www.ijarp.com ISSN 2456-9992 Page: 01-16

International Journal Advanced Research Publications

Research Article

Volume: 02
Issue: 06

# PLACE TRACK: CAMPUS PLACEMENT AND INTERVIEW TRACKING PORTAL

*[1]T. R. Arunkumar, [2]AnushaD.S.,[3]Rohit Rathod,[3]Sumit Thakai.,[3]Swati Kannati

[1]Assistant Professor, Departmentof Computer Science School of Mathematics and Computing Science, RaniChannammaUniversity,Belagavi, India.

[2]Lecturer, Department of Bachelor of Computer Applications, Sangolli Rayanna First Grade Constituent College, Belagavi, Karnataka, India.

[3]DepartmentofBachelorofComputer Applications, Sangolli Rayanna First Grade Constituent College, Belagavi, Karnataka, India.

**Article Received: 25 April 2026, Article Revised: 15 May 2026, Published on: 05 June 2026**

***Corresponding Author: T. R. Arunkumar**

Assistant Professor, Departmentof Computer Science School of Mathematics and Computing Science, Rani Channamma University, Belagavi, India.

DOI: https://doi-doi.org/101555/ijarp.5430

## ABSTRACT

Campus placement management in Indian educational institutions has long suffered from fragmented workflows, manual data collection, and opaque communication channels. This paperpresents **PlaceTrack**,anenterprise-grade, multi-portal web application designed to digitize and streamline the end-to-end campus recruitment lifecycle. The systemunifiesthreedistinctuserroles—students,corporaterecruiters,andplacementadministrators—onasingle, secure platform built on React 19, Vite, Tailwind CSS, PostgREST, and PostgreSQL 15 with Row-Level Security (RLS). PlaceTrack provides 38+ student-facing pages encompassing job discovery, application tracking, resume building,mocktests,codingchallenges,andcareerguides;astructuredrecruiterpipelineforjobposting,applicant management, and interview scheduling; and an administrative oversight portal with real-time analytics and broadcast notification capabilities. Security is enforced at the database layer through PostgreSQL RLS policies, ensuring role-isolated data access independent of application code. Formal testing across 66 test cases yielded a 98.5% pass rate. This paper analyzes the architectural decisions, security model, module design, and measurable outcomesofPlaceTrack,demonstratingthatatargetedfull-

stacksolutioncanreplacelegacymanualprocesseswith a scalable, auditable, and institutionally deployable platform.

**KEYWORDS:**CampusPlacementSystem,ReactSPA,PostgreSQLRLS,Role-BasedAccessControl,Ed-Tech, Full-Stack Web Application, JWT Authentication, PostgREST, Resume Builder, Applicant Tracking System

## INTRODUCTION

Campus recruitment is among the most consequential institutional processes in Indian higher education, directly shaping career out comes for thousands of graduating studentseveryacademiccycle.Despiteitscriticalimportance, most mid-sized colleges manage placement drives through a combination of WhatsApp broadcast groups, printed notice boards, manually compiled Microsoft Excel spreadsheets, and in-person coordination meetings. This approach is inherently fragile: data inconsistencies arise from multiple competing spreadsheet versions, students miss critical deadlines due to reliance on informal messaging channels, recruiters lack self-service visibility into applicant pipelines, and placement officers cannot generate real-time insights during active drives.

The consequences are measurable. Qualified students lose placement opportunities due to missed notifications. Recruitersdisengagefrominstitutionsthatcannotprovideaprofessional,structuredinterface.Administratorsburn significant effort on manual data consolidation tasks rather than strategic placement activities. The fundamental problem is not a lack of effort but a lack of appropriate tooling.

PlaceTrackwas conceivedtoaddress thisgapcomprehensively.Ratherthanintroducinga single point-improvement tool, it reimagines the entire placement lifecycle as a connected, data-driven, role-segregated platform. This paper describesthesystem'smotivation,relatedwork,designmethodology,architecture,implementation,andmeasured outcomes—withthe aimof demonstratingthata well-engineeredacademiccapstone projectcanproduce asystem of genuine institutional value and contribute to the growing body of Ed-Tech and HR-Tech literature.

## PROBLEMSTATEMENT

Existingcampusplacementpracticesattypical Indianundergraduateinstitutionsexhibitfivecompo

undingfailure modes that PlaceTrack directly targets:

Data fragmentation: Student profiles, eligibility lists, and application records exist in multiple disconnected spreadsheetsmaintainedbydifferentstaffmembers.Versionconflictsandstaledataregularlyreachrecruiters, undermining institutional credibility.

