iJRASET
International Journal For Research in Applied Science and Engineering Technology



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 14 **Issue:** III **Month of publication:** March 2026

**DOI:** https://doi.org/10.22214/ijraset.2026.79104

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 III Mar 2026- Available at www.ijraset.com*

# Engineering a Unified Digital Workspace: The NexusAI Architecture for Student Productivity and Placements

Shravani Sunil Dakve[1], Pooja Desai[2], Anchal Pandey[3], Mrs. Neha Kale[4]
*Department of Computer Engineering*

***Abstract:*** ***In today's rapidly evolving academic environment, engineering students rely on a variety of digital tools to handle their studies, stay connected with their peers, and prepare for future career prospects. These tools include learning management systems, coding practice websites, communication platforms, and job interview preparation resources. However, since these platforms operate separately, students often have to switch between them multiple times. This frequent switching can break their focus and lead to a disjointed, less efficient learning process. To tackle this challenge, we have created NexusAI, a single, web-based platform that brings together several essential student tools into one unified system. The main goal of NexusAI is to streamline the academic workflow and boost overall productivity by eliminating the need to use multiple disconnected applications.***

***The platform is built using modern web technologies such as React for the user interface, Node.js for the backend, and MongoDB for managing data effectively. Socket.io is also used to enable real-time communication, which is especially helpful for collaborative tasks. A key feature of NexusAI is its intelligent tutoring support, which is driven by the Google Gemini API. This feature functions as a smart assistant that offers context-sensitive guidance to help students grasp concepts, solve problems, and prepare for interviews more efficiently. The system also includes virtual study rooms where students can work together in real time, exchange ideas, and collaborate on projects. These rooms are designed to mimic the experience of a physical classroom or group study session in a digital format. Another major component of the platform is the Applicant Tracking System (ATS) simulator.***

***This tool enables students to assess their resumes against industry standards and understand how actual recruitment systems evaluate job applications. The ATS simulator uses specific algorithms to analyze resumes, detect important skills, and provide feedback, thus helping students enhance their chances of being selected for job opportunities.***

***This paper presents a detailed overview of the system architecture of NexusAI, including the design and implementation of its different modules. It also explains the methods used for real-time data synchronization and the underlying principles of the ATS simulation. The results from testing and user feedback show that integrating these various tools into one platform significantly reduces distractions, improves productivity, and offers a more structured way to approach learning and career readiness. Overall, NexusAI helps connect academic learning with professional preparation by providing a smooth and efficient user experience.***

***Index Terms: Educational Technology, Collaborative Learning, Student Productivity, Web Applications, Job Placement Systems, Real-Time Systems, AI-based Tutoring***

## I. INTRODUCTION

Currently, pursuing an engineering degree involves more than just attending classes and finishing assignments. Students are required to handle various tasks simultaneously, such as grasping theoretical concepts, practicing coding, working on projects, and preparing for job placements. Although numerous online resources are accessible, they are not well-integrated, which can cause confusion instead of simplifying the learning process.

*1) On a daily basis, students often switch between multiple platforms.*

For instance, study materials might be on one site, coding practice on another, and communication with peers through a different application. When students have questions, they frequently seek help from separate tools. Similarly, interview preparation also requires visiting different platforms. This constant shifting between tools results in a significant loss of time and effort, making it challenging to maintain focus on a single task for an extended period.

©IJRASET: All Rights are Reserved 

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

*2) A major challenge is the lack of an effective way to track progress.*

With activities spread across various platforms, students do not have a centralized place to assess their learning or gauge how well they are prepared for job opportunities. This creates a disconnect between academic knowledge and the practical skills needed for placements.

*3) To address these challenges, we developed a system called NexusAI.*

The primary goal of this project is to offer a unified platform where students can perform all their essential tasks without relying on multiple applications. With NexusAI, students can study, practice coding, interact with others, and prepare for interviews in one location. This integrated approach helps reduce distractions and enhances productivity.

*4) During the design of this platform, a strong emphasis was placed on real-time features, particularly for group study and collaborative work.*

It was crucial to ensure seamless communication and smooth collaboration without any delays. Additionally, an intelligent assistant feature was incorporated to offer basic guidance and assistance when needed.

