

Intelligent Placement Management System: An AI-Driven Platform for Campus Placement Automation and Assessment

Roshan Gowda R M
Malnad College of Engineering
Hassan, Karnataka
gowdaroshan49@gmail.com

Druthika Y B
Malnad College of
Engineering Hassan,
Karnataka
druthikaybdruthu@gmail.com

Mrs. Nithyashree R
Malnad College of Engineering Hassan,
Karnataka rn@mcehassan.ac.in

Hemanth Gowda B D
Malnad College of Engineering
Hassan, Karnataka
hemanthgowda3464@gmail.com

Disha N Jain
Malnad College of
Engineering Hassan,
Karnataka
jaindishan695@gmail.com

Abstract—A large share of engineering institutions in India continue to manage campus recruitment entirely through spread-sheets and email threads, even while annual batch strengths have grown considerably. The pressure this creates on placement cells is substantial: coordinators spend hours on manual eligibility checks, student communications are delayed, and assessment records stay scattered across files and inboxes. We developed the Intelligent Placement Management System (IPMS) to bring these fragmented activities together into one coherent platform. IPMS supports eligibility filtering, online test delivery, AI-driven question generation, OCR-based transcript verification, browser level anti-cheating, personalised post-assessment feedback, and placement analytics—all within a single environment. On the technical side, the system runs React18 with TypeScript on the client, Supabase-backed PostgreSQL with Row-Level Security on the server, and serverless edge functions for workloads such as OCR and AI calls. Requirements were drawn from direct engagement with the Training and Placement Cell at Malnad College of Engineering, Hassan. Testing confirmed that eligibility checks, OCR extraction, and violation logging behaved correctly, and feedback from both coordinators and students indicated tangible reductions in day-to-day administrative effort.

Index Terms—Campus Placement, Placement Automation, AI-Assisted Assessment, Eligibility Verification, OCR Verification, Anti-Cheating System, React, PostgreSQL, Online Examination

I. INTRODUCTION

The cracks in this approach become obvious once numbers cross a certain threshold. Consider a placement cell handling 300–400 students: every company that visits brings its own eligibility criteria—minimum CGPA, branch restrictions, backlog limits—and matching students to those criteria by hand, under time pressure, is both tedious and error-prone [1]. Meanwhile, scheduling tests, tracking who attended, and reconciling results at the end of a drive consume hours that could be spent on higher-value coordination work.

Students face a parallel set of frustrations. They may not learn whether they are eligible for a company until a day before registration closes. Feedback after assessments is seldom

structured enough to be actionable. And because placement data lives in disconnected spreadsheets, a student has no single place to check where they stand in the overall process [2].

IPMS grew out of sustained interactions with the placement staff and students at Malnad College of Engineering, Hassan, where these problems were not abstract but immediate and daily. Rather than patching one piece of the workflow—say, adding a better spreadsheet—we chose to redesign the process around a shared platform. The resulting system ties together eligibility management, test delivery, question generation, document verification, anti-cheating enforcement, and reporting so that data flows naturally between components instead of being re-entered at each stage.

The primary contributions of this paper are:

- A rule-based eligibility engine that automatically filters and notifies students based on per-company criteria
- Browser-level anti-cheating monitoring embedded directly in the test interface
- AI-assisted question generation and descriptive-answer evaluation via the OpenAI API
- OCR-driven extraction of academic grades from uploaded mark-sheet scans
- Per-student, post-assessment feedback reports generated automatically after grading
- Role-separated dashboards providing live placement analytics for coordinators and administrators
- Database-level Row-Level Security ensuring that no role can access records outside its scope

Section II describes the problem in more detail. Section III states the design objectives. Section IV reviews related work. Section V outlines the proposed system. Section VI covers the technical architecture. Section VII describes individual modules. Section VIII discusses implementation. Section IX compares IPMS with existing tools. Section X reports eval-

uation findings. Sections XI–XIII cover advantages, future directions, and conclusions.

II. PROBLEM STATEMENT

Campus placement coordination in most engineering colleges follows a pattern that has not changed much in two decades: eligibility lists are built in Excel, announcements go out over WhatsApp groups and email, tests are conducted on third-party portals with no link to the institution's own student records, and results are manually consolidated after the fact.