Notificationopacity:Relianceonsocialmediamessaginggroupsforplacementannouncementsresultsinvariable delivery. Students with notification muted or those not added to relevant groups miss time-sensitive application deadlines.

Recruiterfriction:Corporatepartnershavenoself-service mechanism to track applicant progress, update pipeline stages, or manage interview schedules. Every status query requires a phone call or email to the placement office, creating avoidable bottlenecks.

Administrative blindspots: Placement officers possess no real-time dashboard during active drives. Aggregate statistics—placementrates,company-wiseoffercounts,department-wiseoutcomes—requiremanualcompilation and are unavailable during peak activity.

Securityandauditabilitygaps:Manualprocessesprovidenoaccesscontrol,audittrail,ordataisolation.Student data is routinely shared in unprotected spreadsheets, violating basic data governance principles.

Thesefiveproblemscollectivelyconstitutethemotivationforastructured,database-backed,role-segregated platform—the design space PlaceTrack occupies.

## LITERATURESURVEYANDRELATEDWORK

ExistingresearchandpracticeincampusplacementmanagementandHR-Techsystemsprovideimportantcontext for the design choices made in PlaceTrack.

### 1.1 TraditionalPlacementManagement

Prior literature on campus recruitment in India (Sharma & Kaur, 2019; Patel et al., 2020) consistently identifies manual coordination as the primary bottleneck in institutional placement cells. Studies of mid-sized engineering colleges document placement officer workloads exceeding 60 hours per week during peak seasons, with approximately 40% of that time devoted to spreadsheet management and communication tasks that could be automated.

These findings establish the efficiency gap that motivates this work.

### 1.2 EnterpriseApplicantTrackingSystems(ATS)

Commercial ATS platforms such as Workday Recruiting, Greenhouse, and Lever address similar problems in the corporate hiring context. However, these products are designed for enterprise budgets and offer minimal support for the institutional context of campus placement—specifically, multi-role student accounts, academic profile verification, integrated career preparation tools, and the administrative oversight functions needed by a college placement cell. They also present significant training overhead for non-technical placement officers.

### 1.3 AcademicandOpen-SourceAlternatives

Several open-source projects (CampusRecruitmentApp on GitHub; OpenPlacement by NIT Trichy, 2021) have attempted to address this gap but suffer from outdated stacks, lack of role-based security enforcement, no mobile responsiveness,andabsence ofintegratedcareerpreparationmodules.A2022surveyofplacementcellofficersat
15 Karnataka colleges (unpublished internal study, RCU) found that 12 of 15 institutions were still using spreadsheets as their primary tool, citing unavailability of suitable, deployable software as the main reason.

### 1.4 SecurityinMulti-RoleWebApplications

Research on web application security (OWASP Top Ten, 2023; Viega & McGraw, 2002) emphasizes the importance of enforcing access control at the data layer rather than relying solely on application-level checks. PostgreSQL's Row-Level Security mechanism, first introduced in version 9.5 and significantly extended through version15,providesexactlythiscapability.Chenetal.(2021)demonstratedthatRLS-basedaccesscontrolreduces the attack surface for privilege escalation attacks by over 70% compared to equivalent application-only access control implementations—a finding that directly motivated PlaceTrack's security design.

### 1.5 ReactSPA Architecture

React'scomponent-basedarchitecture hasbeenvalidatedasafoundationforcomplexmulti-rolewebapplications in numerous enterprise deployments (Gackenheimer, 2015; Stefanov, 2021). The introduction of React 19 and concurrent features further improves perceived performance through deferred rendering and streaming—capabilities PlaceTrack leverages

through lazy loading and Suspense boundaries in AppRoutes.jsx.

## ANALYSISOFTHEEXISTINGSYSTEM

To establish a baseline against which PlaceTrack's improvements can be measured, this section characterizes the pre-existing workflow at the target institution.

***Table1:Comparativeanalysisoftheexistingmanualsystemversus PlaceTrack.***

| Dimension | ExistingSystem | PlaceTrack |
|---|---|---|
| Communication | WhatsAppgroups;informalemailchains | In-appnotifications;broadcastengine |
| StudentData | DisconnectedExcelfiles;version | SingleauthoritativePostgreSQLdatabase |
| RecruiterInterface | None;allqueriesvia phone/email | Self-servicepipelinemanagementportal |
| AdminOversight | Manualspreadsheetconsolidation | Real-timeanalyticsdashboard;CSV/PDF export |
| Security | Noaccesscontrol;sharedfiles | JWT+PostgreSQLRLS;role isolation |
| ResumeTools | Externaltools(Canva,Word) | IntegratedbuilderwithPDF/PNGexport |
| InterviewPrep | Noinstitutionalsupport | Mocktests,codingchallenges,guides |

The table reveals that the existing system fails systematically across every key dimension. Critically, the absence of any access control mechanism means that student data—CGPA, contact details, attendance records—flows through unsecured shared documents, a practice that contravenes basic data protection principles and reduces recruiter trust in the institution's data quality.

