# AI-Driven Email and Report Drafting System Using Generative Artificial Intelligence (Google Gemini)

G. Satish[1], Settibathula Jyothi[2], Motupalli Vishnu[3], Maddimsetti Rishika[4], Pilli Suma Sri[5], Dr.Y. Venkat[6],
[1]*Assistant Professor, Department of Artificial Intelligence and Machine Learning, Srinivasa Institute of Engineering and Technology*
[2,3,4,5]*UG Scholars, Department of Artificial Intelligence and Machine Learning, Srinivasa Institute of Engineering and Technology*
[6]*Professor, Department of Artificial Intelligence and Machine Learning, Srinivasa Institute of Engineering and Technology*

***Abstract*—Email is a valuable tool, but sometimes it creates some challenges like time consumption for writers. Miscommunication occurs frequent when people have different expectations about the messages that they send and what they receive. Students and organizational people frequently spend much time writing and revising reports and emails. Proficient communication through emails and reports is tiring and time-consuming in both corporate and academic situations. Preserving reliability in structure, clarity and tone often needs significant manual effort, especially under time restrictions. Recent developments in Generative Artificial Intelligence (GenAI) especially Large Language Models (LLMs), have empowered automatic generation of human-like text based on prompt or context. This made a way to discover an AI-Driven Email and Report Drafting System that uses Google Gemini Large Language Model to create or generate context-aware, high-quality professional content. The system is executed using an enterprise-grade N-Tier architecture by using React.js with a TypeScript frontend, a RESTful backend with Flask, and PostgreSQL relational database. Secure access is given with JWT (JSON Web Token) based authentication and Role-Based Access Control (RBAC) to separate admin and standard user privileges. Instruction based prompt engineering techniques helps in controlling the tone, structure, and quality of the generated content. The shaped content is stored for review and future reference. This demonstrates how Generative AI can be integrated into secure, scalable software systems, while maintaining consistency for academic and real-world use.**

***Index Terms*—Generative Artificial Intelligence, Google Gemini, Prompt Engineering, JWT Authentication, N-Tier Architecture, Flask, React, PostgreSQL**

## I. INTRODUCTION

Communication is very important in professional settings especially written communication, proficient communication has a serious role in administrative workflows, that includes internal and external management, communication, reporting, and documentation. In spite of digital tools availability, conscripting good emails and reports remains mostly manual process which need time, attention, and writing skills. But the growth of Generative Artificial Intelligence (GenAI) and Large Language Models (LLMs) has introduced new techniques for automating content. Models for example Google Gemini demonstrates strong capabilities in instruction-following, summarization, and structured content generation. But many existing AI tools work as standalone applications without proper authentication, data perseverance, or innovativeness architectural design. This project aims to address this by integrating Generative AI into a secure, enterprise-style web application.

## II. PROBLEM STATEMENT

Irrespective of changes in emerging digital communication tools, several challenges continue in qualified content drafting, the challenges need to be addressed are, Time Overwhelming, Manual writing of emails and reports takes valuable time that could be used for other important tasks. Consistency, maintaining a constant tone and structure across the documents is sometimes difficult and challenging, especially in limited time constraints. Safety Gaps,

Many AI-based tools have deficiency in proper authentication, authorization, and data perseverance mechanisms. Architectural Limitations, Small-scale AI tools every so often ignore innovativeness design principles, restricting scalability and maintainability. Quality, Output quality of the generating content depends on individual writing skills and time convenience.

This project reports the challenges by developing a secured, accessible, and AI-driven drafting system built by means of enterprise architectural patterns.

## III. LITERATURE REVIEW

Generative Artificial Intelligence, a kind of artificial intelligence that is used to create new content like text, images, audio, video, and code, mainly learning from the patterns from existing data. Unlike traditional AI which make analyzes or classifies data, Generative AI uses machine learning models to generate novel or new, human like outputs. Recent research and developments high spot the growing acceptance of Large Language Models for automatic text generation or creation [6]. Revisions on Generative AI reveal that instruction-based prompting meaningfully improves output relevance and structure [1]. Prompt engineering is developed as a key principle for controlling LLM performance. Research on Security highlights the importance of stateless authentication methods using like as JSON Web Tokens (JWT) for modern web applications [5]. JWT-based systems provide scalability, security and reduced server-side state management when compared to traditional session-based authentication. Enterprise architecture literature advocates N-Tier designs for separating concerns between presentation, application logic, and data storage layers. Such separation improves maintainability, testability, and system evolution. Google Gemini represents a new generation of LLMs augmented for instruction following and long-form content generation [1], making it appropriate for professional communication use cases.