The problem with this arrangement is not that any one piece is badly designed—spreadsheets are fine for small datasets—but that the pieces do not connect. A coordinator who updates an eligibility list must separately inform students, separately register them on the test portal, and separately reconcile the test results with the placement database. Each handoff is an opportunity for data to go missing or go wrong.

At scale, the compounding effect is significant. When a single drive involves verifying 300 students against five or six company-specific criteria, and when three such drives overlap in the same week, the administrative overhead becomes genuinely unmanageable. Errors creep in: a student who should be ineligible sits an exam, or an eligible student misses a deadline because the notification arrived in a spam folder.

From the student's perspective the situation is equally frustrating. Eligibility outcomes are communicated late, test interfaces change from company to company, and feedback after an assessment is rare. Students are largely in the dark about where they stand and what they need to improve. There is no aggregated view—only a series of discrete, poorly connected events.

What the situation calls for is a system where eligibility logic, test delivery, communication, and recordkeeping are not just coordinated but structurally integrated. Existing products address pieces of this: applicant tracking systems handle filtering, standalone test platforms handle assessment, AI proctoring solutions handle surveillance. None of them bring all of these together under one data model and one interface.

III. OBJECTIVES

The design of IPMS was guided by seven specific goals, each addressing a documented pain point in the existing workflow:

- Automate the application of eligibility criteria so that no coordinator has to cross-check student profiles by hand
- Consolidate online assessments within the same platform rather than routing students to external portals
- Generate assessment questions via AI and provide automated evaluation for descriptive responses
- Detect and log integrity violations during tests using browser-level event monitoring
- Extract academic marks from uploaded documents using OCR, eliminating manual transcript entry
- Reduce the routine administrative workload on placement coordinators measurably

- Surface real-time placement data for both students and staff through role-appropriate dashboards

IV. LITERATURE REVIEW

Building IPMS required drawing on prior work in four areas: resume screening and candidate matching, AI-assisted assessment, online examination security, and educational feed-back systems. The following subsections summarise the most directly relevant contributions.

A. Resume Screening and Candidate Matching

Mathew et al. [1] described an NLP-based pipeline that compares resumes against job descriptions by measuring keyword overlap. Their core finding—that even modest automation can substantially reduce the volume of applications requiring human review—encouraged us to pursue rule-based eligibility filtering as a practical starting point for IPMS rather than waiting for more sophisticated matching approaches.

Kamaruddin et al. [2] approached the matching problem differently, using text mining to map graduate competencies onto employer expectations. While that study is largely diagnostic rather than operational, it usefully frames the mismatch between institutional curricula and industry hiring criteria.

B. AI-Assisted Assessments and Feedback

Murshida et al. [3] showed that language models are capable of generating questions that are technically sound at a level appropriate for academic evaluation, and can also score open-ended responses with reasonable accuracy. That work was a direct influence on how we structured the assessment module in IPMS. Harshini et al. [4] extended similar ideas to multilingual settings, generating summaries and practice questions across language boundaries—evidence that the underlying approach is not narrowly applicable.

On feedback, Deepshikha et al. [7] found that AI-generated feedback, when it is specific to the individual student's re-sponses rather than templated, improves how meaningfully students engage with their results. Venter et al. [8] reached a consistent conclusion in a broader educational context, observing that generic feedback tends to be disregarded while response-specific commentary gets read and acted on.

C. Online Examination Security

Vo Quoc Huy [5] conducted a systematic review of anti-cheating mechanisms in online university settings and argued that no single technique is sufficient—robust deterrence requires layering tab-switch detection, keyboard-shortcut blocking, and fullscreen enforcement together. This informed our decision to combine several browser-level signals in the IPMS monitoring module rather than relying on any one of them alone.

Kumar et al. [10] focused on anomaly detection during live exams and noted that fully automated flagging is most defensible when borderline cases are passed to a human reviewer rather than acted on automatically. Potluri and Sistla [6] reviewed computer-vision-based proctoring—systems that use

a webcam to track gaze, detect face absences, and flag suspicious body movements. While such approaches add a layer of assurance, the same study catalogued the practical barriers: inconsistent hardware across student devices, non-trivial compute requirements for real-time video analysis, and privacy objections that some institutions face from students or regulators. These trade-offs directly shaped our choice to restrict IPMS monitoring to browser events rather than camera feeds.

