International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Smart Academic and Placement Assistant Using Graph-Rag

Dr. Nandish M[1], Sanjana S.A[1], Shreya K.U[1], Spoorthi D Shet[1], Syed Faizan[1]

[1]Department of Computer Science and Engineering, Jawaharlal Nehru New College of Engineering, Shimoga, India

Abstract — The increasing volume of academic and placement-related information creates challenges for students in efficiently finding relevant and contextual resources. This research addresses this gap by proposing a Smart Academic and Placement Assistant using Graph-RAG, which combines graph-based knowledge representation with Retrieval-Augmented Generation for intelligent information retrieval. The proposed system provides a web-based platform for accessing academic notes, question papers, placement resources, and AI-based assistance. The system uses a Python and FastAPI-based backend with a React and TypeScript frontend, while relevant academic content is processed and retrieved to generate context-aware responses using a large language model. The approach improves the organization and accessibility of academic information while supporting students in learning and placement preparation. The proposed system demonstrates the potential of Graph-RAG-based retrieval for providing relevant and efficient academic assistance through a unified platform.

Keywords — Graph-RAG, Retrieval-Augmented Generation, Knowledge Graph, Large Language Model, Academic Assistant, Placement Assistance, Intelligent Information Retrieval.

I. INTRODUCTION

The increasing use of digital platforms in education has made academic and placement-related information easily available to students. However, the information is often distributed across different sources such as academic notes, question papers, learning materials, skill resources, and placement preparation materials. Finding the required information from these sources can be time-consuming and inefficient. Conventional search methods mainly depend on keyword matching and may not provide sufficient contextual understanding of student queries. This creates a need for an intelligent and centralized system that can assist students in accessing relevant academic and placement information efficiently.

This project focuses on developing a Smart Academic and Placement Assistant using Graph-RAG that integrates intelligent information retrieval with a user-friendly web-based platform. At its core, the proposed system combines Retrieval-Augmented Generation (RAG) with graph-based knowledge representation to improve contextual retrieval and response generation. The system provides students with a single platform for accessing academic notes, previous question papers, placement resources, and AI-based assistance. Academic resources can be managed through an administrative module, where authorized administrators can upload PDF-based learning materials. The system organizes the available information so that relevant content can be retrieved when students submit queries. This centralized approach improves accessibility and reduces the effort required to search for academic and placement-related resources from multiple sources.

The system is implemented using React and TypeScript for the frontend and Python with FastAPI for backend processing. Uploaded PDF documents are processed using PyPDF, which extracts the textual content and stores the processed information in a JSON-based knowledge base. When a student enters a query, the request is sent from the frontend to the FastAPI backend through REST APIs. The relevant academic information is retrieved from the available knowledge base and provided as context to the Google Gemini API for generating an appropriate response. The generated response is then displayed through the web interface, along with the relevant source information where applicable.

The proposed system is significant because it brings academic assistance and placement preparation together within a single platform. By reducing information searching time and providing context-aware AI assistance, the system can help students access learning resources more effectively and prepare for examinations and placement activities. The administrative module also provides a convenient way to manage academic resources and keep the available content organized. Overall, the project demonstrates the application of AI, RAG, graph-based knowledge representation, and web technologies in developing an intelligent academic support system. It provides a foundation for future enhancements such as personalized recommendations, improved graph-based reasoning, and expanded placement assistance.

II. LITERATURE SURVEY

Retrieval-Augmented Generation and graph-based information retrieval have become active areas of research, with significant contributions focusing on contextual retrieval, knowledge representation,

graph-based reasoning, and intelligent question answering. This survey reviews the key literature in these areas, summarizing the methodologies, findings, and limitations relevant to the proposed Smart Academic and Placement Assistant using Graph-RAG.

Lewis et al. [1] proposed Retrieval-Augmented Generation (RAG), which combines information retrieval with language models to improve factual accuracy and reduce hallucination. However, traditional RAG mainly focuses on unstructured text and has limitations in understanding complex relationships and multi-hop reasoning.

H. Han et al. [2] introduced Graph-RAG by integrating knowledge graphs with language models, enabling relational understanding, context-aware retrieval, and multi-hop reasoning. The approach improves retrieval quality but involves challenges related to graph construction, computational requirements, and scalability.

Wang et al. [3] discussed Knowledge Graph-based Question Answering systems, where entities and relationships are represented using graph structures to support structured information retrieval and better explainability. Their work highlighted the usefulness of knowledge graphs for domain-specific applications, although dependency on predefined schemas and graph construction remains a limitation.

T. Kipf and M. Welling [4] presented Graph Neural Networks (GNNs), which learn representations by aggregating information from neighboring nodes. GNNs are useful for graph representation learning, recommendation, and retrieval, but computational complexity and scalability remain challenges.

A. Grover and J. Leskovec [5] proposed Node2Vec, which uses random walks to learn node representations and structural relationships for similarity-based retrieval and recommendation.

B. Perozzi et al. [6] introduced DeepWalk, another graph embedding approach that learns node relationships through random walks and supports clustering, classification, and retrieval tasks. However, both approaches mainly focus on graph structure and have limited contextual and semantic understanding.

Y. Yasunaga et al. [7] discussed Neural-Symbolic Retrieval, combining neural language understanding with symbolic reasoning to support context-aware and multi-hop retrieval.

X. Wu et al. [8] proposed graph-based retrieval and pruning techniques to reduce irrelevant graph components and improve retrieval efficiency, while highlighting the risk of losing useful information during pruning.

The reviewed literature shows the evolution of intelligent retrieval from traditional RAG to graph-based and hybrid retrieval approaches. These studies demonstrate improvements in contextual understanding, relationship mapping, and intelligent retrieval while identifying challenges related to scalability, computational complexity, graph construction, and semantic understanding. These findings provide the foundation for developing the proposed Smart Academic and Placement Assistant using Graph-RAG for efficient academic and placement-related information retrieval.

III. PROPOSED METHODOLOGY

The stages of the proposed methodology are as follows:

A. Input

The Input block serves as the entry point of the proposed Smart Academic and Placement Assistant, allowing students and administrators to interact with the system through a web-based interface. Students can enter academic and placement-related queries through the chatbot and can access available academic notes, question papers, and placement resources. The input interface is designed to provide simple navigation and convenient access to the major functions of the system. Administrators can access the admin module through authentication and upload academic PDF resources for further processing. These input mechanisms provide a centralized and user-friendly environment for both students and administrators.

B. Academic Resource Upload

The Academic Resource Upload block is responsible for managing academic resources provided by the administrator. After successful authentication, the administrator can upload academic PDF documents through the admin dashboard. The system validates the uploaded files to ensure that the supported PDF format is used and stores the documents in the local notes directory. The uploaded file information is also maintained for further processing and retrieval. This module helps administrators maintain and organize academic resources in a centralized location. The availability of managed academic resources allows students to access relevant learning materials through the system without depending on multiple external sources.

C. PDF Text Extraction

The PDF Text Extraction block converts the uploaded academic documents into textual information that can be used by the AI assistant. The system uses PyPDF to read the uploaded PDF documents and extract text from their individual pages. The extracted content is combined and stored along with the corresponding file name and file path in a JSON-based knowledge base. This process makes the information contained in academic documents

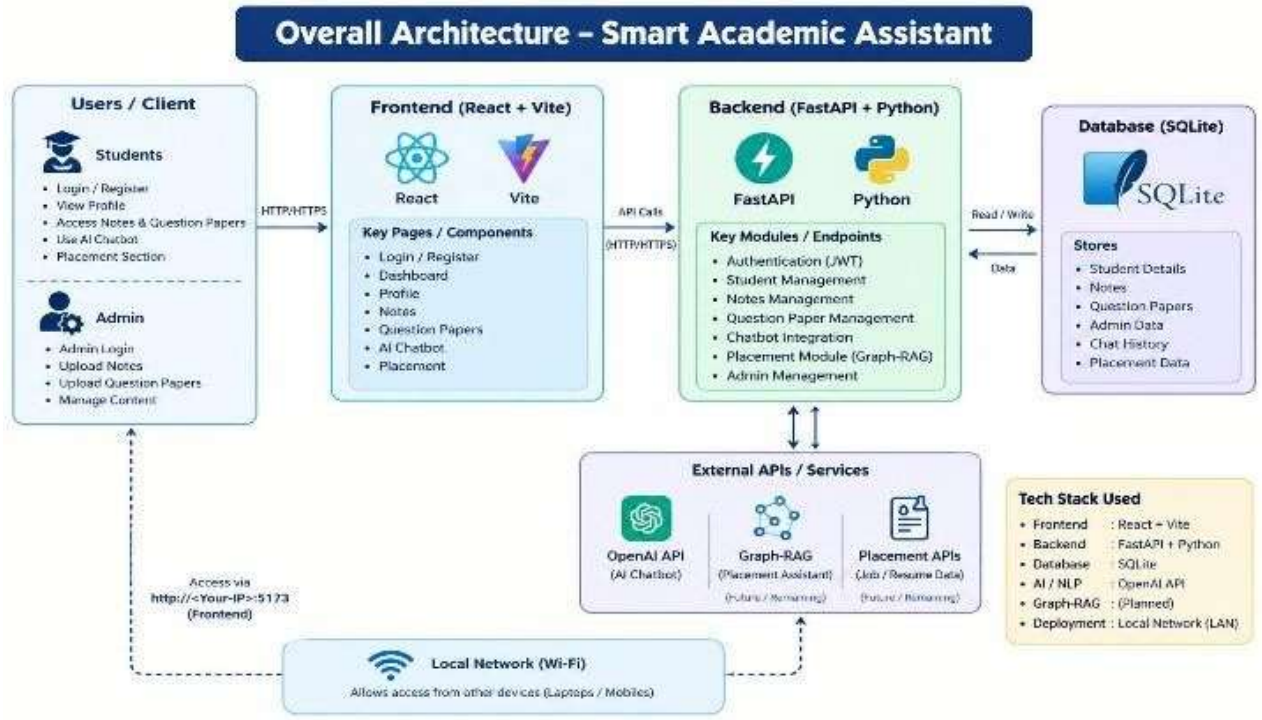


Fig. 1 Overall System Architecture of the Smart Academic and Placement Assistant

available for retrieval during query processing. The extracted text also preserves the connection between the retrieved information and its original academic PDF source. Thus, PDF processing acts as an important stage between resource upload and intelligent response generation.

D. Query Processing and Retrieval

The Query Processing and Retrieval block handles questions submitted by students through the chatbot interface. The frontend, developed using React and TypeScript, sends the user query to the FastAPI backend through REST API communication. The backend receives and processes the query and searches the available academic knowledge base for relevant information. The retrieved content is then prepared as contextual information for the response generation stage. This retrieval process helps the system provide responses based on the available academic resources rather than depending only on general knowledge. The approach is intended to improve the relevance and usefulness of responses for academic queries.

E. AI Response Generation

The AI Response Generation block is responsible for producing the final response to the student's query. The relevant information retrieved from the academic knowledge base is provided as context to the Google Gemini API. The language model processes the query together with the retrieved information and generates a meaningful response. By using the retrieved academic context, the system can provide more relevant and context-aware assistance to students. The generated response can address questions related to academic concepts and available learning resources, supporting students in their learning and preparation activities.

F. Response and Resource Access

The Response and Resource Access block represents the final stage of the proposed methodology. The generated response from the backend is returned to the React-based frontend and displayed to the student through the chatbot interface. Where applicable, the system also provides the relevant academic source information associated with the retrieved content. Students can additionally access available notes, question papers, and placement-related resources through the platform. The complete workflow connects user input, resource processing, information retrieval, AI response generation, and resource access into a single system. This integrated workflow provides students with a convenient platform for academic learning and placement preparation.

G. Placement Assistance

The Placement Assistance block supports students in preparing for placement activities through the available placement-related resources. Students can access interview questions, aptitude materials, skill-related content, and other placement preparation resources through the system. The retrieved information can be used to provide relevant guidance based on the student's query. This module extends the academic assistance functionality by bringing placement preparation into the same platform, helping students manage both academic learning and career preparation in a unified environment.

IV. RESULTS AND SNAPSHOTS

The Smart Academic & Placement Assistant was developed as a centralized academic resource and placement preparation platform integrating a React-based user interface, a TypeScript/Node.js backend, and a generative AI chatbot. The application provides role-based access for students and administrators, allowing administrators to publish academic PDF resources and students to access learning materials and ask questions through the AI Academic Assistant.



Fig. 2 Dashboard Interface of the Smart Academic and Placement Assistant

The chatbot uses the Google Gemini API through the Google Gen AI SDK, with the configured model gemini-3.8-flash. The backend retrieves relevant passages from indexed PDF pages and supplies the retrieved context alongside the user's query to the language model. The answer-generation pipeline is designed to produce a direct explanation, supporting details, and examples where relevant. Retrieved source references are displayed with the PDF filename, page number, and text excerpt, enabling students to verify the information against the uploaded academic notes.

The PDF retrieval component uses the pdf-parse library for page-wise text extraction and sql.js for SQLite-based storage of indexed page content. The chatbot response is returned through the Express REST endpoint POST /api/chat. The frontend then presents the generated explanation and its associated PDF references in the chatbot interface.

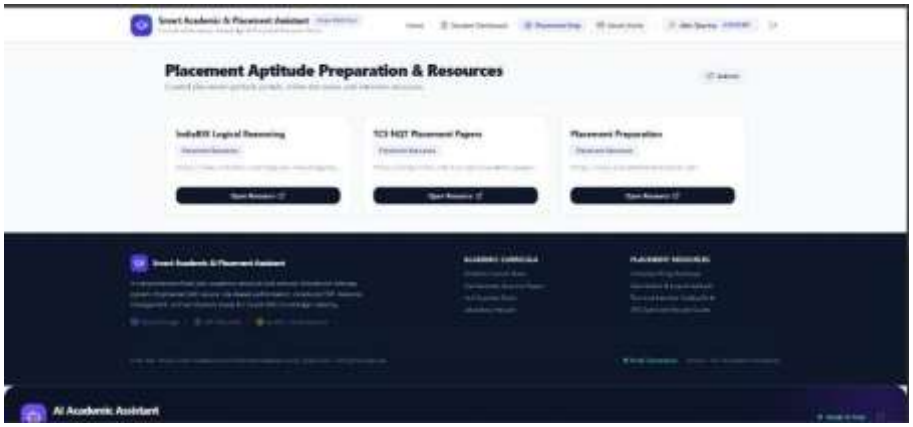


Fig. 3 Placement Aptitude Preparation and Resource Module

The system can be evaluated using answer-generation latency, retrieval relevance, answer completeness, and source-reference accuracy. Response-time measurements should be collected from repeated queries under the same operating conditions to determine the average time required by the configured model to

generate a response. These measurements provide a reproducible basis for assessing the responsiveness of the academic question-answering system.

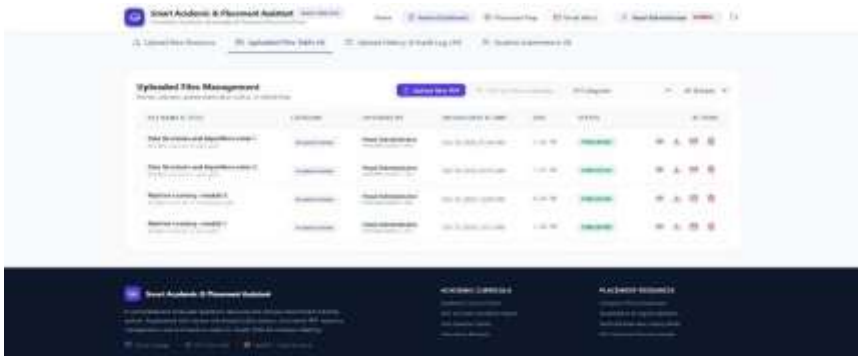


Fig. 4 Strict Role-Based Access Control (RBAC) Module for Administrators and Students

TABLE I : IMPLEMENTATION ENVIRONMENT AND CHATBOT CONFIGURATION

SL.NO	Component	Technology /Model	Version/Configuration
1	Programming Language	Python	3.12.10
2	Frontend Framework	React	18.3.1
3	Frontend Language	TypeScript	5.7.2
4	Frontend Build Tool	Vite	6.0.5
5	Backend Framework	Express.js	Verify installed version
6	AI Model	Gemini Flash	Gemini-3.8-flash
7	AI Backend API	Google Gemini API	@google/genai
8	PDF Text Extraction	PDF-Parse	2.4.5
9	PDF Indexing/Storage	SQLite using sql.js	Verify installed version
10	Chat Endpoint	Express REST API	POST/api/chat
11	PDF Retrieval	Indexed PDF-Page Search	Top 5 matching passages
12	Response Format	Generated Answer with PDF References	Answer text, file name, page number, excerpt

REFERENCES

[1] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[2] H. Han et al., “Retrieval-Augmented Generation with Graphs (Graph-RAG),” arXiv preprint arXiv:2501.00309, 2025.

[3] Y. Wang et al., “Knowledge Graph-based Question Answering Systems,” IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 5, pp. 2103–2115, 2022.

[4] T. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” International Conference on Learning Representations (ICLR), 2017.

[5] A. Grover and J. Leskovec, “Node2Vec: Scalable Feature Learning for Networks,” Proceedings of the 22nd ACM SIGKDD Conference, 2016.

[6] B. Perozzi et al., “DeepWalk: Online Learning of Social Representations,” Proceedings of the 20th ACM SIGKDD Conference, 2014.

[7] Y. Yasunaga et al., “QA-GNN: Reasoning with Language Models and Knowledge Graphs,” NAACL, 2021.

[8] X. Wu et al., “Graph-based Retrieval for Knowledge-Intensive Tasks,” IEEE Access, 2023.