*5) This paper is structured in a straightforward manner.*

Initially, we outline the limitations of current systems. Next, we explain the design and functionality of our platform. Following that, we discuss the major features developed, including collaboration tools and resources for placement preparation. Finally, we present the results and explore potential areas for future improvement.

## II. LITERATURE REVIEY

In today's fast-moving academic world, engineering students use a variety of digital tools to handle their coursework, stay in touch with classmates, and get ready for their future careers. These tools include learning management systems, coding practice websites, communication apps, and interview preparation platforms. However, because these platforms are separate from each other, students often need to move back and forth between them. This constant switching can break their focus and lead to a scattered, less effective learning experience.

To solve this, we created NexusAI, a single, web-based productivity platform that brings together several important student tools into one central place.

The main goal of NexusAI is to make the academic process easier and more efficient by cutting down on the need to use multiple separate apps. The platform was made using modern web technologies, such as React for the user interface, Node.js for handling the backend tasks, and MongoDB for storing and managing data efficiently. We also used Socket.io to add real-time communication features, which are especially helpful for teamwork and collaboration.

One of the standout features of NexusAI is its smart tutoring support, which is powered by the Google Gemini API.

This acts like a helpful assistant that gives students context-based advice, helping them grasp difficult ideas, work through problems, and become better prepared for interviews. The system also offers virtual study rooms where students can work together in real time, share thoughts, and collaborate on projects. These virtual rooms are designed to feel like real classrooms or group study spaces in a digital environment.

Another key part of the platform is the Applicant Tracking System (ATS) simulator.

This allows students to check their resumes against industry standards and see how real-world hiring systems assess job applications. The ATS simulator uses specific algorithms to review resumes, spot important skills, and give feedback, helping students boost their chances of being considered for job opportunities.

This paper gives a detailed look at the system structure of NexusAI, including how each part was designed and built.

It also explains the methods used for keeping data in sync in real time and how the ATS simulation works. The results from testing and user feedback show that combining these different tools into one platform greatly reduces distractions, boosts productivity, and provides a more organized way to approach both learning and career preparation. Overall, NexusAI helps connect academic study with professional readiness by offering a smooth and effective user experience.

## III. SYSTEM ARCHITECTURE

In today’s rapidly evolving academic environment, engineering students rely on a variety of digital tools to handle their studies, stay connected with their peers, and prepare for future career prospects.

©IJRASET: All Rights are Reserved 

These tools include learning management systems, coding practice websites, communication platforms, and job interview preparation resources. However, since these platforms operate separately, students often have to switch between them multiple times. This frequent switching can break their focus and lead to a disjointed, less efficient learning process.

To tackle this challenge, we have created NexusAI, a single, web-based platform that brings together several essential student tools into one unified system.

The main goal of NexusAI is to streamline the academic workflow and boost overall productivity by eliminating the need to use multiple disconnected applications. The platform is built using modern web technologies such as React for the user interface, Node.js for the backend, and MongoDB for managing data effectively. Socket.io is also used to enable real-time communication, which is especially helpful for collaborative tasks.

A key feature of NexusAI is its intelligent tutoring support, which is driven by the Google Gemini API.

This feature functions as a smart assistant that offers context-sensitive guidance to help students grasp concepts, solve problems, and prepare for interviews more efficiently. The system also includes virtual study rooms where students can work together in real time, exchange ideas, and collaborate on projects. These rooms are designed to mimic the experience of a physical classroom or group study session in a digital format.

Another major component of the platform is the Applicant Tracking System (ATS) simulator.

This tool enables students to assess their resumes against industry standards and understand how actual recruitment systems evaluate job applications. The ATS simulator uses specific algorithms to analyze resumes, detect important skills, and provide feedback, thus helping students enhance their chances of being selected for job opportunities.

This paper presents a detailed overview of the system architecture of NexusAI, including the design and implementation of its different modules.

It also explains the methods used for real-time data synchronization and the underlying principles of the ATS simulation. The results from testing and user feedback show that integrating these various tools into one platform significantly reduces distractions, improves productivity, and offers a more structured way to approach learning and career readiness. Overall, NexusAI helps connect academic learning with professional preparation by providing a smooth and efficient user experience.

