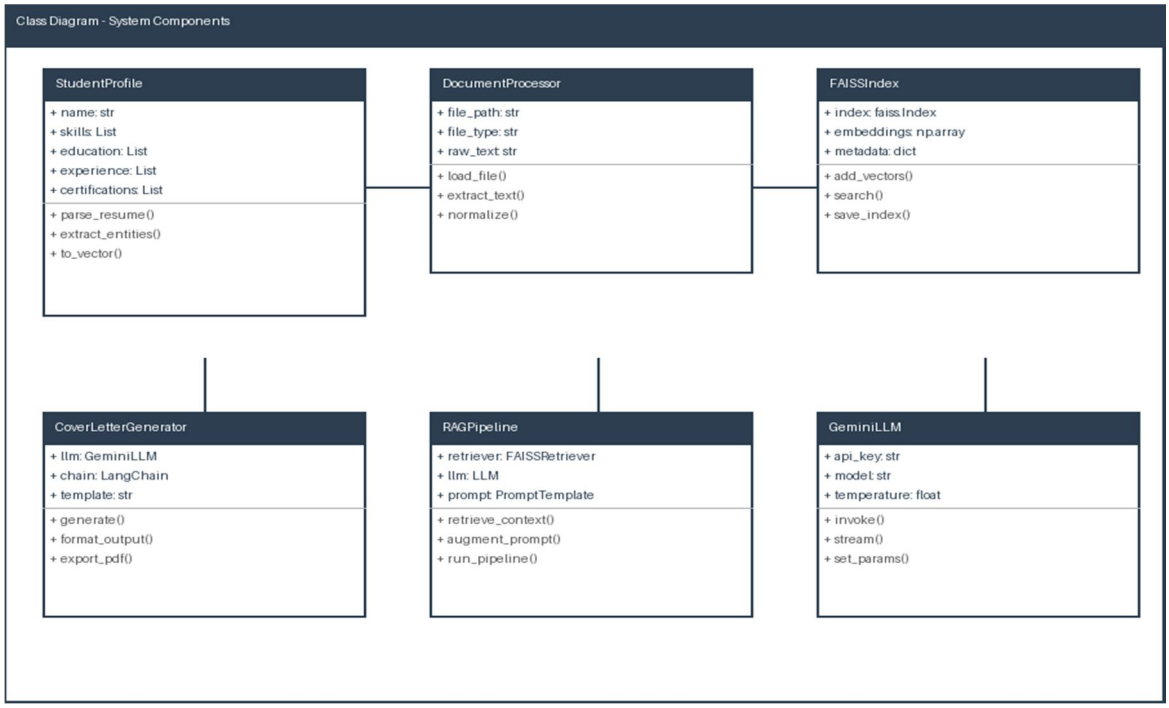