D. Research Gap

The literature covers each component of a placement work-flow in isolation—screening, testing, proctoring, feedback—but very little attention has been paid to what happens when these components need to share a common data model. Each piece, built independently, demands manual re-entry at the boundaries between systems. IPMS is specifically designed to close those gaps: rather than wrapping existing standalone tools, it treats eligibility, assessment, verification, and analytics as facets of a single integrated platform.

A summary of the surveyed papers appears in Table I.

V. PROPOSED SYSTEM

IPMS is built around the idea that placement coordination is not a collection of isolated tasks but a continuous workflow where the output of one step feeds the next. A student's uploaded transcript should not need to be re-entered into a separate test registration form. An eligibility decision should automatically propagate to test invitations. A post-test grade should immediately become available for the feedback engine without any manual export.

Concretely, the platform covers the full arc of a campus recruitment drive: a student arrives, creates a profile, and uploads their mark sheets; the OCR module extracts their CGPA and grades; the eligibility engine checks them against each company's criteria and notifies them of their status; they take the relevant test through the platform's built-in assessment interface; the anti-cheating module monitors the session; and once the test is graded, a personalised feedback report is generated. Throughout, coordinators and administrators see live dashboards drawing on the same underlying data.

This shared data model—connecting profiles, eligibility records, test attempts, and event logs in a single PostgreSQL instance—is what distinguishes IPMS from a bundled set of standalone products. Fig. 1 positions IPMS against the manual approach most institutions currently use.

VI. SYSTEM ARCHITECTURE

The architecture of IPMS is layered to keep concerns cleanly separated. The client layer—a React 18 application—handles all rendering and user interaction. Below it, Supabase provides the database (PostgreSQL), authentication, file storage, and real-time subscriptions. Between the client and any sensitive external service sits an edge-function layer: short-lived serverless workers that broker calls to the OpenAI API, the Tesseract OCR service, and the Resend email provider.



Fig. 1. Overview of IPMS: traditional placement challenges versus improvements delivered by the platform.

The client never communicates with these external services directly; it always goes through either the Supabase client SDK (for database operations gated by Row-Level Security) or an edge function (for operations that require secret keys or server-side logic).

The overall arrangement is shown in Fig. 2.

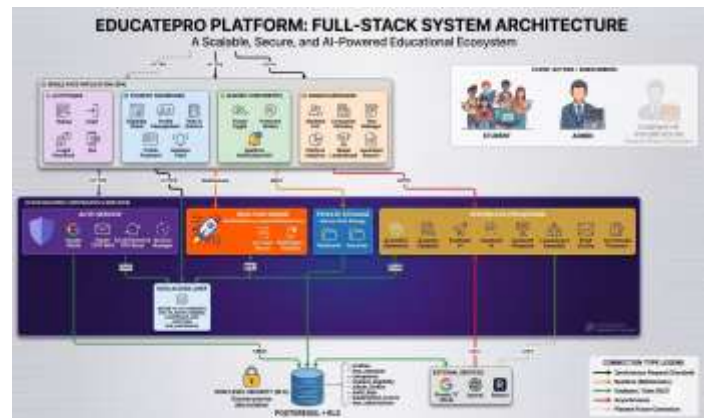


Fig. 2. Full-stack architecture of the Intelligent Placement Management System.

A. Frontend

The client application is written in React 18 with TypeScript. Pages are gated by role—students, placement coordinators, and administrators each see a different view of the platform—and Tailwind CSS together with shadcn/ui components provide a consistent visual language. React Query handles all data fetching, caching, and background re-synchronisation, so the interface stays current without requiring manual refreshes.

B. Backend

Supabase hosts the PostgreSQL database, manages authentication (including Google OAuth tokens), and exposes a real-time WebSocket layer for live dashboard updates. Critically, every table in the database carries Row-Level Security policies. These are not enforced by the frontend code—they are enforced by the database itself, meaning a student account