## IV. SYSTEM ARCHITECTURE

The system architecture contains a four-layer N-Tier architecture

- Presentation Layer: React.js with TypeScript frontend
- Application Layer: Flask-based REST API
- Data Layer: PostgreSQL relational database
- AI Service Layer: Google Gemini API

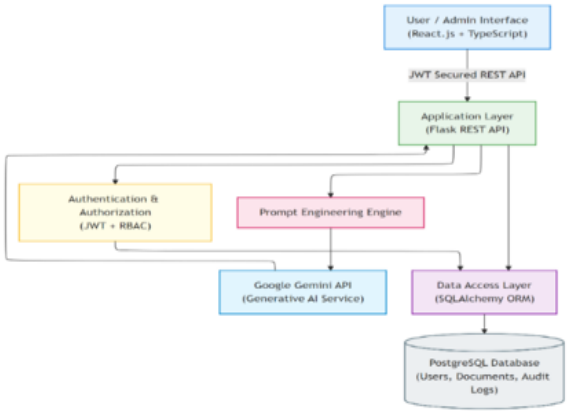



Figure 1. High-level architecture of the GenAI Email and Report Drafting System, showing communication between the presentation layer, application layer, AI service layer, and data layer.

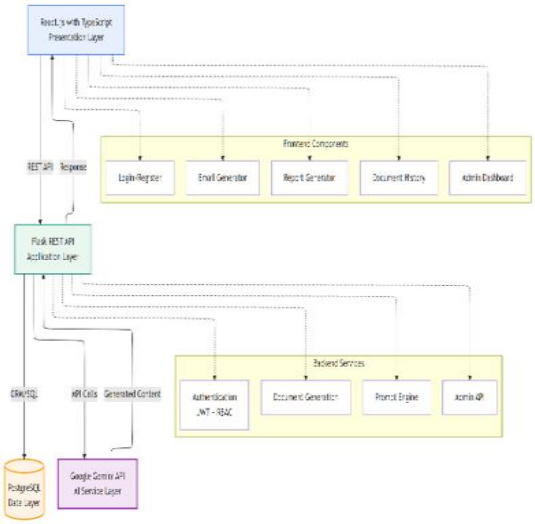
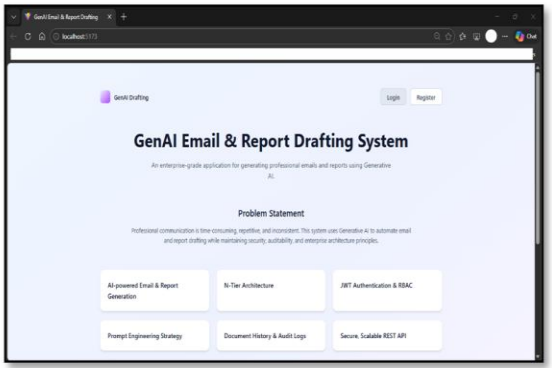


Figure 2. Detailed system architecture of GenAI Email and Report Drafting System portraying frontend components, backend services, AI integration, and data preservation.

## V. METHODOLOGY AND IMPLEMENTATION

The frontend is implemented using React.js, [3] an emerging technology with TypeScript to ensure the type security and modularity. Users interact with interface to login, generate email, generation of report, to view document history, and administrative dashboards.




The backend is implemented using python Flask [2] with latest updated version and disclosures RESTful APIs protected with JWT authentication. The logic, authorization checks, logins and AI interactions are handled completely in the backend.

The Artificial Intelligence service layer assimilates Google Gemini using a dedicated service module. Instruction based prompts defines the role, task, tone, and output limitations. This approach safeguards consistent and professional output.

PostgreSQL [4] is used to store user accounts such as login status, login details, generated documents details, and audit logs. The database schema contains three main tables, users (for authentication and role-based management), documents (for storing of generated content), and audit_logs (to store system activity and tracking). Referential veracity is maintained using foreign keys, and timestamps are used for historical tracking.