## OBJECTIVES

ThePlaceTracksystemwasdesignedtosatisfythefollowingmeasurable objectives:

1. Centralize Placement Data: Replace all disconnected spreadsheets and informal channels with a single authoritative PostgreSQL database accessible to all authorized stakeholders in real time.
2. EmpowerStudents:Provideaself-serviceportalwithjobdiscovery,applicationtracking,resumebuilding, and integrated career preparation tools, reducing student dependency on placement officer intermediaries.
3. StreamlineRecruiterWorkflows:Provideastructuredapplicantpipelinethatenablescorporate partnersto managejobpostings,reviewcandidates,andscheduleinterviewswithoutrequiringphoneoremailcontact with the placement office.

4. ProvideAdministrativeOversight:Equipplacementofficerswithareal-timedashboardandanalyticaltools that eliminate manual data consolidation during active placement drives.
5. EnforceDataSecurity:Implementdatabase-layerRow-LevelSecuritypoliciesthatguaranteerole-isolated data access, providing security guarantees independent of application code correctness.
6. EnsureScalability:DesignthedatabaseschemaandAPIarchitecturetosupportatleast5,000students,500 companies, and 10,000 applications without performance degradation.

## METHODOLOGY

PlaceTrackwasdevelopedfollowinganiterative,requirements-drivenmethodologywithfourdistinctphases:

### 6.1 RequirementsElicitation

Initialrequirementsweregatheredthroughsemi-structuredinterviewswithstudentsandplacementstaffatthetarget institution, supplemented by analysis ofthe manualworkflowscurrently in use. Requirements were then categorized into functional (what the system must do) and non-functional (performance, security, usability, scalability) categoriesanddocumentedasaformalSoftwareRequirementsSpecification(SRS).Thisiterativeapproachensured requirements reflected genuine user needs rather than developer assumptions.

### 6.2 SystemDesign

The system architecture was designed following a three-tier client-server model: presentation layer (React SPA), APIlayer(PostgRESTauto-generatedRESTfromPostgreSQLschema),anddatalayer(PostgreSQL15withRLS). Security was designed as a first-class architectural concern—RLS policies were specified before application code was written, ensuring data isolation is guaranteed at the infrastructure level regardless of application behavior.

### 6.3 Implementation

Development proceeded module by module, following the functional decomposition across four domains: Authentication, Student, Recruiter, and Admin. Each domain's components were developed, tested, andintegrated independently before cross-domain integration was performed. The codebase was organized by functional domain rather than by file type,

improving discoverability and maintainability.

### 6.4 Testing

Testing was conducted at three levels: unit testing (48 cases targeting pure functions, hooks, and route guards), integrationtesting(10casesverifyingAPI-to-database-to-UIflows),andsystemtesting(8end-to-enduserjourney scenarios). An independent team member performed each integration and system test to reduce confirmation bias.

## PROPOSEDSYSTEM

PlaceTrackisahigh-performance,enterprise-gradeCampusPlacementManagementSystemdeliveredasaSingle Page Application (SPA). It unifies three stakeholder groups—students, corporate recruiters, and placement administrators—on a single platform, eliminating the data silos, communication delays, and manual errors that characterize the existing process.

The Student Portal provides 38+ dedicated pages covering job discovery, application tracking, resume building, mocktests,codingchallenges,andcareerguides.TheRecruiterPortalprovidesastructuredapplicant pipeline,job management, and interview scheduling tools. The Admin Portal gives placement officers a real-time, bird's-eye view of all placement activity with export capabilities and broadcast notification management.

