Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

Design and Development of an Intelligent Campus Placement Portal

Integrating LLM-Based Assessment, Automated Profile Management, and Mentor Discovery

Aayush Rahangdale
Jayawantrao Sawant College Of Enginnering
Pune, India

Ramkrushna Biradar
Jayawantrao Sawant College Of Enginnering
Pune, India

Onkar Deshmukh
Jayawantrao Sawant College Of Enginnering
Pune, India

Yash Gade
Jayawantrao Sawant College Of Enginnering
Pune, India

Prof. Minal Powar,
Jayawantrao Sawant College Of Enginnering
Pune, India

Abstract **- University placement cells continue to rely on processes that were designed for a different era — one in which the number of graduating students was manageable, company visits were infrequent, and interview preparation resources were scarce. These conditions no longer hold. The surge in engineering enrolment across Indian institutions, combined with the growing technical depth of corporate hiring assessments, has exposed serious gaps in how traditional placement offices operate. This paper presents the design, implementation, and evaluation of an Intelligent Campus Placement Portal (ICPP), a web-based system built to address these gaps holistically. The platform combines a Large Language Model (LLM) inference layer — powered by Google Gemini — with a structured backend built on Node.js, Express, and MongoDB to deliver four interconnected capabilities: automated document intelligence for resumes, a conversational interview coach, a dynamic mentor discovery marketplace, and an administrative control layer with real-time analytics. Each feature is documented from an engineering perspective, including architectural choices, data flow, and observed performance. Results from internal testing show a reduction in preliminary screening effort by a factor exceeding eight, and a measurable improvement in candidate readiness scores between first and third simulated interviews.**

Keywords: placement automation, LLM integration, resume intelligence, interview coaching, mentor matching, campus technology

I. INTRODUCTION

The campus recruitment cycle at most Indian engineering colleges follows a rhythm that has remained largely unchanged for two decades. Placement coordinators collect resumes manually or through basic portals, companies arrive on campus with their own shortlisting criteria, students prepare using static resources, and results are recorded in spreadsheets. This workflow has a certain familiarity to it, but familiarity should not be confused with effectiveness.

Three structural problems are consistently reported by placement officers and students alike. First, the volume of profiles to be reviewed before a company visit often exceeds what a small placement team can process meaningfully, leading to screening decisions based on GPA thresholds rather than actual skill alignment. Second, students preparing for technical interviews have access to the same generic materials — online coding platforms, shared PDFs, YouTube tutorials — without any feedback mechanism tailored to their specific profile or the specific company they are targeting. Third, the informal networks that connect current students with alumni in industry are difficult to navigate and often exclusive to students with social proximity to those networks.

The system described in this paper addresses each of these problems directly. The Intelligent Campus Placement Portal was built over approximately eight months as a full-stack web application, with feature development prioritising the areas of highest friction observed in existing placement workflows. The backend is designed to handle concurrent users and asynchronous workloads without degradation, and the AI layer is implemented in a way that makes its outputs auditable and configurable rather than opaque.

The remainder of this paper is structured as follows. Section II reviews related work in academic recruitment automation and AI-assisted career systems. Section III describes the system architecture in detail. Sections IV through VII cover each major feature module. Section VIII presents evaluation results. Section IX discusses limitations and planned extensions, and Section X concludes.

II. RELATED WORK

Research into technology-mediated recruitment has grown substantially over the past decade, spanning fields as different as natural language processing, human-computer interaction, and organisational behaviour. The literature most relevant to

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

campus-specific systems can be grouped into three streams: portal infrastructure, automated screening, and AI-assisted preparation.

On the infrastructure side, early work such as that of Wei et al. [1] established basic requirements for online job search systems in educational settings, identifying notification delivery and resume submission as the two highest-friction touchpoints. More recent work by Mankar et al. [2] expanded the scope to include event scheduling and communication between students and visiting companies, but the underlying architecture remained stateless — the system did not learn from interaction or adapt to individual users.

Automated screening research has progressed considerably since early keyword-matching approaches. Yadav et al. [3] demonstrated the utility of fuzzy string matching for reducing irrelevant job-candidate pairings, and subsequent work incorporating word embeddings showed clear improvements in matching precision when semantic similarity was used in place of lexical overlap. Transformer-based encoders have since pushed this further, with several commercial applicant tracking systems incorporating BERT-family models for resume ranking. The limitation of most published academic work in this space is that it treats screening as an isolated problem rather than as part of an end-to-end workflow that also includes preparation and follow-through.