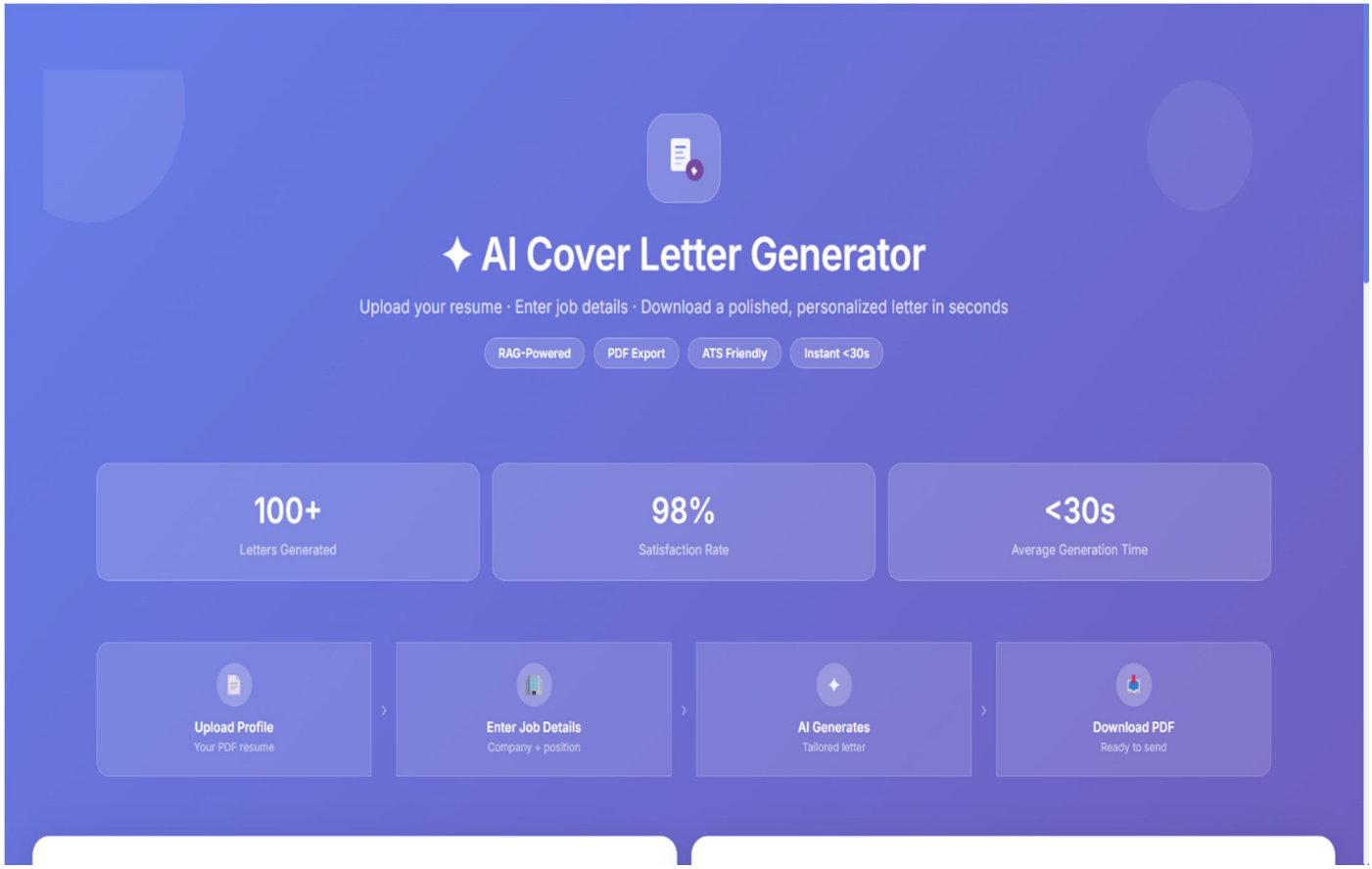
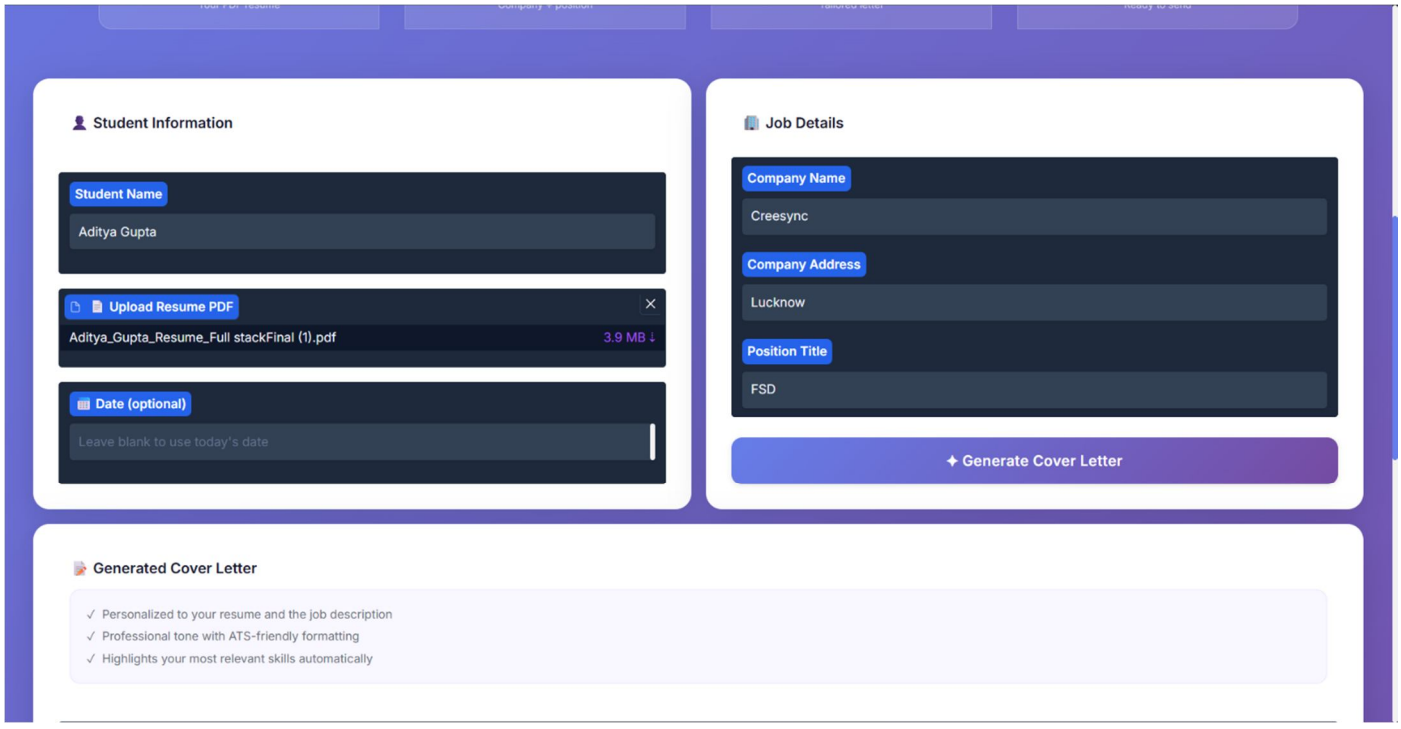