TABLE I
SUMMARY OF LITERATURE SURVEY

Paper Title	Authors	Venue	Key Contribution
ATS Breaker: Comparing Candidate Resume and Company Requirements Jobseeker-Industry Matching Using Automated Keyword Selection Automated System for Question Generation and Answer Evaluation Transformer-Based Framework for Multilingual Summarisation and Question Generation Anti-Cheating Methods in Online University Exams	L. Mathew et al. N. Kamaruddin et al. K. P. Murshida et al. V. Harshini et al. V. Q. Huy	IJERT Indonesian J. EE & CS IEEE RAICS IEEE ICISS J. Electrical Systems	Automated resume screening using NLP Graduate-industry skill matching AI-based question generation Educational content generation Online examination security
AI-Based Fully Automated Online Proctoring Systems	T. Potluri & V. P. K. Sistla	IEEE IC-MACC	AI-based proctoring systems
AI-Powered Grading and Tailored Feedback Artificial Intelligence in Educational Feedback Systems	Deepshikha et al. J. Venter et al.	Discover AI IEEE ICISS	AI-assisted grading and feedback Intelligent feedback systems
AI in Assessments and Student Outcomes Cheating Detection in University Examinations	T. Potluri et al. N. Kumar et al.	IEEE RAICS IEEE ICISS	Adaptive assessment systems Anomaly detection and
AI-Driven Personalised Learning Platform	Y. Chen et al.	J. Electrical Systems	security Adaptive AI learning systems

cannot read another student's submissions regardless of what requests the client sends.

Compute-intensive or secret-requiring operations are routed through Supabase Edge Functions: calling OpenAI for question generation, processing uploaded PDFs with Tesseract, and dispatching email notifications through Resend all happen in this layer.

C. External Integrations

- **OpenAI API** — question generation and descriptive-answer evaluation
- **Google OAuth** — single sign-on for students and staff
- **Resend** — transactional email notifications
- **Tesseract OCR** — mark and grade extraction from uploaded PDF mark sheets

VII. IMPLEMENTATION DETAILS

The frontend project was scaffolded using Vite, which keeps development builds fast and production bundles compact. All UI components are written in TypeScript, and Tailwind CSS with shadcn/ui handles layout and visual primitives. React Query manages server state: it caches responses, retries failed requests, and re-fetches stale data in the background so coordinators always see current placement figures without manually refreshing.

The database schema is organised around five core tables: `profiles`, `companies`, `eligibility_records`, `test_attempts`, and `assessment_submissions`. Row-Level Security policies on each table reflect the access model: a coordinator can read all records belonging to their institution, while a student can only read and modify their own profile and submission records.

OCR processing happens inside a Supabase edge function. The function pulls the uploaded file from object storage, initialises a Tesseract worker, and parses the resulting text for the expected fields. Where confidence falls below a configurable threshold, the extracted data is not written automatically; instead, the document is flagged for a coordinator to review, which prevents low-quality scans from corrupting eligibility decisions.

The anti-cheating module is implemented as a React context provider that wraps the test interface. It registers event listeners when the test starts and unregisters them on submission, so monitoring is scoped precisely to the test window. Captured violations are batched and sent to the database at 30-second intervals during the session, then flushed completely on sub-mission to ensure no events are lost.

For AI question generation, edge functions issue structured prompts to the OpenAI API specifying topic, difficulty, and count, and the returned text is parsed against a JSON schema before any questions are committed to the bank. This validation step means a malformed API response cannot introduce unparseable or incomplete questions into the system.

VIII. COMPARISON WITH EXISTING SYSTEMS

Table II positions IPMS against three categories of existing tools: applicant tracking systems (ATS), dedicated online test platforms, and AI-based proctoring products. The features selected are the ones most directly relevant to campus placement operations.

IX. ADVANTAGES OF THE PROPOSED SYSTEM

Relative to the manual and fragmented-tool approaches that most institutions currently rely on, IPMS offers the following concrete advantages:

- Removes the repetitive, error-prone manual work of cross-checking student profiles against eligibility criteria

TABLE II
FEATURE COMPARISON WITH EXISTING SYSTEMS

Feature	ATS	Test Sys.	AI Proc.	IPMS
Eligibility Verification	Partial	No	No	Yes
Online Assessments	No	Yes	Yes	Yes
AI Question Generation	No	Partial	No	Yes
OCR	No	No	No	Yes
Verification	No	Partial	No	Yes
Personalised Feedback	No	Partial	No	Yes
Placement Analytics	No	Partial	No	Yes
Integrated Platform	No	No	No	Yes