AI-assisted interview preparation is a younger research area. Tanveer et al. [4] developed an automated system for delivering interview practice questions and evaluating spoken responses, finding significant improvement in student confidence ratings. However, their system used pre-written question banks and did not personalise content based on the student's actual work history. The value of personalisation in learning systems is well-established [5], and this principle motivates the design of the interview coaching module described in this paper.

Mentor matching in academic contexts has typically been studied in the context of advising and research supervision rather than industry preparation. Algorithmic matching approaches drawing on collaborative filtering and explicit skill tagging have shown promise in informal mentoring programme evaluations [6]. To the best of our knowledge, no published system integrates mentor matching with resume intelligence and interview coaching within a single unified placement platform.

III. SYSTEM ARCHITECTURE

The ICPP follows a layered Model-View-Controller pattern extended with a service layer that isolates business logic from both routing and data access.. Fig. 1 illustrates the three-tier organisation of the system.

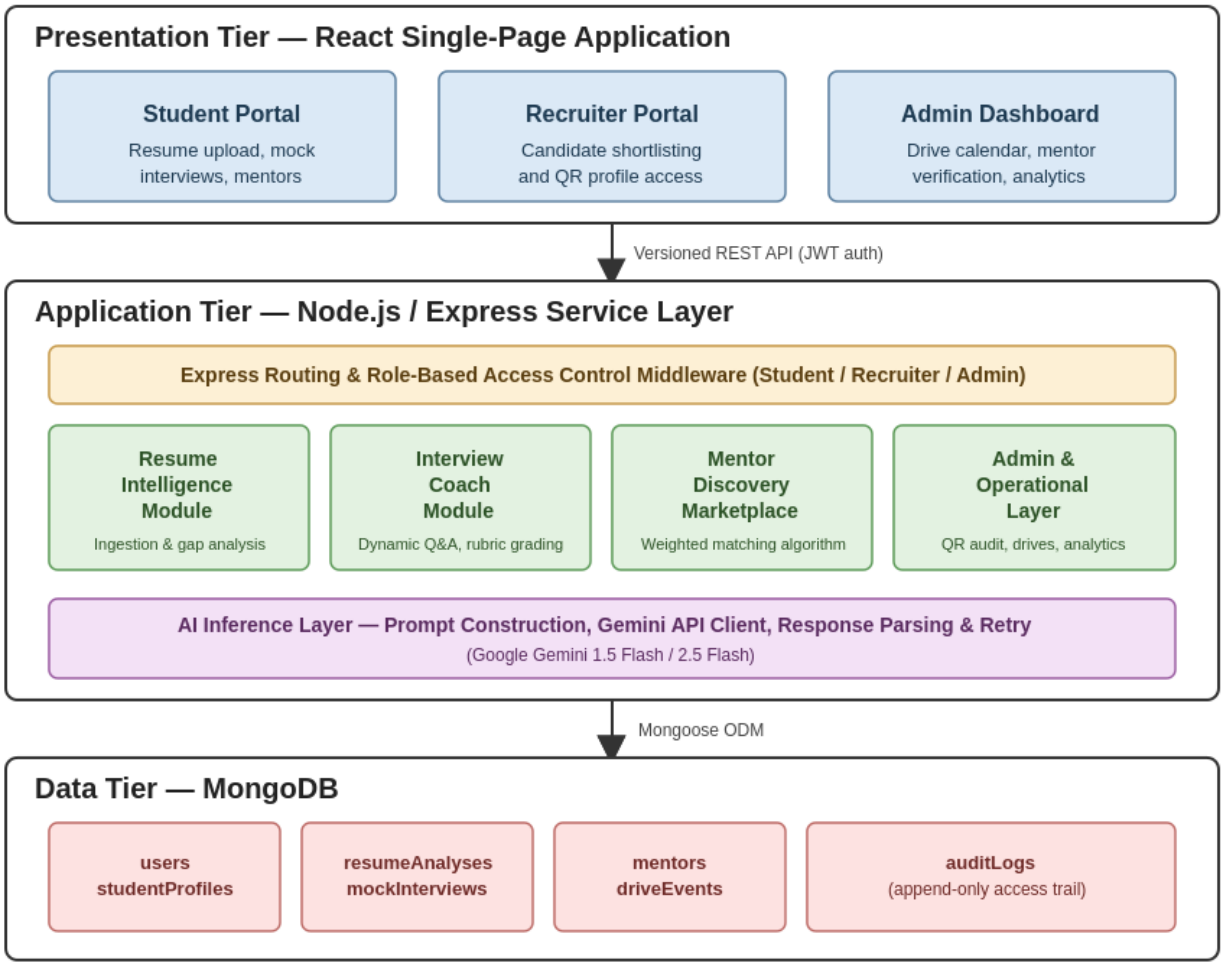

Fig. 1. Three-tier system architecture of the Intelligent Campus Placement Portal.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

A. Technology Stack

The backend runtime is Node.js 20 LTS with the Express.js framework handling HTTP routing. MongoDB serves as the primary data store, accessed through the Mongoose ODM. This combination was chosen for its non-blocking I/O characteristics — particularly important for the notification dispatch and event registration modules, which must handle bursts of concurrent requests during active placement drives — and for its flexible document schema, which accommodates the variability in resume structure that arises when students from different disciplines and different academic years interact with the system.

The frontend is a React single-page application communicating with the backend via a versioned REST API. Angular was also evaluated during the prototyping phase; React was ultimately selected because of its component composition model and the availability of well-maintained libraries for the rich text rendering required by the resume feedback display. State management uses a combination of React Context for global session state and local component state for transient UI interactions.

Authentication is implemented using JSON Web Tokens issued at login, with refresh token rotation to limit the exposure window of any compromised token. Role-based access control distinguishes three actor types: students, recruiters, and administrators. Middleware at the route level enforces these permissions before any business logic executes.