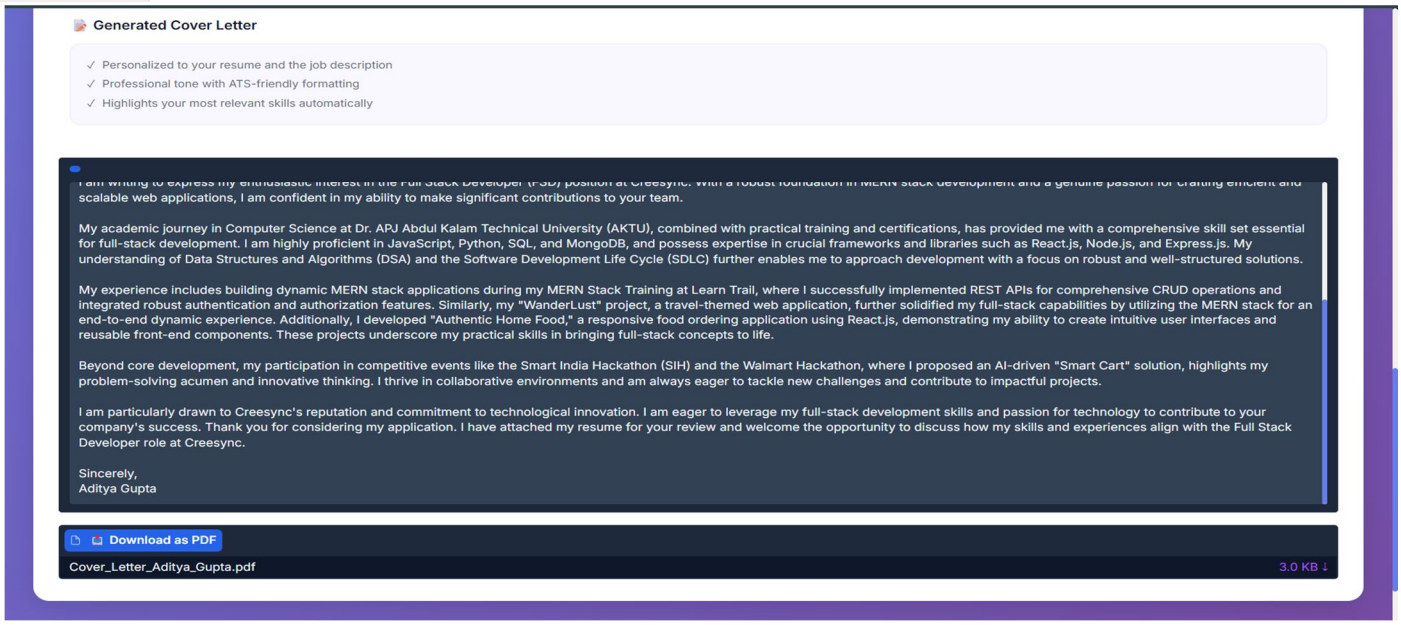