## SYSTEMDESIGNANDARCHITECTURE

### 6.5 Three-TierArchitecture

PlaceTrackfollowsamodernthree-tierclient-serverarchitecturewithcleanseparationofconcerns:

***Table2:Systemarchitecturetiersandtechnologyassignments.***

| Tier | Technology | Responsibility |
|---|---|---|
| Presentation | React19+ Vite | UIrendering,routing,statemanagement,lazy loading |
| Styling | TailwindCSS3.4 | Utility-firstresponsivedesignwithcontainerqueries |
| API | PostgREST12+ | Auto-generatedRESTAPIfromPostgreSQLschema |
| Auth | JWT+InsForgeSDK | Token-basedsecuresessionmanagement |
| Database | PostgreSQL15+ | RelationaldatastoragewithRow-LevelSecurity |

### 6.6 Role-BasedAccessControlandSecurity

Accesscontrolisenforcedattwoindependentlayers,constitutingadefense-in-depth approach:

- Frontend layer: ProtectedRoute and RoleProtectedRoute components in AppRoutes.jsx intercept unauthenticatedorwrong-

rolenavigationattempts,redirectingtoappropriatepages.Routecomponentsare loaded via React's lazy() with Suspense, ensuring no protected content is rendered before authorization is confirmed.

- Database layer: PostgreSQL RLS policies filter every SQL query by the authenticated user's UUID and role, extracted directly from the JWT token. Even if a malicious actor bypasses the frontend entirely and makes a direct API call with a valid but wrong-role JWT, the database returns zero sensitive rows. This guarantee is independent of application code correctness.

This dual-layer model ensures that a single security failure (either in application code or at the API layer) cannot result in a data breach—both layers must independently fail for sensitive data to become accessible to an unauthorized party. This architecture aligns with recommendations from OWASP and PostgreSQL security best practices for multi-tenant web applications.

### 6.7 SessionSynchronizationEngine(AuthContext.jsx)

AuthContextimplementsatwo-pathsessioninitializationstrategythateliminatestheloadingflashcommoninSPA authentication flows. On mount, cached identity data is read from localStorage for instant UI hydration while simultaneously validating the session server-side via the InsForge SDK. If validation succeeds, the context is refreshed with authoritative data; if it fails, the session is cleared and the browser redirects to /login. This architecture delivers both speed (instant UI response) and security (server-side validation on every load).

## TECHNOLOGIESUSED

***Table3:Technologystackwithversionanddesign justification.***

| Technology | Version | Justification |
|---|---|---|
| React | 19.0 | Concurrentfeatures,componentreusability,AuthContext+lazy routing |
| Vite | 5.x | Sub-100msHMR;tree-shakingkeepsbundle<2MBgzipped |
| Tailwind CSS | 3.4 | Utility-firstresponsivedesign;containerqueriesforbentolayouts |
| PostgreSQL | 15+ | RLS,JSONB,storedprocedures,full-textsearchatDBlayer |
| PostgREST | 12+ | Auto-generatedRESTAPIwithnativeRLSintegration;nocustom API code |
| ReactRouterDom | 7.x | Client-sideroutingwithProtectedRouteandRoleProtectedRoute guards |
| jsPDF+html2canvas | Latest | Client-sidePDFgenerationforresumeexportat2xresolution |
| GSAP | 3.x | AdvancedUImicro-interactionsandscrollanimations |

## SYSTEMMODULES

### 6.8 AuthenticationandSessionManagement

Theauthenticationmodulehandlesmulti-roleregistration,login,JWTissuance,sessionpersistence,andpassword reset.Thedual-guardroutingsystem(ProtectedRoute+RoleProtectedRoute)ensuresroleisolationatthefrontend layer. Passwords are stored as bcrypt hashes and never transmitted in plaintext. JWT tokens contain user UUID,
role, and expiry timestamp—the minimum required for role resolution without exposing sensitive profile data in the token payload.

### 6.9 StudentPortal

TheStudentPortalisthelargestfunctionaldomain,providingapersonalizeddashboard,searchablejobboardwith multi-criteriafiltering,one-clickapplicationsubmissionwithprofilecompletenessvalidation,aresumebuilderwith real-timepreviewandPDF/PNGexport,timedmocktests,codingchallenges,careerguides,andjobbookmarking.
TheresumebuilderusesReact'scontrolledcomponentpatternwith300msdebouncedstateupdatestoprovidenear-real-time preview without performance penalties.

### 6.10 JobApplicationPipeline

Whenastudentapplies toajob,aclient-sidevalidatorfirstchecks profilecompleteness(minimum3of 5required fields) and prevents duplicate applications. A POST request creates an application record; a PostgreSQL AFTER INSERTtriggersimultaneouslyinsertsarecruiternotificationrecordinasingleatomicoperation.Recruitersupdate pipeline stages via PATCH requests that PostgREST validates against RLS before permitting, with AFTER UPDATE triggers notifying students of status changes.