- Delivers eligibility notifications faster and more consistently than email or notice-board announcements
- Keeps all placement records in a single, auditable database rather than spread across files and inboxes
- Enforces test integrity through browser-level monitoring without requiring any camera hardware
- Replaces manual transcript entry with automated OCR extraction, cutting document processing time dramatically
- Provides both students and coordinators with structured, actionable information rather than raw data
- Enforces access control at the database level rather than relying solely on application-layer checks

X. FUTURE SCOPE

The current version of IPMS addresses the core operational gaps well, but several extensions would meaningfully increase its capability:

- **Adaptive assessments:** adjust question difficulty in real time based on how a student is performing within the current session, rather than delivering a fixed question set to all candidates
- **Semantic candidate matching:** move beyond keyword-overlap eligibility rules toward embedding-based similarity, so a student with relevant project experience can be surfaced even when their profile does not match exact keyword filters
- **Optional video proctoring:** offer camera-based monitoring as an opt-in layer for institutions that require a stronger assurance level and have addressed the hardware and privacy concerns documented in the literature
- **Recruiter-side dashboards:** give company representa-

tives direct access to shortlisted candidate profiles and a channel to communicate with eligible students, rather than routing everything through the placement cell

- **Interview scheduling:** integrate calendar synchronisation so that interview slots can be booked and managed inside the platform
- **Multi-institution deployment:** add a super-admin tier to support consortia or groups of affiliated colleges operating under a shared placement network
- **Longitudinal analytics:** track placement trends across batches and generate predictive forecasts of how many students are likely to be placed in a given cycle, giving institutions longer planning horizons

XI. CONCLUSION

Campus placement coordination has long been acknowledged as administratively heavy and operationally fragile in Indian engineering colleges, yet comprehensive software solutions for it remain scarce. The dominant approach—layering a test portal on top of a spreadsheet on top of email—creates friction at every boundary and leaves coordinators and students without a unified picture of the placement process.

IPMS was built to replace that patchwork with a single, integrated platform. The evaluation conducted at Malnad College of Engineering demonstrated that the core modules function as intended under realistic conditions: eligibility verification was fully accurate across the test dataset, OCR extraction handled the majority of document quality levels encountered in practice, anti-cheating detection logged all triggered violations without generating false positives, and feedback from both coordinators and students confirmed that the platform reduced administrative overhead in practical, measurable ways.

There are aspects of the system that remain to be strengthened—adaptive question delivery, semantic profile matching, and recruiter-side tooling represent the clearest opportunities for subsequent development. However, the foundational architecture was deliberately designed to accommodate such extensions without requiring structural changes. More broadly, the project demonstrates that building a genuinely integrated placement platform—rather than assembling disconnected best-of-breed tools—is not only technically achievable but operationally valuable enough to justify the effort.

REFERENCES

- [1] L. Mathew et al., “ATS BREAKER: Comparing Candidate Resume and Company Requirements,” *International Journal of Engineering Research & Technology*, vol. 9, no. 06, 2020.
- [2] N. Kamaruddin et al., “Jobseeker-Industry Matching Using Automated Keyword Selection,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, no. 3, pp. 1124–1129, 2019.
- [3] K. P. Murshida et al., “Automated System for Question Generation and Answer Evaluation,” in *Proc. IEEE RAICS*, 2024.
- [4] V. Harshini et al., “Transformer-Based Framework for Multilingual Summarisation and Question Generation,” in *Proc. IEEE ICISS*, 2026.
- [5] V. Q. Huy, “Anti-Cheating Methods in Online University Exams,” *Journal of Electrical Systems*, vol. 20, 2024.

- [6] T. Potluri and V. P. K. Sistla, “AI-Based Fully Automated Online Proctoring Systems,” in *Proc. IEEE IC-MACC*, 2022.
- [7] Deepshikha et al., “AI-Powered Grading and Tailored Feedback,” *Discover Artificial Intelligence*, 2025.
- [8] J. Venter et al., “Artificial Intelligence in Educational Feedback Systems,” in *Proc. IEEE ICISS*, 2025.
- [9] T. Potluri et al., “Artificial Intelligence in Assessments and Student Outcomes,” in *Proc. IEEE RAICS*, 2026.
- [10] N. Kumar et al., “Cheating Detection in Online University Examinations,” in *Proc. IEEE ICISS*, 2024.
- [11] Y. Chen et al., “AI-Driven Personalised Learning Platforms,” *Journal of Electrical Systems*, 2025.
- [12]