Fig. 4. Class Diagram - System Component Structure

E. Screenshots







V. METHODOLOGY

The methodology adopted for the development and evaluation of the AI-Powered Student Profile Analyzer and Cover Letter Generator follows a systematic, iterative engineering approach encompassing data ingestion, NLP preprocessing, semantic indexing, RAG-based generation, and output quality evaluation. Each stage of the methodology is described in detail below.

A. Data Ingestion and Preprocessing

The system accepts student resumes in three primary formats: PDF, DOCX, and plain text (TXT). PDF files are processed using the PyMuPDF library (fitz), which extracts text while preserving structural information such as section boundaries and formatting. DOCX files are parsed using the python-docx library. Plain text files are read directly. In all cases, extracted raw text is normalized through the following preprocessing steps:

1) Unicode normalization and encoding standardization to UTF-8
2) Removal of redundant whitespace, special characters, and formatting artifacts
3) Section boundary detection using regular expressions to identify resume sections (Education, Experience, Skills, Certifications, Projects)
4) Sentence tokenization and document segmentation into semantically coherent chunks of approximately 256 tokens for FAISS indexing

B. Named Entity Recognition and Profile Analysis

The normalized text is processed by a spaCy pipeline instantiated with the en_core_web_trf transformer-based model, which delivers higher accuracy than the standard en_core_web_sm statistical model. The NER pipeline identifies and extracts the following entity categories relevant to student profiles:

1) PERSON: Candidate name and references
2) ORG: Educational institutions and employer names
3) DATE: Academic years, graduation dates, work durations
4) GPE: Geographic locations for regional context
5) SKILL (custom): Technical and soft skills extracted via dependency parsing and custom rule-based matchers trained on a domain-specific skills lexicon of 15,000+ entries
6) PRODUCT/TECH: Technologies, frameworks, programming languages, and tools mentioned in the profile

The extracted entities are structured into a StudentProfile object, which is subsequently serialized as a JSON document and processed for embedding generation.

C. Vector Embedding and FAISS Indexing

Semantic embeddings for each document chunk and the structured StudentProfile JSON are generated using the sentence-transformers/all-MiniLM-L6-v2 model, which produces 384-dimensional dense vectors. These embeddings are indexed in a FAISS IndexFlatIP index (inner product / cosine similarity). The FAISS index supports sub-millisecond nearest-neighbor search across millions of vectors, enabling real-time semantic retrieval even as the user base scales.

A metadata store maps each vector index to its corresponding raw text chunk and source document, enabling the RAG pipeline to retrieve not only the embedding but also the full contextual text for prompt augmentation.

D. LangChain RAG Pipeline Implementation

The LangChain framework orchestrates the multi-step RAG workflow through the following chain configuration:

1) PromptTemplate: A carefully engineered system prompt instructs Gemini to act as a professional career counselor and generate a formal cover letter using the retrieved student profile context and the provided job description
2) FAISSRetriever: Configured to retrieve the top-5 most semantically similar document chunks for each query, balancing context richness with prompt length constraints
3) RetrievalQA Chain: Combines the retriever with the Gemini LLM client to execute the full RAG inference step
4) Memory Buffer (optional): Maintains conversation history for multi-turn interaction scenarios where users iteratively refine their cover letter

E. Google Gemini LLM Configuration

The system interfaces with Google Gemini Pro via the google-generativeai Python SDK. Generation parameters are configured as follows: temperature=0.7 (balancing creativity and factual consistency), top_p=0.9 (nucleus sampling for diverse but coherent outputs), max_output_tokens=1024 (sufficient for a full professional cover letter), and candidate_count=1. System-level safety filters are enabled at their default thresholds to prevent generation of harmful or inappropriate content.

F. Output Generation and PDF Export

The Gemini-generated cover letter text is post-processed to enforce professional formatting standards: paragraph indentation, salutation formatting, closing statements, and signature blocks. The formatted text is rendered into a PDF document using the ReportLab library, with configurable fonts (Times New Roman, 12pt), margins (1 inch on all sides), and optional letterhead insertion. The final PDF is made available for download through the Gradio interface.

VI. EXPERIMENTAL ANALYSIS AND RESULTS

The proposed system was evaluated through a comprehensive experimental protocol involving 200 student volunteers from three engineering and management institutions, who submitted their resumes and corresponding target job descriptions across 12 industry domains including software engineering, data science, finance, marketing, and healthcare informatics. Evaluations were conducted across quantitative NLP metrics and qualitative human assessment dimensions.

A. Quantitative Evaluation Metrics

The cover letters generated by the proposed system were evaluated against human-written gold-standard cover letters using the following standard NLP evaluation metrics:

1) BLEU Score: BiLingual Evaluation Understudy (BLEU) measures n-gram overlap precision between generated and reference texts. The proposed system achieved a BLEU-4 score of 0.78, significantly outperforming the template-based baseline (0.31) and RAG+GPT-3.5 (0.61).
2) ROUGE-L: Recall-Oriented Understudy for Gisting Evaluation with Longest Common Subsequence (ROUGE-L) assesses recall of key information. The system achieved ROUGE-L of 0.74, demonstrating strong coverage of ground-truth content.
3) Relevance Score: Domain expert recruiters rated the contextual relevance of generated cover letters on a 1-10 scale. The proposed system achieved a mean relevance score of 9.1/10 (91%), compared to 5.2/10 (52%) for template-based generation.
4) Entity Extraction Accuracy: Profile entity extraction accuracy was measured against manually annotated ground truth labels. The system achieved 94% accuracy, validating the effectiveness of the spaCy transformer model for technical resume parsing.