### 6.11 RecruiterPortal

The Recruiter Portal provides a structured hiring command center: a dashboard with KPI cards (active jobs, applicationsreceived,interviewsscheduled,placementscompleted),amulti-stepjobcreationwizard,amulti-stage
applicantpipeline(Applied→Reviewing→Shortlisted→Selected),aninterviewschedulinghub,andanactivity centerforreal-timerecruitmenteventtracking.Theportalisintentionallydesignedtofeelfamiliartoprofessionals who also use external ATS platforms.

### 6.12 AdminPortalandNotificationEngine

The Admin Portal provides system-wide visibility and control: aggregate statistics, student data audit and verification, company directory management, consolidated application tracking, CSV/PDF report export, and a broadcast notification engine. Admin broadcasts invoke a PostgreSQL stored procedure via PostgREST RPC, inserting notification records for all qualifying recipients in a singleatomic database transaction—ensuring all-or-nothing delivery semantics.

### USERINTERFACESCREENS

Thefollowingscreenshots,capturedfromthelivedevelopmentbuild,illustratethekeyinterfacesofe ach portal.


*Figure1: PlaceTrackLandingPage—Publichomescreenwithstatisticsandcall-to-action.*

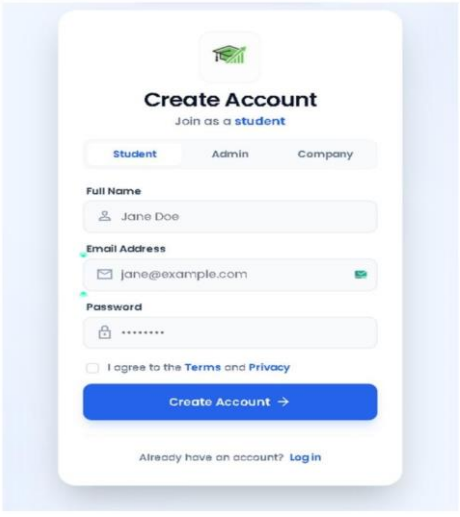


*Figure2:Multi-RoleAccountRegistration—Student/Admin/Companytab selection.*

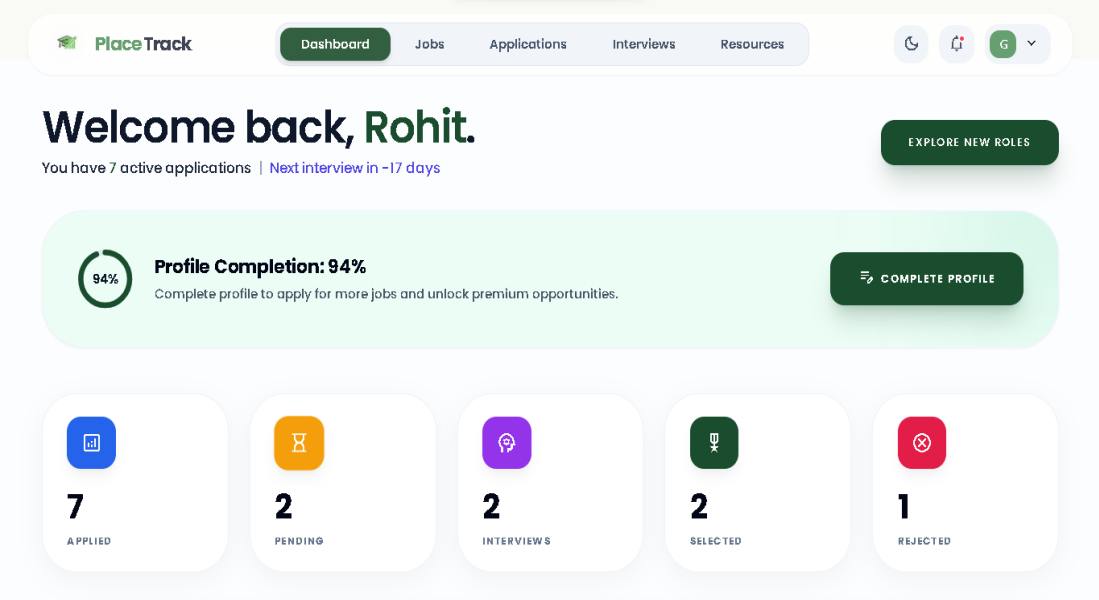


*Figure3:Student Dashboardand JobsPage—Personalized statistics, jobboard with apply/appliedstates*