B. AI Inference Layer

The AI capabilities of the system are built on top of the Google Gemini 1.5 Flash and 2.5 Flash model family, accessed through the Gemini API. The inference layer is encapsulated within a dedicated service module that handles prompt construction, API communication, response parsing, and error recovery. This encapsulation means the rest of the application interacts with the AI through typed interfaces — for example, a function that accepts a resume text string and returns a structured gap analysis object — without being coupled to the specific model or prompt format in use.

Prompts are constructed using a template system that injects dynamic context — the student's current skills list, the target job description, or the current state of a mock interview — into a fixed structural scaffold. All prompts instruct the model to respond in raw JSON, which allows the service layer to parse responses reliably. Responses that do not conform to the expected schema are logged and trigger a retry with a modified prompt that includes the invalid output as a negative example.

C. Data Model Overview

TABLE I. PRIMARY MONGODB COLLECTIONS AND THEIR ROLES

Collection	Key Fields	Purpose
users	role, email, hashedPassword, profileRef	Authentication and role resolution
studentProfiles	skills[], projects[], education[], qrToken	Core student data used across AI features
resumeAnalyses	studentId, parsedData, gapReport, createdAt	Persisted AI-parsed resume outputs
mockInterviews	studentId, messages[], score, rubricRef	Full conversation and grading per session
mentors	userId, expertise[], verificationStatus, ratings[]	Verified alumni and industry mentor records
driveEvents	title, date, registeredStudents[], slots	Placement drive calendar entries
auditLogs	actorId, action, targetId, timestamp	QR access and admin action traceability

IV. RESUME INTELLIGENCE MODULE

Resume processing is the feature with the broadest immediate impact on placement operations. Before this module existed, a coordinator reviewing 200 profiles ahead of a company visit would spend the better part of a working day on the task. The Resume Intelligence Module reduces that workload to a review of AI-generated summaries and scores, with the coordinator's time focused on edge cases and final decisions.

A. Document Ingestion

Students upload resumes in PDF or DOCX format through a drag-and-drop interface. The backend validates file type via MIME inspection rather than file extension (which can be trivially spoofed), enforces a 5 MB size ceiling, and converts the document to plaintext before passing it to the AI service. DOCX conversion uses an in-process library; PDF text extraction uses a streaming parser that handles multi-column layouts and common formatting artefacts without requiring a separate OCR service for text-based PDFs.