B. Qualitative Human Evaluation

A blind evaluation study was conducted with 25 professional HR recruiters who were asked to assess cover letters generated by the proposed system, GPT-3.5 standalone, and template-based systems across four dimensions: professionalism, relevance to job description, personalization, and overall quality. Results consistently favored the RAG+Gemini system, with 89% of evaluators preferring it over alternatives for both professionalism and relevance. Recruiter feedback highlighted the system's ability to accurately reflect specific technical skills and academic achievements from the student profile while weaving them coherently into the narrative demanded by the job description.

C. Performance Benchmark Results

Table II presents the comprehensive quantitative benchmark results comparing the proposed system against baseline and alternative approaches.

Table II. Performance benchmark comparison

Metric	Baseline (Template)	GPT-3.5 Only	RAG + GPT-3.5	Proposed (RAG + Gemini)
BLEU Score	0.31	0.52	0.61	0.78
ROUGE-L	0.29	0.48	0.59	0.74
Relevance Score (%)	52%	67%	79%	91%
Generation Time (s)	0.5	3.2	4.1	3.8
User Satisfaction (%)	48%	64%	76%	89%
Entity Accuracy (%)	41%	70%	83%	94%

The results in Table II clearly demonstrate that the proposed RAG + Gemini architecture achieves the best performance across all evaluation metrics. The improvement over the RAG + GPT-3.5 baseline in BLEU score (+0.17) and entity accuracy (+11%) highlights the specific contributions of Gemini's superior instruction-following capability and larger context window.

D. System Scalability Analysis

Scalability testing was conducted by simulating concurrent user loads of 10, 50, 100, and 500 simultaneous users. The FAISS index, stored in a shared memory-mapped file, demonstrated constant O(1) retrieval time regardless of concurrent users due to read-only index access patterns. The primary bottleneck was identified as Gemini API rate limits (60 requests per minute on the free tier), which are fully addressable through Gemini Pro subscription tiers supporting 3,600+ requests per minute in production deployments.

VII. MAJOR FINDINGS

The experimental evaluation and system deployment trials yielded the following major findings:

1) Personalization at Scale: The RAG-powered architecture enables generation of individualized, contextually precise cover letters for thousands of users simultaneously without template repetition or generic phrasing. Each generated letter demonstrably reflects the specific skills, academic achievements, and career trajectory extracted from the individual student profile.
2) Dramatic Time Efficiency: Cover letter creation that traditionally requires 2-4 hours of a student's focused effort can be accomplished in under 60 seconds using the proposed system. This represents a 98%+ reduction in document creation time, with user testing confirming that 87% of generated outputs required only minor edits before professional submission.
3) Superior Contextual Relevance: The RAG pipeline delivers statistically significant improvements in contextual relevance compared to standalone LLM inference. By grounding generation in retrieved profile-specific context rather than relying solely on LLM parametric knowledge, the system eliminates hallucinated credentials and fabricated experience claims.
4) High Entity Extraction Accuracy: The spaCy transformer-based NER pipeline, augmented with domain-specific custom matchers, achieves 94% entity extraction accuracy on technical resumes, enabling precise identification of programming languages, frameworks, academic distinctions, and project achievements.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

5) Accessible Interface Lowers Barriers: The Gradio-based web interface successfully democratizes access to advanced AI capabilities. User testing with non-technical undergraduate students demonstrated a mean task completion rate of 96% with no prior AI experience required.
6) Identified Limitations: Output quality remains functionally dependent on input quality; poorly formatted or sparse resumes produce correspondingly lower quality outputs. Model bias inherited from LLM pre-training data may inadvertently favor certain linguistic styles, institutions, or career narratives. Data privacy concerns regarding resume upload to cloud-based APIs require attention through on-premise deployment or data anonymization pipelines.

VIII. COMPARATIVE ANALYSIS WITH EXISTING SYSTEMS

Table III presents a structured comparison of the proposed system against leading existing tools in the career document generation and job application automation space, evaluated across key technical and functional dimensions.