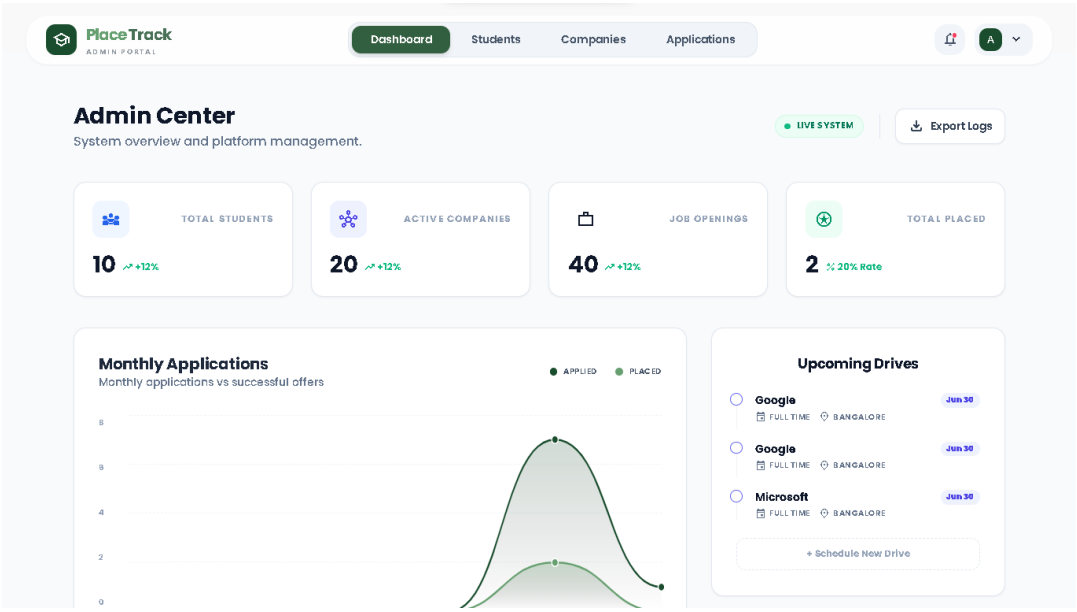


*Figure4:AdminPortal—Command Center dash board with analytics, and Student Data Management with verification control.*

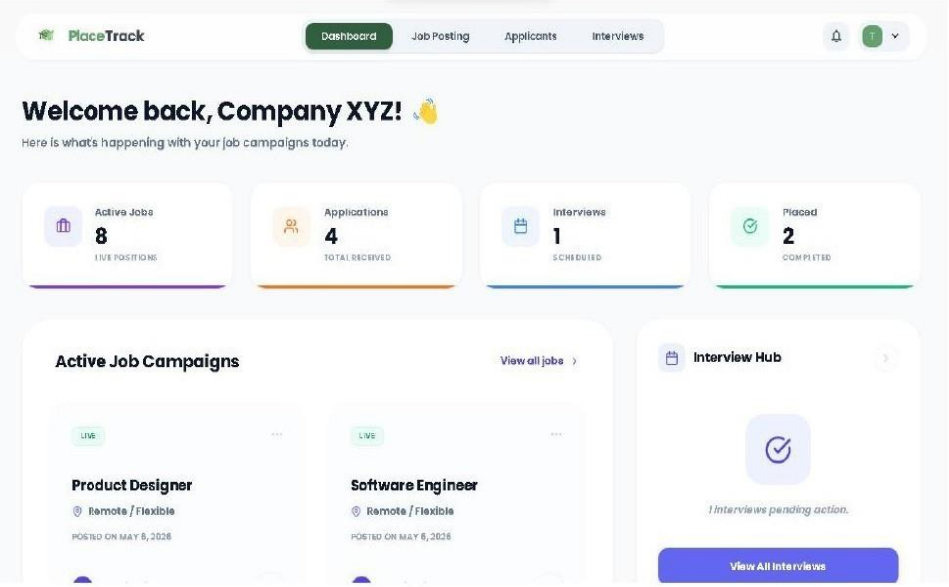


***Table3:Technologystackwithversionanddesign justification***

**DATABASEDESIGNANDROW-LEVELSECURITY**

Thedatabaseschemacomprises10coretablesorganizedaroundtheplacementlifecycle:users,profiles,companies, jobs, applications, notifications, interviews, saved_jobs, mock_tests, and resources. Key architectural decisions include:

- UUIDprimarykeysacrossalltablesensuregloballyuniqueidentifierssuitablefordistributeddeployment and resist sequential enumeration attacks.
- JSONBcolumnsintheprofilestable(experience,target_roles,preferred_locations)provideschema flexibility for semi-structured data without sacrificing relational integrity.
- Compositeuniqueconstraints(e.g.,(student_id,job_id)onsaved_jobs)enforcebusinessrulesatthe database level, preventing duplicates regardless of application behavior.
- AFTER INSERT/UPDATE triggers on the applications table automatically create recruiter and studentnotification records, ensuring event propagation is atomic with the originating data change.