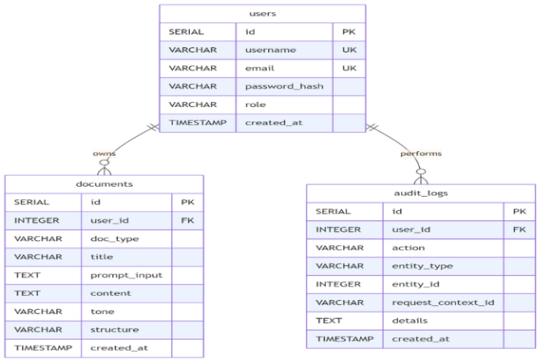



Figure 4. Database Entity-Relationship Diagram representing the schema structure with users, documents, and audit logs tables along with their relationships and constraints.

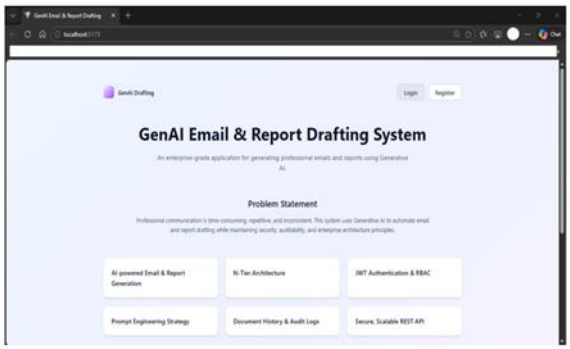
The system contains JWT based authentication, confirming scalability and secure API access. Passwords are stored by means of using secured hashing methods. Role-Based Access Control (RBAC) imposes authorizations for User and Admin roles. The authentication and authorization workflow are illustrated in Figure 3.

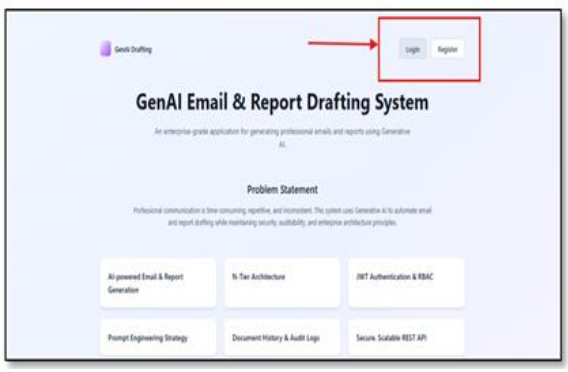
- Unit tests for backend services
- Integration tests for authentication and document generation
- Mocked AI API calls to avoid external dependencies
- CI pipelines for automated validation

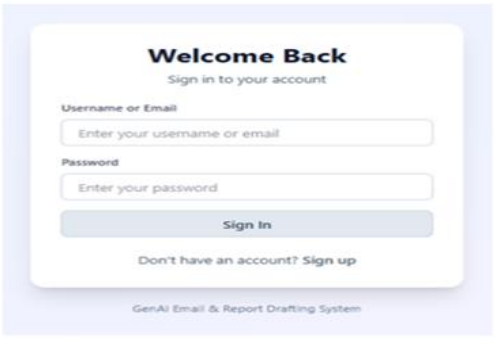
Over 127+ unit and integration tests validate core workflows and security-critical paths.

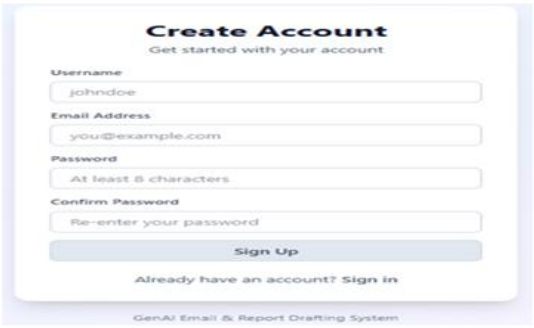
## VI. RESULTS AND EVALUATION

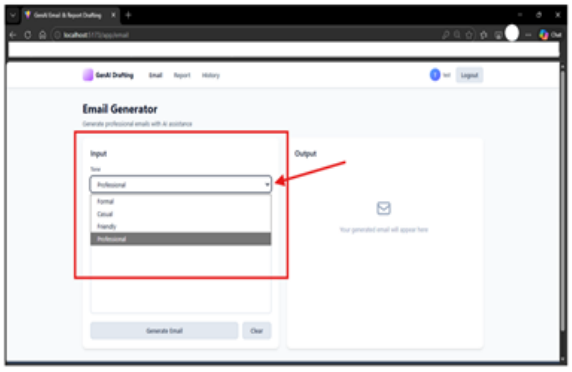
The system effectively produces professional emails and reports within seconds. Generation of Email can be done under 3 seconds per request under normal load, with report generation completing within 5 seconds. Valuation metrics include content relevance, tone accuracy, time efficiency, and system reliability. Compared to manual drafting of emails and reports, the system significantly reduces effort while improving consistency by producing emails and reports based on prompt given accurately.