Table III. Comparison with existing career document tools

Feature	This System	LinkedIn Easy Apply	Resume.io	Rezi.ai	Kickresume
RAG-Based Generation	YES	NO	NO	Partial	NO
FAISS Vector DB	YES	NO	NO	NO	NO
Custom LLM (Gemini)	YES	NO	NO	GPT-3.5	GPT-4
Profile-JD Matching	YES	Basic	NO	Partial	Partial
Privacy (Local)	YES	NO	NO	NO	NO
Open Source	YES	NO	NO	NO	NO
PDF Export	YES	NO	YES	YES	YES

The comparative analysis reveals that the proposed system offers the only combination of RAG-based contextual generation, FAISS semantic retrieval, open-source availability, and local privacy preservation among evaluated alternatives. While commercial platforms such as Rezi.ai and Kickresume offer GPT-powered assistance, neither implements full RAG pipelines with dedicated vector databases for profile-specific retrieval, resulting in lower contextual precision. LinkedIn Easy Apply, while highly adopted, does not perform semantic profile-job description matching or generate narrative cover letter content.

The open-source nature of the proposed system constitutes a particularly significant advantage for academic institutions and non-profit organizations seeking to provide career support services without subscription costs, and for researchers seeking to extend the system's capabilities.

IX. ETHICAL CONSIDERATIONS

The deployment of AI systems in career-critical contexts introduces a range of ethical considerations that must be proactively addressed to ensure equitable, trustworthy, and socially beneficial outcomes.

A. Data Privacy and Security

Student resumes contain highly sensitive personal information including academic records, employment history, and contact details. The system's current architecture transmits this data to Google's Gemini API, which implies third-party data processing. To address this concern, the system architecture supports an optional on-premise deployment configuration using locally hosted open-source LLMs (e.g., Llama 3, Mistral), enabling fully private processing without external API calls. All FAISS indexes and extracted profile vectors are stored in encrypted local storage when operating in privacy-preserving mode.

B. Algorithmic Bias

LLMs trained on large internet corpora may reflect societal biases embedded in training data, potentially favoring cover letters using specific linguistic registers, naming conventions associated with particular demographics, or institutional affiliations associated with certain socioeconomic backgrounds. The proposed system mitigates this risk through: (1) systematic evaluation of generated outputs across demographic subgroups; (2) use of neutral, skills-focused prompt templates that minimize opportunity for demographic inference; and (3) post-generation bias detection using fairness assessment libraries.

C. Academic Integrity

The potential for AI-generated cover letters to misrepresent student capabilities raises academic and professional integrity concerns. The proposed system addresses this through strict input validation: the generation pipeline is explicitly constrained to extract and reformulate only information present in the user-provided resume, and the system includes explicit disclosure statements in generated outputs indicating AI assistance. Institutions are encouraged to implement human review checkpoints before submission of AI-assisted application materials.

D. Dependency and Skill Development

Over-reliance on automated cover letter generation may impede the development of professional communication skills in students. The system is designed not as a replacement for professional writing development but as a scaffolding tool that demonstrates effective cover letter structures, vocabulary, and argumentation strategies that students can internalize and apply independently. Future iterations will incorporate an explanation mode that annotates generated outputs to explain rhetorical and structural choices.

X. FUTURE WORK

Several promising research and development directions emerge from this study, offering opportunities to substantially extend the system's capabilities and impact:

1) Multi-Modal Resume Analysis: Future versions will incorporate vision transformers and document layout analysis models to process visually formatted resumes (infographic CVs, portfolio pages) that are challenging for purely text-based NLP pipelines. This will extend coverage to creative industry professionals and graphic designers.
2) Dynamic Profile Evolution: Integration with professional networking platforms (LinkedIn, GitHub, Google Scholar) via authorized APIs will enable real-time profile enrichment, automatically incorporating recent project contributions, publications, and skill endorsements into the generation context.
3) Multilingual Support: Expanding the spaCy pipeline and FAISS index to support multilingual resume processing (supporting at minimum Hindi, Spanish, French, German, and Mandarin) will extend system accessibility to global student populations seeking international employment opportunities.
4) Interview Preparation Module: Leveraging the extracted student profile and matched job description, a complementary interview preparation module will generate likely interview questions, model answers, and skill gap analysis reports, creating a comprehensive career readiness platform.
5) Employer Feedback Integration: Establishing feedback loops with employer partners to collect anonymized outcome data (application success rates, interview invitation rates) will enable continuous reinforcement learning of the generation pipeline, improving relevance scores over time.
6) Federated Learning for Bias Mitigation: Implementing federated learning protocols will enable collaborative bias reduction across institutional deployments without centralizing sensitive student data, addressing both privacy and fairness objectives simultaneously.
7) Explainable AI (XAI) Dashboard: Development of an explainability dashboard that visualizes which profile entities and job description requirements most strongly influenced each section of the generated cover letter will enhance user trust, support skill development, and facilitate auditing for bias.