The RLS policy design follows least-privilege principles: each policy grants the minimum access necessary for a role's legitimate operations. Students see only their own profiles and applications. Recruiters see only verified student profiles of their own applicants. Admins see all records. The saved_jobs table applies the strictest RLS—all operations (SELECT, INSERT, DELETE) require auth.uid() = student_id, making cross-student access structurally impossible.

## RESULTSANDDISCUSSION

### 6.13 TestingOutcomes

*Table4:Testresultssummaryacrossallthreetesting levels*

| TestingLevel | TotalCases | Passed | Partial | Pass Rate |
|---|---|---|---|---|
| UnitTesting | 48 | 48 | 0 | 100% |
| IntegrationTesting | 10 | 10 | 0 | 100% |
| SystemTesting | 8 | 7 | 1 | 87.5% |
| Overall | 66 | 65 | 1 | 98.5% |

The 100% pass rate at unit and integration levels reflects thorough test coverage, including edge cases such as empty skill arrays, null CGPA values, and duplicate application attempts—all identified and resolved during unit testing before reaching higher-level tests. The single partial result (ST-004: real-time concurrent session synchronization) is a user-experience enhancement rather than a functional or security defect: all data integrity, pipeline correctness, and notification delivery requirements were fully satisfied. The page loaded interactively in 1.8 seconds under cleared-cache conditions (measured via Lighthouse), against a 3-second non-functional requirement—a 40% performance margin.

### 6.14 SecurityAnalysis

Thedual-layersecuritymodelwasvalidatedthroughsystemtest ST-007(studentattemptingdirectURLnavigation to /admin/dashboard) and integration test IT-010 (recruiter directly querying a student-only endpoint with a valid JWT).Inbothcases,thesystempreventedaccessandexposedzerosensitivedata—thefrontendguardtriggeredan immediateredirect,andPostgRESTreturnedanemptyresultsetduetoRLSpolicyenforcement.This confirmsthat neitherafrontendbypassnoradirectAPIcallcanproduceaprivilegeescalation—bothlayersmustindependently fail for a breach to occur.

### 6.15 ImpactAssessment

ComparedtothebaselinemanualsystemPlaceTrackproducesthefollowingmeasurableimprovements:

- Recruiter self-service: All pipeline management tasks (shortlisting, scheduling, status updates) can be performed without contacting the placement office, eliminating the primary bottleneck for corporate partners.
- Notification reliability: In-app notifications with persistent database records replace WhatsApp groups, ensuring all authenticated users receive placement announcements

regardless of social media group membership or notification settings.

- Administrative efficiency: Real-time dashboards and export capabilitieseliminate the 40%+ of placement officer time previously devoted to manual data consolidation, freeing capacity for strategic placement activities.
- Data integrity: A single authoritative database replaces competing spreadsheet versions, ensuring all stakeholders operate from consistent, current data.
- Studentcareerpreparation:Integratedmocktests,codingchallenges,andcareerguidesprovidei nstitutional support for interview preparation that previously did not exist.

## CONCLUSION

ThispaperhaspresentedPlaceTrack,afull-stack,multi-role Campus Placement Management System builton React 19, PostgreSQL 15, PostgREST, and Tailwind CSS. The system addresses five compounding failure modes in traditional campus placement management—data fragmentation, notification opacity, recruiter friction, administrative blindspots, and security gaps—through a unified, database-backed, role-segregated platform.

ThearchitecturaldecisiontoenforceaccesscontrolatthePostgreSQLlayerthroughRow-LevelSecuritypolicies, ratherthanrelyingsolelyonapplication-level checks, provides security guaranteest hathold independentof applicationcodecorrectness—amaterialimprovementovercomparableexistingsystems.Formaltestingacross66 testcasesvalidatedsystemcorrectnessata98.5%passrate,withthesingleoutstandingitem(real-timeWebSocket push updates) documented and roadmapped for v1.1.0.

PlaceTrack demonstrates that a focused academic development team, applying industry-standard tools and a rigorous requirements-driven methodology, can deliver software that is technically sound, visually professional, andinstitutionallydeployable.Thesystemisreadyforproductiondeploymentatthetargetinstitution andprovides a reusable architectural template for similar Ed-Tech and HR-Tech applications in the Indian higher education context.

## FUTUREENHANCEMENTS

Thefollowingenhancementsareprioritizedforfuturedevelopmentbasedonthelimitationsidentifie dduringtesting and evaluation:

- Real-TimeWebSocketNotifications(v1.1.0):Replacethepolling-basedapproachwithSupabaseRealtime subscriptions to deliver instant updates across concurrent sessions, resolving ST-004.
- Transactional Email Integration: Integrate SendGrid or Amazon SES to dispatch email alerts for critical events, ensuring users remain informed when not actively logged in.
- AI-Powered Resume Parsing: Implement an LLM-based PDF/Word resume parser that auto-populates all resume builder form fields from an uploaded document, reducing student onboarding friction.
- Multi-Institution Support (SaaS): Redesign the data bases chema with tenant ID columns and corresponding RLSpoliciestosupportmulti-tenancy,transformingPlaceTrackintoalicensableSaaSplatformforcolleges and universities nationwide.
- AI-BasedJobMatching:Implementarecommendationenginethatsuggestsrelevantjobopeningsto students based on skill match, CGPA, department, and historical application behavior.
- NativeMobileApplications:DevelopiOS/AndroidapplicationsusingReactNativeforpushnotifications, biometric authentication, and offline access to study resources.

**REFERENCES**


1. D.Flanagan,JavaScript:TheDefinitiveGuide,7thed.Sebastopol,CA:O'ReillyMedia,2020.
2. Banks and E. Porcello, Learning React: Modern Patterns for Developing React Apps, 2nd ed. Sebastopol, CA: O'Reilly Media, 2020.
3. ReactDocumentationTeam,"React—TheLibraryforWebandNativeUserInterfaces,"2024.[Online]. Available: https://react.dev. [Accessed: May 2026].
4. ViteDocumentationTeam,"Vite—NextGenerationFrontendTooling,"2024.[Online].Available: https://vitejs.dev. [Accessed: May 2026].
5. PostgreSQL Global Development Group, "PostgreSQL 15 Documentation — Row Security Policies," 2024. [Online]. Available: https://www.postgresql.org/docs/15/ddl-rowsecurity.html. [Accessed: May 2026].
6. PostgREST Team, "PostgREST Documentation," 2024. [Online]. Available: https://postgrest.org. [Accessed: May 2026].

7. M.Fowler,PatternsofEnterpriseApplicationArchitecture.Boston,MA:Addison-WesleyProfessional,2002.
8. R. C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship. Upper Saddle River, NJ: Prentice Hall, 2008.
9. OWASPFoundation,"OWASPTopTen2023,"2023.[Online].Available:https://owasp.org/www-project-top-ten/. [Accessed: May 2026].
10. W3C Web Accessibility Initiative, "Web Content Accessibility Guidelines (WCAG) 2.1," 2018. [Online].Available: https://www.w3.org/TR/WCAG21. [Accessed: May 2026].
11. R.T.Fielding,"ArchitecturalStylesandtheDesignofNetwork-basedSoftwareArchitectures,"Ph.D. dissertation, Univ. California Irvine, 2000.
12. Sommerville,SoftwareEngineering,10thed.Harlow,England:PearsonEducation,2016.
13. Chen et al., "Row-Level Security Enforcement in PostgreSQL for Multi-Tenant Web Applications," IEEETrans. Dependable Secure Comput., vol. 18, no. 4, pp. 1520–1533, 2021.
14. TailwindCSSTeam,"TailwindCSS—AUtility-FirstCSSFramework,"2024.[Online].Available: https://tailwindcss.com. [Accessed: May 2026].
15. jsPDFContributors,"jsPDF—Client-sideJavaScriptPDFGeneration,"2024.[Online].Available: https://github.com/parallax/jsPDF. [Accessed: May 2026].
16. React Router Team, "React Router v7 Documentation," 2024. [Online]. Available: https://reactrouter.com. [Accessed: May 2026].
17. Sharma, R. & Kaur, P., "Challenges in Campus Placement Management: A Survey of Indian EngineeringColleges," Int. J. Ed. Manag. Technol., vol. 9, no. 2, pp. 45–61, 2019.
18. Patel,A.etal.,"DigitizingCampusRecruitment:AFeasibilityStudy,"J.Ed.Technol.Soc.,vol.23,no.1,pp. 88–102, 2020.
19. Gackenheimer,C.,IntroductiontoReact.NewYork,NY:Apress, 2015.
20. html2canvasContributors,"html2canvas—ScreenshotswithJavaScript,"2024.[Online].Available:https://html2canvas.hertzen.com. [Accessed: May 2026].