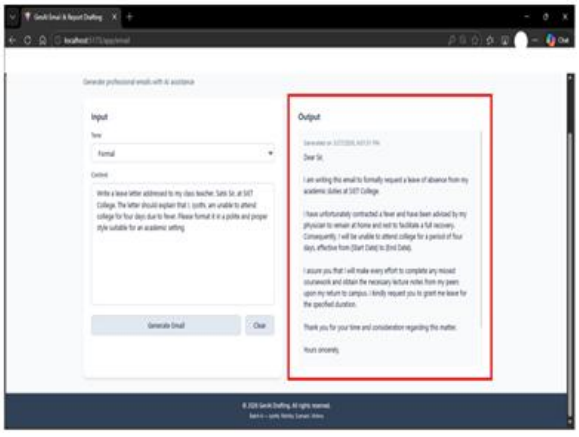


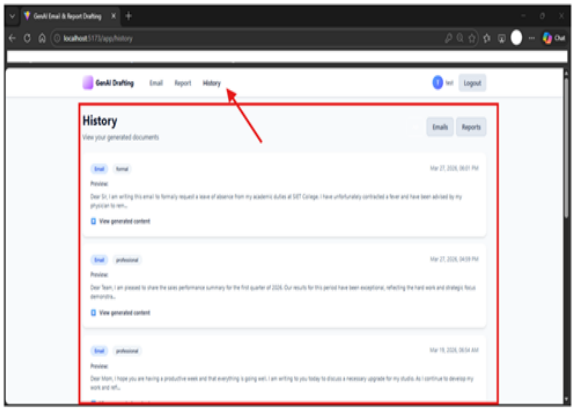






Future Enhancements

- Integration with enterprise IAM solutions
- Analytics and reporting dashboards
- PDF and email export functionality
- Retrieval-Augmented Generation (RAG)
- Multilingual content generation

## VII. CONCLUSION

This project determines and demonstrates the successful integration of Generative Artificial Intelligence into a secures and enterprise model graded web application. With the combination of Google Gemini with N-Tier architecture, JWT authentication, and prompt engineering, this entire system provides a scalable and academically robust solution for automated professional content generation.

## REFERENCES

[1] Google, “Gemini API documentation,” Google AI, 2024. [Online]. Available: https://ai.google.dev/gemini-api

[2] Pallets Projects, “Flask web framework documentation,” 2024. [Online]. Available: https://flask.palletsprojects.com

[3] Meta, “React.js official documentation,” 2024. [Online]. Available: https://react.dev

[4] PostgreSQL Global Development Group, “PostgreSQL documentation,” 2024. [Online]. Available: https://www.postgresql.org/docs

[5] M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” IETF RFC 7519, May 2015. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7519

[6] Y. Zhang et al., “A survey of large language models for text generation,” ACM Comput. Surv., vol. 55, no. 12, pp. 1–38, 2023.
[7] T. Brown et al., “Language models are few-shot learners,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020.
[8] OpenAI, “GPT-4 technical report,” arXiv preprint arXiv:2303.08774, 2023.
[9] H. Touvron et al., “LLaMA: Open and efficient foundation language models,” arXiv preprint arXiv:2302.13971, 2023.
[10] S. Bubeck et al., “Sparks of artificial general intelligence: Early experiments with GPT-4,” arXiv preprint arXiv:2303.12712, 2023.
[11] M. Grinberg, Flask Web Development: Developing Web Applications with Python. Sebastopol, CA, USA: O’Reilly Media, 2018.
[12] M. Abadi et al., “TensorFlow: Large-scale machine learning on heterogeneous systems,” Google Research, 2016. [Online]. Available: https://www.tensorflow.org
[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2019.
[14] Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017.
[15] D. Li et al., “Survey of web application security using JWT and OAuth2,” IEEE Access, vol. 10, pp. 12345–12360, 2022.