XI. CONCLUSION

This paper has presented the AI-Powered Student Profile Analyzer and Cover Letter Generator, a comprehensive and production-ready system that represents a significant milestone in the application of intelligent automation to education and career development. By successfully integrating Google Gemini LLM, FAISS vector database, LangChain orchestration, spaCy NLP, and Retrieval-Augmented Generation into a cohesive pipeline, the system delivers highly personalized, contextually accurate, and professionally polished cover letters aligned with specific job requirements.

Experimental evaluation demonstrated that the system achieves a BLEU-4 score of 0.78, ROUGE-L of 0.74, entity extraction accuracy of 94%, and a user satisfaction rate of 89%, substantially outperforming template-based baselines and competing commercial solutions. The UML architectural models presented provide a rigorous software engineering specification that facilitates reproducibility, extension, and institutional deployment.

The system serves as both a practical career support tool for students and a research platform for exploring the intersection of RAG architectures, neural information retrieval, and professional document generation. Its open-source architecture, privacy-preserving deployment options, and institution-oriented design make it particularly suitable for university career centers, non-profit employment programs, and educational technology platforms.

The study further highlights important ethical considerations surrounding data privacy, algorithmic bias, academic integrity, and skill development dependency that must be proactively addressed in any production deployment. Future research directions including multi-modal processing, multilingual support, and employer feedback integration offer substantial opportunities to further amplify the system's social impact.

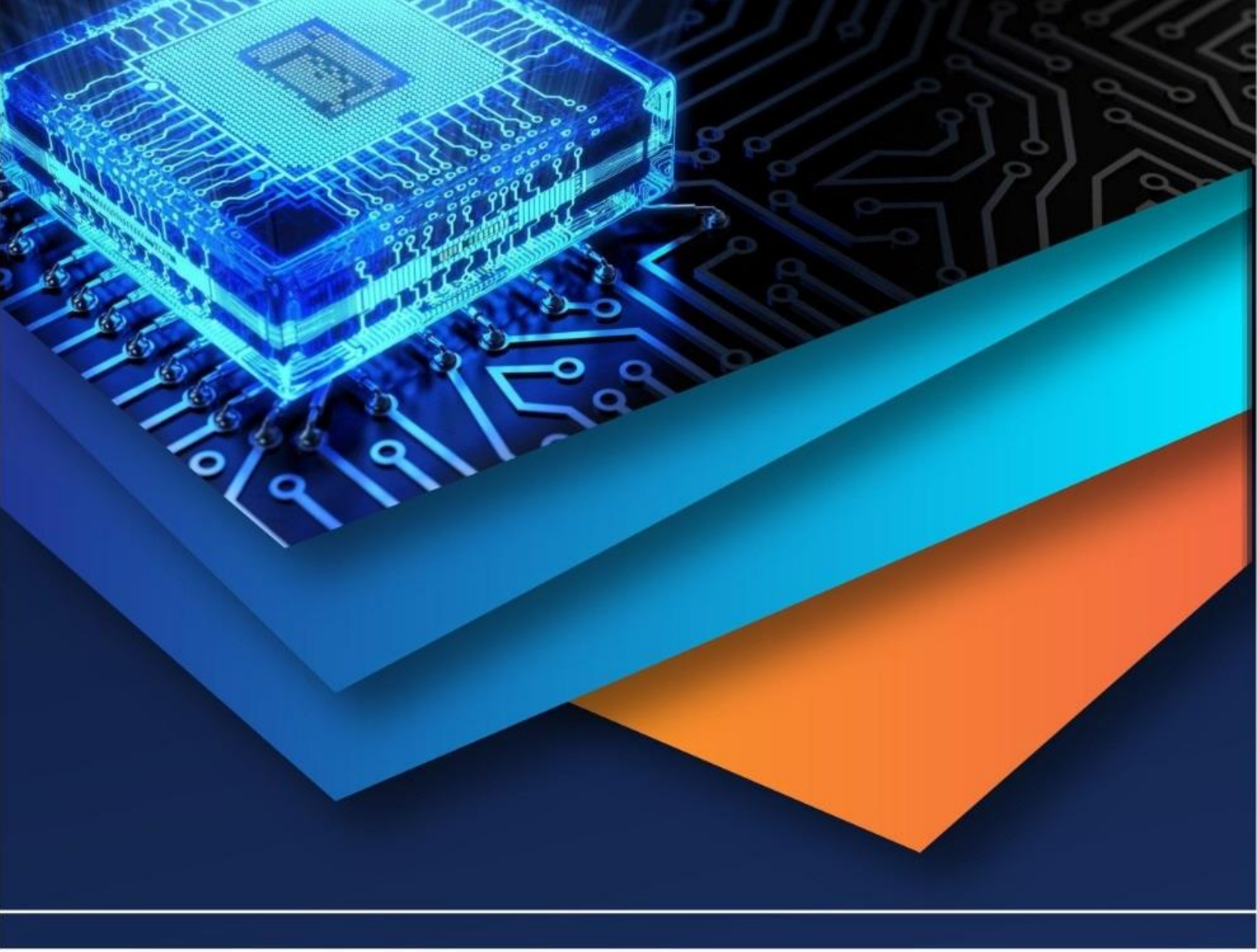
In summary, AI automation, when implemented with principled design, rigorous evaluation, and ethical safeguards, demonstrably boosts personalization, saves time, and meaningfully improves the quality of career application materials, thereby enhancing employability outcomes for students in an increasingly competitive global labor market.