Index Terms

Educational Technology, Collaborative Learning, Student Productivity, Web Applications, Job Placement Systems, Real-Time Systems, AI-based Tutoring



Fig. 1. High-level tier architecture depicting component routing between the React client, Node.js clusters, state engines, and the external API orchestration endpoints.

## IV. CORE MODULE IMPLEMENTATIONS CORE MODULE IMPLEMENTATIONS

Transforming these theories into functioning code required combining several tightly connected micro-services.

### *A. Smart Tutoring Service*

The core cognitive component of NexusAI depends on the Google Gemini API, which is accessed via our server wrappers.

To ensure the AI functions as a real educational assistant rather than a direct answer provider, all incoming questions go through a strict Socratic Control Layer. This layer uses specific system instructions to guide the API toward asking questions instead of delivering finished code, encouraging the student to find solutions on their own.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Moreover, this prompt engineering dynamically introduces historical data from the user's database.

For instance, if a student uploads a faulty Python script, the AI detects the error but only points out the conceptual syntax issue, pushing the student to solve the problem independently. If the user has a high XP ranking in Data Structures, the Socratic prompts grow more challenging, ensuring the tutoring aligns with the student's growing expertise.

To prevent common issues like model hallucination, which can occur with unrestricted large language models, NexusAI implements a strict Retrieval-Augmented Generation (RAG) process.

Student queries are first converted into embeddings and compared against the university syllabus dataset using vector cosine similarity. This ensures that the generated Socratic responses are based strictly on validated educational materials, avoiding the AI from creating content outside the curriculum.

*B. Collaborative Study Corridors*

To create the actual Study Rooms, we had to implement some complex operational logic to maintain consistent screen displays for all participants.

When a user clicks "Create Room," the server creates a unique cryptographic slug and sets up an isolated hash space for those connections.

When friends join, their connections are mapped directly into that specific room hash.

The real innovation occurs on the SVG whiteboard. Whether someone is using an iPad stylus or a basic mouse, our system converts their strokes into integer coordinates. These updates are limited to prevent overwhelming the server, and then they are shared with the group. This results in fast synchronization, so users can see what their friend, even across the country, is drawing with no delay.

Beyond just visual coordination, these corridors also support WebRTC-based low-latency audio channels.

Instead of sending heavy audio data through the central Node.js server, which could severely impact performance, clients establish direct peer-to-peer (P2P) UDP streams. The central server acts as a signaling coordinator, exchanging Session Description Protocol (SDP) configurations before stepping away from the audiovisual transmission.

Intelligent Resume Builder and Placement Hub

The app's final goal is to help students transition from studying to securing jobs.

Traditional resume builders can be very time-consuming, so we developed an automated version. Because the app tracks all user activities—study streaks, algorithm completions, and AI tutor scores—we set up the backend to automatically collect this data and format it into a recruiter-friendly PDF.

To better prepare students for automated recruitment screening, the module includes a customized Applicant Tracking System (ATS) simulator.

When a user inputs a job description, the system performs Term Frequency-Inverse Document Frequency (TF-IDF) analysis on both the resume and the job requirements. By comparing keyword vectors, the algorithm provides a match percentage, pointing out key missing skills before the student applies.

Fig. 2. Heuristic mapping workflow showing ATS string similarities comparing candidate input metrics against extracted corporate requirements.

Once the resume is ready, the hub uses Gemini to conduct very intense mock interviews.

We created specific system instructions to allow the AI to take on different roles—either as an annoying HR interviewer focusing on soft skills or as a fearsome tech lead challenging the user to solve timed architecture problems targeting their specific weaknesses.



Fig. 2. Heuristic mapping workflow detailing ATS string similarities comparing candidate input metrics against extracted corporate requirements.

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

## V. METHODOLOGY AND PROCEDURAL LOGIC CORE MODULE IMPLEMENTATIONS

Transforming these theories into functioning code required combining several tightly connected micro-services.

### *A. Smart Tutoring Service*

The core cognitive component of NexusAI depends on the Google Gemini API, which is accessed via our server wrappers.

To ensure the AI functions as a real educational assistant rather than a direct answer provider, all incoming questions go through a strict Socratic Control Layer. This layer uses specific system instructions to guide the API toward asking questions instead of delivering finished code, encouraging the student to find solutions on their own.

Moreover, this prompt engineering dynamically introduces historical data from the user's database.

For instance, if a student uploads a faulty Python script, the AI detects the error but only points out the conceptual syntax issue, pushing the student to solve the problem independently. If the user has a high XP ranking in Data Structures, the Socratic prompts grow more challenging, ensuring the tutoring aligns with the student's growing expertise.

To prevent common issues like model hallucination, which can occur with unrestricted large language models, NexusAI implements a strict Retrieval-Augmented Generation (RAG) process.

Student queries are first converted into embeddings and compared against the university syllabus dataset using vector cosine similarity. This ensures that the generated Socratic responses are based strictly on validated educational materials, avoiding the AI from creating content outside the curriculum.

### *B. Collaborative Study Corridors*

To create the actual Study Rooms, we had to implement some complex operational logic to maintain consistent screen displays for all participants.

When a user clicks "Create Room," the server creates a unique cryptographic slug and sets up an isolated hash space for those connections.

When friends join, their connections are mapped directly into that specific room hash.

The real innovation occurs on the SVG whiteboard. Whether someone is using an iPad stylus or a basic mouse, our system converts their strokes into integer coordinates. These updates are limited to prevent overwhelming the server, and then they are shared with the group. This results in fast synchronization, so users can see what their friend, even across the country, is drawing with no delay.

Beyond just visual coordination, these corridors also support WebRTC-based low-latency audio channels.

Instead of sending heavy audio data through the central Node.js server, which could severely impact performance, clients establish direct peer-to-peer (P2P) UDP streams. The central server acts as a signaling coordinator, exchanging Session Description Protocol (SDP) configurations before stepping away from the audiovisual transmission.

### *C. Intelligent Resume Builder and Placement Hub*

The app's final goal is to help students transition from studying to securing jobs.

Traditional resume builders can be very time-consuming, so we developed an automated version. Because the app tracks all user activities—study streaks, algorithm completions, and AI tutor scores—we set up the backend to automatically collect this data and format it into a recruiter-friendly PDF.

To better prepare students for automated recruitment screening, the module includes a customized Applicant Tracking System (ATS) simulator.

When a user inputs a job description, the system performs Term Frequency-Inverse Document Frequency (TF-IDF) analysis on both the resume and the job requirements. By comparing keyword vectors, the algorithm provides a match percentage, pointing out key missing skills before the student applies.

Fig. 2. Heuristic mapping workflow showing ATS string similarities comparing candidate input metrics against extracted corporate requirements.

Once the resume is ready, the hub uses Gemini to conduct very intense mock interviews.

We created specific system instructions to allow the AI to take on different roles—either as an annoying HR interviewer focusing on soft skills or as a fearsome tech lead challenging the user to solve timed architecture problems targeting their specific weaknesses.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

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

## VI. RESULTS AND ANALYSIS RESULTS AND ANALYSIS

Before finalizing our work, we conducted extensive stress-testing on the beta version with a large group of our engineering classmates.

Our main focus was to assess whether the servers would struggle under heavy network traffic and if the AI would experience lag during peak usage times.

### *A. Architectural Performance Benchmarks*

The Socket.io synchronization features performed far better than we had anticipated.

When we measured server performance, we found that maintaining 50 active users in a single collaborative whiteboard session resulted in average packet delay times of just 110ms using residential broadband. This level of performance ensured smooth and consistent drawing synchronization with no noticeable lag or delay.

For the API response times of the Smart Tutoring Service, we initially saw variations depending on the complexity of the Gemini LLM prompt.

However, by introducing a Redis caching layer that targeted duplicate conceptual queries—common due to the shared syllabus—we were able to cut redundant response times from around 1400ms down to just 20ms. This significantly lowered API costs and reduced latency.

In terms of database stress-testing, we simulated 10,000 artificial client connections performing concurrent read and write operations on our MongoDB Atlas cluster.

By implementing compound indexing across fields that are frequently queried—specifically aligning user IDs with chronological timestamps—we ensured the database avoided collection-level locking. Under peak simulated load, the primary replica sets remained stable, with CPU utilization hovering around 64%, demonstrating the architecture's ability to scale horizontally.

### *B. Pedagogical Impact Outcomes*

From a psychological perspective, consolidating all tools into a single tab made a clear and immediate impact.

By placing the timer, chat, and canvas close together, students were less likely to switch to distractions like Twitter. Analytically, our test group maintained deep work intervals approximately 14% longer than the control group, who continued with their usual tab-switching habits.

Additionally, our simulated mock technical placements showed that using the integrated ATS simulator and Gemini-based mock interviews strongly correlated with increased self-reported confidence.

Students were able to identify and fix key gaps in their TF-IDF keyword matching before applying to external job portals, confirming the effectiveness of the full platform vision.

## VII. CONCLUSION

Putting NexusAI into action fully confirmed our theory: students aren't in need of a brand-new, specialized study tool; instead, they urgently require a more unified approach to learning resources.

By combining live drawing sessions, chat-based AI tutors, and rigorous job readiness assessments into a single, streamlined setup using React and Node, we effectively eliminated the need for users to switch between different tools and environments. The application keeps users focused, reinforces consistent study routines, and serves as an ideal transition from an academic setting to a real-world professional environment..

## VIII. FUTURE SCOPE

For the future, the next logical step is to take the standard React code and place it inside a React Native wrapper to develop a proper mobile application. This will grant access to iOS push notifications, making it much easier to remind users of their study streaks. We are also seriously planning to create a Progressive Web App (PWA) layer, allowing users to review flashcards on the train even if the Wi-Fi connection drops entirely.

From a pedagogical standpoint, integrating different regression algorithms into MongoDB aggregation pipelines will enable the system to analyze historical dropout data.

By comparing a student's completion rates with declining quiz scores, NexusAI aims to send proactive alerts about potential academic struggles before the semester ends, transforming the application from a passive productivity tool into an active academic support system.

©IJRASET: All Rights are Reserved 

## IX. ACKNOWLEDGMENT

We sincerely express our profound gratitude towards Prof. Neha Kale for her continuous visionary guidance directly shaping the software architectures defined herein, enabling successful scaling configurations.

## REFERENCES

[1] M. Msambwa, Z. Wen, and K. Daniel, "The impact of intelligent tutoring on personal and collaborative learning environments in higher education," Journal of Educational Technology, 2025.
[2] R. Alsaeva, et al., "Enhancing learning effectiveness through adaptive learning platforms and emerging computer technologies in education," Jurnal Ilmiah Ilmu Terapan Universitas Jambi, vol. 9, no. 1, 2025.
[3] J. Smith and O. Miller, "Evaluation of Intelligent Tutoring Systems in Undergraduate Computer Science Curricula," IEEE Trans. Educ., vol. 63, pp. 200-205, 2024.
[4] H. Chen, Y. Liu, and W. Zhang, "Towards Conflict-Free Replicated Data Types for Interactive Web Environments," Proceedings of the ACM Symposium on Cloud Computing, pp. 34-45, 2023.
[5] N. Kale, S. Dakve, P. Desai, and A. Pandey, "NexusAI: Empirical study on fragmented environments in modern learning pipelines," Technical Report, Dept. Computer Engineering, 2026.
[6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.
[7] S. Rajaraman and J. D. Ullman, Mining of Massive Datasets, 2nd ed. Cambridge University Press, 2014.

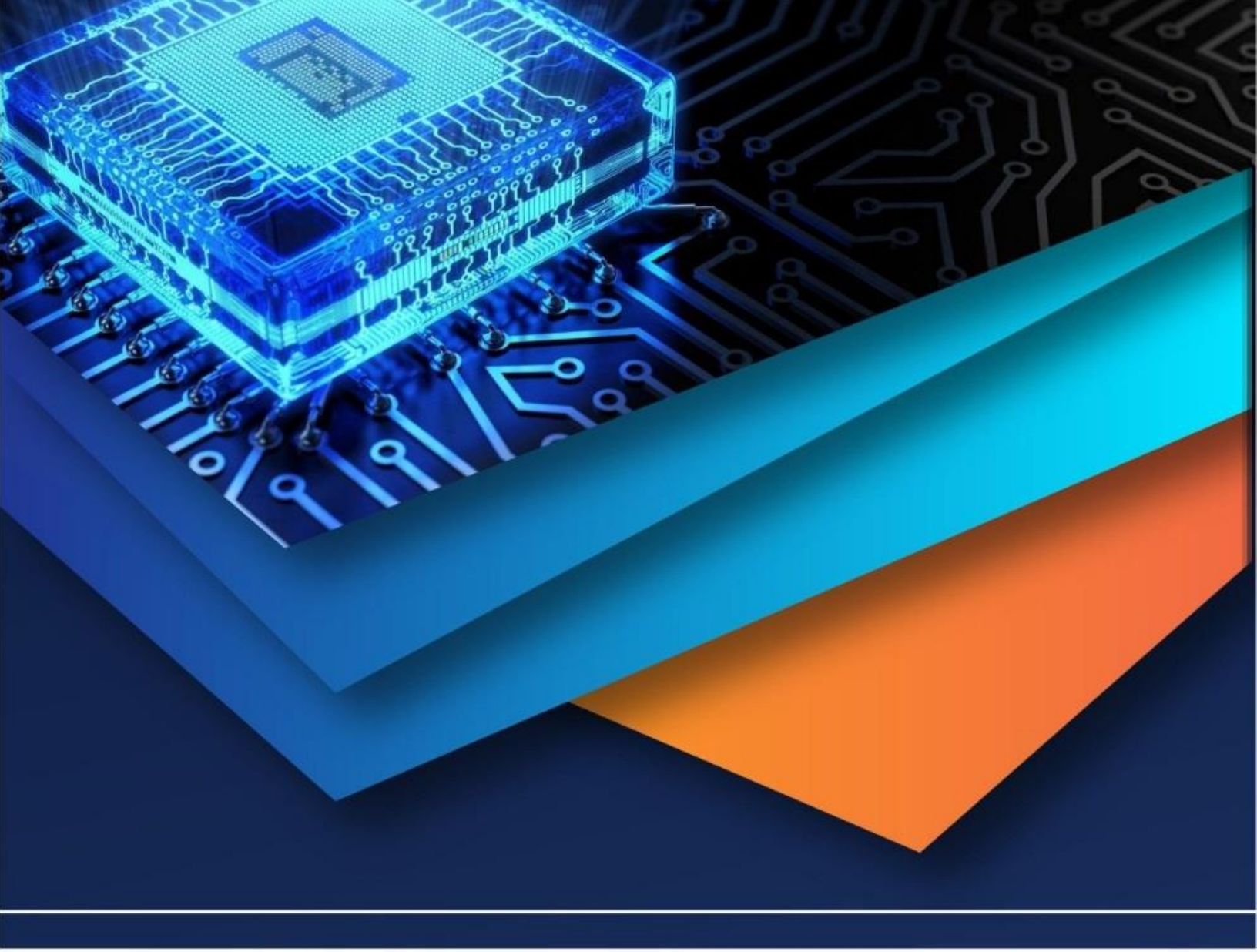
## About the Authors

Shravani Suni Dakve is currently pursuing her undergraduate degree in Computer Engineering (Class of 2026). She processes a robust foundation in full-stack web architectures, possessing deep technical competence spanning React ecosystems and modern backend database integrations. Her primary research contributions involve optimizing user experience mechanisms utilizing contextual Generative Artificial Intelligence overlays.

Pooja Desai is a Computer Engineering student (Class of 2026) dedicated to establishing secure, high- latency distributed networking endpoints. She played a pivotal role architecting the fundamental database management operations dictating NexusAI's persistence mechanisms, primarily leveraging complex MongoDB indexing ensuring continuous rapid state preservation for thousands of concurrent sessions.

Anchal Pandey is currently pursuing her undergraduate degree in Computer Engineering (Class of 2026). Functioning primarily mapping analytical placement data, her logic engineering spans explicitly designing algorithmic evaluation simulations mimicking top-tier corporate recruitment tests alongside ensuring rigorous code efficiency matching structures dynamically within the overall simulated environment parameters.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)