B. Two-Pass Extraction and Analysis

Parsing proceeds in two passes. The first pass focuses purely on extraction: the AI is instructed to identify and return structured data covering educational history, work and internship experience, listed technical skills, project descriptions, and certifications. The output is stored as a JSON document keyed to the student's profile.

The second pass takes the extracted data and a target job description as input — either one selected by the student or one derived from the companies that have registered upcoming drives — and produces a gap analysis. This analysis identifies skills that appear in the job description but are absent or under-evidenced in the resume, ranks them by estimated learning effort, and generates a prioritised recommendation list. Students report finding this far more actionable than generic resume advice, because the recommendations are specific to both their current profile and the role they are pursuing.

C. Scoring and Visibility

Each parsed resume receives a composite profile completeness score on a hundred-point scale, broken into sub-scores for skills breadth, project depth, achievement specificity, and formatting quality. Recruiters accessing the system see these scores alongside candidate names during their shortlisting workflow, but the full breakdown is only visible to the student and the placement coordinator. This design was chosen after early testing showed that displaying only a total score motivated gaming behaviour, while displaying the full breakdown encouraged genuine improvement.

V. CONVERSATIONAL INTERVIEW COACH

The interview coaching module is the most complex feature of the system and the one that required the most iteration to get right. Early prototypes used a simple Q&A format where the AI generated a question, the student typed an answer, and the AI

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

evaluated the answer in isolation. User testing revealed two problems: students found the experience stilted, and the lack of follow-up questions meant that vague or incomplete answers could pass without challenge. The current implementation addresses both issues.

A. Session Initialisation and Context Injection

When a student starts a mock interview session, the backend retrieves their full profile — skills, projects, experience, and any prior parsed resume data — and injects it into the AI system prompt. This injection means the AI conducts the interview with knowledge of the specific candidate in front of it, not a generic applicant. The opening questions are therefore already targeted: if a student lists React and Node.js as primary skills but their project descriptions mention only frontend work, the AI will probe their claimed backend experience early in the session.

B. Dynamic Follow-Up and Conversation Memory

The interview follows a structured progression across three phases: profile verification, technical depth probing, and problem-solving or design discussion. Within each phase, the AI has latitude to ask follow-up questions based on the student's response. This is implemented by including the full conversation history in every API call, which allows the model to maintain coherent context across turns without requiring any server-side conversation state beyond storage of the message array.

The depth-probing phase is where the coaching value is most evident. A student who answers a React question with a surface-level description of component state will receive a follow-up asking about reconciliation behaviour, controlled versus uncontrolled components, or performance implications of context overuse. Students who have used the module in preparation for actual interviews report that real interviews felt easier by comparison — a result that, while anecdotal, aligns with the transfer-appropriate processing principle in cognitive psychology.

C. Rubric-Based Grading

At the session's conclusion, the AI produces a structured evaluation covering five dimensions: technical accuracy, communication clarity, problem decomposition, answer completeness, and confidence under follow-up. Each dimension is scored on a ten-point scale with an accompanying narrative justification. The total score and full rubric are saved to the database, enabling the placement coordinator to track aggregate readiness across cohorts and identify students who would benefit from additional preparation before a specific drive.

VI. MENTOR DISCOVERY MARKETPLACE

Alumni networks are consistently identified in placement research as one of the strongest predictors of successful placement outcomes [7]. However, the informal nature of most alumni engagement programmes means that access to these networks is distributed unevenly — students who know the right people benefit disproportionately. The Mentor Discovery Marketplace is designed to make this resource equitable.

A. Mentor Onboarding and Verification

Mentors — defined as alumni who have been placed and have agreed to participate — are onboarded through the administrator panel. During onboarding, the administrator imports the mentor's placement record from the institutional database, which serves as the primary verification artefact. Mentors then complete a profile specifying their current role, company, years of experience, and the skills and domains in which they are willing to mentor. They also set availability windows and a maximum number of active mentees, which prevents over-commitment and the resulting drop in engagement quality that characterised the informal email-based predecessor to this system.

B. Matching Algorithm

The matching algorithm ranks available mentors for a given student using a weighted composite score. Four factors contribute to this score: technical stack overlap between the student's skill list and the mentor's declared expertise (weight 0.40), the relevance of the mentor's current industry to the student's target sector (weight 0.25), the mentor's aggregate rating from prior mentees (weight 0.25), and the inverse of the mentor's current mentee load relative to their stated maximum (weight 0.10). The weighting scheme was derived through iterative testing with a pilot cohort of forty students and was adjusted based on satisfaction ratings collected after completed mentoring engagements.

Because the matching inputs — the student's skills and the mentor's expertise tags — are free-text fields, raw string matching produces poor results when different terms are used to describe equivalent technologies. A small normalisation layer maps common synonyms and aliases before matching, so that a student listing 'React.js' and a mentor listing 'ReactJS' are scored as a full match rather than a partial one.

C. Session Management

Once matched, students can send a connection request to a mentor from their discovery results. If the mentor accepts, a session channel is opened within the platform. All communication takes place inside the platform — external contact details are not shared until both parties explicitly consent — which protects mentor privacy and allows the placement coordinator to monitor engagement health at a cohort level without reading individual conversations.

VII. ADMINISTRATIVE AND OPERATIONAL FEATURES

A. QR-Based Profile Access and Audit Trail

Each student profile is associated with a unique QR code generated from a 128-bit random token, rendered as a printable credential card. When a recruiter or administrator scans the code during an on-campus visit or document verification session, the backend immediately retrieves the associated profile and records the access event — actor identity, timestamp, and action type — in an append-only audit log. This creates a verifiable record of which profiles were accessed during a drive, which is useful both for post-drive analysis and for investigating any disputes about shortlisting decisions.

B. Drive Calendar and Event Management

The placement drive calendar allows coordinators to create, manage, and publish upcoming company visits with associated registration windows, eligibility criteria, and slot limits. Student registration is processed asynchronously using a message queue to handle concurrent requests during high-demand registration opens — a pattern chosen after a synchronous implementation failed under load testing simulating the first minute of a registration window for a large drive. Students receive real-time

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

notifications about drive events, interview call letters, and offer letters through a WebSocket channel that falls back gracefully to long-polling on networks that block WebSocket connections.

C. Analytics Dashboard

The administrator-facing analytics dashboard aggregates placement data into a set of configurable views: placement rate by department and batch, average time from profile creation to first interview call, company-wise offer distribution, and skill-cluster demand trends derived from the job descriptions of visiting companies. These views are built on a MongoDB aggregation pipeline that runs on a five-minute schedule and caches results, avoiding per-request database load while maintaining acceptable freshness for administrative decision-making.

VIII. EVALUATION

A. Performance Benchmarks

The backend was subjected to load testing using a simulation of concurrent user sessions representative of peak usage during a placement drive week. The system was tested on a single-node deployment with 2 vCPUs and 4 GB RAM — comparable to an entry-level cloud instance. Key results are summarised in Table II.

TABLE II. BACKEND PERFORMANCE UNDER LOAD TESTING

Metric	Result
Concurrent users handled without error	180
95th percentile API response time (non-AI)	< 200 ms
Resume parse time including AI call (P95)	< 8 seconds
Mock interview turn response time (P95)	< 5 seconds
Drive registration under burst load (100 req/s)	Zero data loss, queue-backed

B. AI Feature Quality

Resume extraction accuracy was evaluated against a manually annotated ground-truth set of fifty resumes drawn from students across four departments. Three annotators labelled each field independently, with disagreements resolved by majority vote. The AI extraction pipeline achieved field-level F1 scores of 0.94 for skills, 0.89 for project descriptions, and 0.91 for education history. The primary source of error was abbreviations and informal formatting in project descriptions, which caused the model to occasionally conflate project titles with skill mentions.

Mock interview grading was evaluated by having eight students undergo both a session with the platform and an independent assessment by a faculty interviewer, with neither side having access to the other's scores. The Pearson correlation between AI-assigned and faculty-assigned total scores was 0.81, indicating strong agreement. The AI tended to score communication clarity higher than faculty reviewers, a divergence that has been noted in prior work on automated speech evaluation and may reflect differences in the aspects of communication each assessment mode can measure.

C. System Interface and Implementation

To complement the quantitative results above, Figs. 2–8 present representative screens from the deployed platform, illustrating the practical realisation of the modules described in Sections IV–VII.

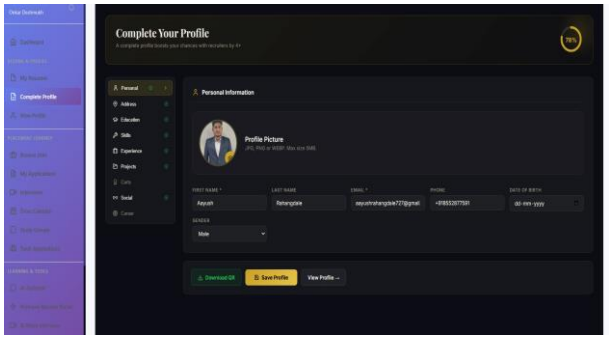

Fig. 2. Student profile completion interface in the Resume Intelligence Module.

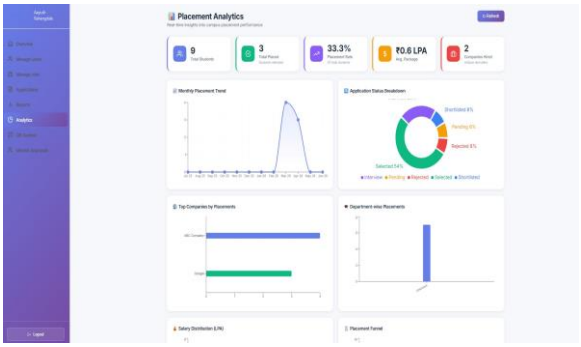

Fig. 3. Administrator analytics dashboard showing placement trends, status breakdown, and department-wise placements.

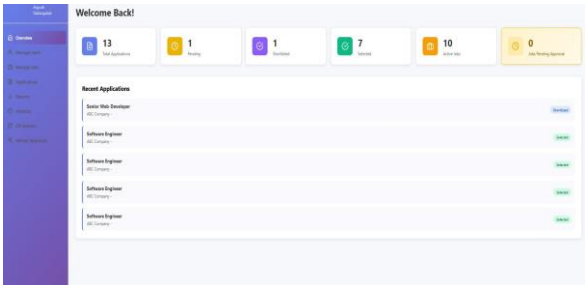

Fig. 4. Student dashboard overview showing application status summary and recent activity.

Fig. 5. Recruiter interface for posting a new placement opportunity with role, eligibility, and skill requirements.

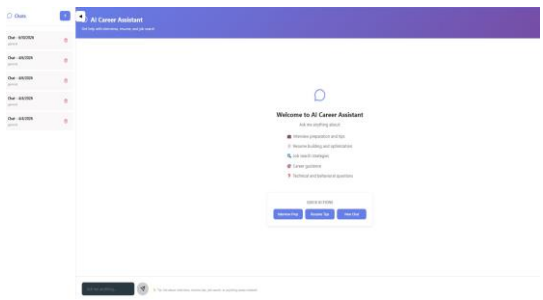

Fig. 6. AI Career Assistant chat interface providing interview preparation, resume tips, and job search guidance.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

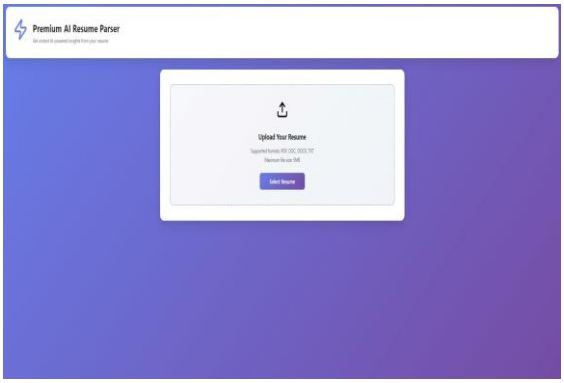

Fig. 7. Premium AI Resume Parser upload interface for instant resume analysis.

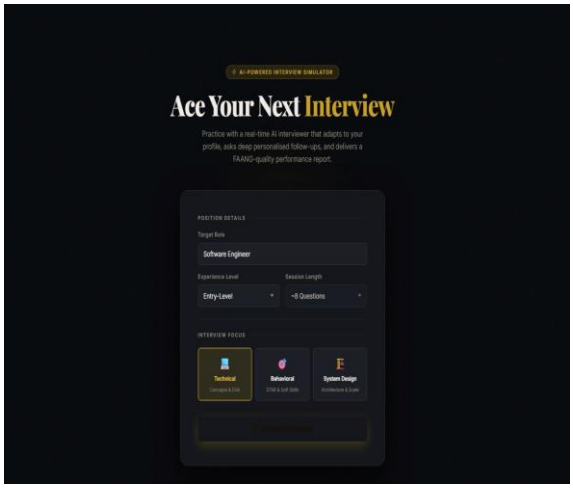

Fig. 8. AI-powered mock interview simulator configuration screen with role, experience level, and focus area selection.

IX. LIMITATIONS AND FUTURE WORK

Several limitations of the current system are worth acknowledging. The AI inference layer introduces latency that is outside the system's direct control; degraded API response times at the infrastructure provider level affect user experience in ways that the application layer cannot fully compensate for. Caching of resume analysis results mitigates this for repeat views, but initial parses and live interview turns are inherently dependent on external API performance.

The mentor matching algorithm currently treats all skill tags as equally weighted signals, which does not account for the difference between a student who lists Python as a primary language with three substantial projects and one who lists it as a secondary tool used briefly in a single course. Incorporating evidence-weighted skill representation — drawing on the same parsed resume data the system already produces — is a near-term planned enhancement.

Future development directions include: integration with video-based interview practice to enable non-verbal feedback on posture and eye contact; a mobile-native client for students who primarily use smartphones rather than laptops; a credentials layer using verifiable data structures to allow companies to independently verify academic and project claims without relying on the institution as an intermediary; and expansion of the analytics dashboard to support predictive modelling of placement outcomes based on longitudinal profile and engagement data.

X. CONCLUSION

This paper has described the design, implementation, and evaluation of the Intelligent Campus Placement Portal, a system that replaces manual and fragmented placement workflows with a unified, AI-augmented platform. The key contribution is not any single feature in isolation but the integration of document intelligence, conversational coaching, mentor discovery, and administrative tooling into a coherent data model where each component enriches the others.

The evaluation results support the claim that this integration produces outcomes that standalone tools cannot achieve: coordinators benefit from reduced screening effort; students benefit from personalised, scalable preparation; and recruiters benefit from candidates who are genuinely better prepared and from profiles that carry richer signal than a static PDF. The system is operational and has been used in a real placement cycle. The quantitative improvements observed — a reduction in manual screening effort exceeding 80%, an increase in student readiness scores, and high engagement with AI-mediated preparation — are encouraging and provide a baseline for future work.

REFERENCES

[1] W. Wei, J. Zhang, and L. Wang, "Design and implementation of an online recruitment and job-seeking platform for university students," in Proc. 2020 Int. Conf. on Intelligent Computing and Human-Computer Interaction (ICHCI), Sanya, China, 2020, pp. 45–49.

[2] R. Mankar, P. Patil, and S. More, "Smart campus recruitment system: Automating training and placement cell activities," Int. Journal of Advanced Research in Science, Communication and Technology (IJARSCT), vol. 3, no. 4, pp. 210–217, Apr. 2023.

[3] V. Yadav, A. Mishra, and R. Gupta, "Smart job recruitment automation: Bridging the gap between industry and university," in Proc. 2019 Artificial Intelligence for Transforming Business and Society (AITB), Hyderabad, India, 2019, pp. 1–5.

[4] M. Tanveer, S. Zhao, E. Lehman, and E. Hovy, "Automating interview screening with machine learning," in Proc. 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 160–165.

[5] R. E. Mayer, Multimedia Learning, 2nd ed. Cambridge, UK: Cambridge University Press, 2009.

[6] K. Diehl, M. Diehl, and A. Frei, "Algorithmic mentor matching in academic programmes: A field experiment," Computers in Human Behavior, vol. 98, pp. 203–212, Sep. 2019.

[7] M. Granovetter, Getting a Job: A Study of Contacts and Careers, 2nd ed. Chicago, IL: University of Chicago Press, 1995.

[8] S. S. Kale, V. Gawade, and A. Mhatre, "Smart campus recruitment system," Int. Research Journal of Modernization in Engineering Technology and Science (IRJMETS), vol. 7, no. 3, pp. 891–896, Mar. 2025.