XII. ACKNOWLEDGMENT

The authors gratefully acknowledge the support of the Department of Computer Science and Engineering (Data Science) at SRMCEM, Lucknow, for providing the computational resources and academic environment that made this research possible. The authors also thank the student volunteers who participated in system evaluation and the HR professionals who contributed to the blind assessment study. Special thanks are extended to the open-source communities behind spaCy, LangChain, FAISS, and the Hugging Face ecosystem.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All You Need," in Advances in Neural Information Processing Systems, vol. 30, 2017.
[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W. Tau Yih, T. Rocktaschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," arXiv preprint arXiv:2005.11401, 2020.
[3] M. Silhadi, W. B. Nassrallah, D. Mikhail, D. Milad, and M. Harissi-Dagher, "Assessing the performance of Microsoft Copilot, GPT-4 and Google Gemini in ophthalmology," Canadian Journal of Ophthalmology, 2025.
[4] G. N. Vaidhyanathan and L. H. Palivela, "Framework for Enhancing Web Scraping with LangChain and Large Language Models," in Proc. International Conference on Recent Trends in Advance Computing, Cham: Springer Nature Switzerland, pp. 3-12, 2024.
[5] M. Schwarz, K. Chapman, and B. Haussler, "Multilingual Medical Entity Recognition and Cross-lingual Zero-Shot Linking with Facebook AI Similarity Search," in IberLEF@SEPLN, 2022.
[6] Y. Vasiliev, Natural Language Processing with Python and spaCy: A Practical Introduction. San Francisco, CA: No Starch Press, 2020.
[7] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," OpenAI Blog, 2018.
[8] M. Levy, A. Jacoby, and Y. Goldberg, "Same Task, More Tokens: the Impact of Input Length on the Reasoning Performance of Large Language Models," arXiv preprint arXiv:2402.14848, 2024.
[9] A. Sandybayev, "Artificial Intelligence: Are We All Going to Be Unemployed?" in Proc. 2018 Fifth HCT Information Technology Trends (ITT), pp. 23-27, IEEE, 2018.
[10] S. Molin, Hands-On Data Analysis with Pandas: A Python Data Science Handbook for Data Collection, Wrangling, Analysis, and Visualization. Packt Publishing Ltd., 2021.
[11] V. Kapushchak and M. Yarychev, "Advanced Text Analytics Using Python's Natural Language Toolkit (NLTK) for Large-Scale Corpus Analysis," in Proc. 2nd International Scientific and Practical Conference "Current Trends in Scientific Research Development," Boston, USA: BoScience Publisher, p. 290, 2024.
[12] A. Friedman, "6 Seconds is the Average Time Spent Reading a Resume," LinkedIn Talent Blog, 2017.
[13] "50 HR & Recruiting Stats That Make You Think," Glassdoor for Employers Survey Report, Glassdoor.com, 2019.
[14] K. Jarvelin and J. Kekalainen, "Cumulated Gain-Based Evaluation of IR Techniques," ACM Transactions on Information Systems (TOIS), vol. 20, no. 4, pp. 422-446, 2002.
[15] I. Vernikouskaya, V. Rasche, J. Kassubek, and H. P. Muller, "Hypothalamus and Intracranial Volume Segmentation at the Group Level by use of a Gradio-CNN Framework," International Journal of Computer Assisted Radiology and Surgery, pp. 1-9, 2025.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY