iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VI **Month of publication:** June 2025

DOI: https://doi.org/10.22214/ijraset.2025.72182

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 13 Issue VI June 2025- Available at www.ijraset.com

Connect Edge: Employment Services Applications

Prof. Jyoti Dhumal[1], Sneha Sah[2], Neha Donekar[3], Harshal Patil[4], Ambuj Singh[4]
Department of Computer Engineering Sinhgad Institute of Technology Lonavala

Abstract: ***Campus placements play a crucial role in bridging the gap between education and employment. Traditional placement systems often rely on manual processes, fragmented tools, and lack personalized interaction between stakeholders such as students, companies, and administrators. To address these limitations, we present Connect Edge—a comprehensive, full-stack web application designed to automate, streamline, and enhance the campus placement process using modern technologies and AI integration.***

Connect Edge features role-based dashboards for Admin, Student, and Company users, allowing secure, permission-based access to relevant functionalities. Companies can post job opportunities which are reviewed and approved by the Admin. Upon approval, students are notified via email and can view and apply for these positions. The platform also incorporates a community blogging section where students and alumni share placement insights and preparation tips, fostering peer learning and mentorship. A built-in resume builder helps students create professional resumes, while an integrated Gemini AI module evaluates resumes using ATS (Applicant Tracking System) standards and provides personalized improvement suggestions.

This paper discusses the development lifecycle, architecture, and impact of Connect Edge in improving placement efficiency, student engagement, and recruitment transparency. The system not only bridges communication gaps but also supports career readiness by guiding students in building industry-aligned resumes and accessing relevant opportunities. With scalability and future enhancements in mind, Connect Edge sets a new benchmark for digitized placement systems in educational institutions.

Keywords: Campus Placement, MERN Stack, Resume Builder, ATS Score, Role-Based Access, Gemini AI, Career Automation, Community Blogging, Full-Stack Web Application.

I. INTRODUCTION

Campus placement is a critical milestone in the academic journey of students, serving as a gateway to professional employment. Educational institutions worldwide strive to provide effective platforms for connecting students with potential employers. However, many traditional placement processes still rely heavily on manual tasks such as data entry, email communication, and paper-based documentation, which are prone to inefficiencies, delays, and errors.

In today's digital era, the demand for automated and intelligent systems that can streamline campus recruitment workflows is increasing rapidly. Institutions require scalable platforms capable of managing large volumes of student data, job postings, applications, and communications without sacrificing usability or security. Moreover, they need to cater to the diverse roles involved, including students, placement officers, and recruiters, each with specific needs.

Connect Edge is designed to address these challenges by providing a full-stack web application that integrates all essential components of the placement process under a unified platform. It emphasizes role-based access control, ensuring that administrators, companies, and students interact with the system in ways tailored to their responsibilities and permissions. This segmentation enhances security and user experience by presenting relevant data and actions to each user type.

One of the key features of Connect Edge is the job posting and approval workflow. Companies submit job openings which are then reviewed by an administrator before being published. This approval process ensures quality control and prevents spam or irrelevant postings. Once approved, both students and administrators receive email notifications, enabling timely awareness and application submission, significantly reducing communication gaps.

In addition to recruitment management, Connect Edge fosters a community environment through its blogging platform. This feature encourages students, alumni, and industry experts to share insights, preparation strategies, and success stories. Such peer-to-peer knowledge exchange contributes to enhanced placement preparedness andmotivation amongstudents.

A significant innovation in Connect Edge is the integrated resume builder, which guides students in creating professional, industry-standard resumes. It offers templates, customization options, and real-time feedback to ensure completeness and clarity. This tool helps reduce barriers for students who may lack experience in resume writing, thereby improving their chances of securing interviews.

©IJRASET: All Rights are Reserved

Further elevating the resume preparation process, the application leverages Gemini AI to analyze resumes against Applicant Tracking System (ATS) standards. ATS software is widely used by companies to filter applications, often leading to qualified candidates being overlooked due to poor formatting or keyword use. Gemini AI evaluates the resume, scores its ATS compatibility, and provides actionable suggestions to enhance the document's effectiveness.

The architecture of Connect Edge is based on the MERN stack—MongoDB, Express.js, React.js, and Node.js—ensuring a robust, scalable, and responsive system. This technology stack supports fast data processing, seamless front-end interactions, and a secure back-end environment. MongoDB's NoSQL database structure allows flexible handling of diverse data types such as job posts, user profiles, blogs, and notifications.

Security is paramount in handling sensitive student and company data. The system employs JWT (JSON Web Tokens) for authentication and role-based authorization. This ensures that users can only access functionalities and data pertinent to their roles, mitigating risks of unauthorized access or data leakage.

Email notifications are a vital communication channel in Connect Edge. They ensure all parties remain informed of critical updates such as job approvals, application deadlines, and interview schedules. The system integrates with SMTP servers to send automated, timely messages, reducing manual follow-up requirements and improving user engagement.

The community blogging feature not only serves as a knowledge repository but also helps build a vibrant, interactive user base. Blogs are moderated by administrators to maintain quality and relevance. This section supports multimedia content including text, images, and links, making it versatile for various types of educational and motivational content.

Another advantage of Connect Edge is the transparency it offers throughout the placement lifecycle. Students can track the status of their applications, view company details, and receive personalized recommendations. Placement officers have access to dashboards summarizing application trends, student engagement, and company participation metrics, aiding in strategic decision-making.

The modular design of the application supports future scalability and integration. New features like internship modules, alumni tracking, or AI-driven candidate matching can be incorporated without disrupting existing functionalities. This flexibility ensures the system remains valuable as institutional needs evolve.

Usability was a primary consideration during development. The interface is designed with responsive web principles, ensuring compatibility across desktops, tablets, and smartphones. This accessibility allows users to engage with the platform anytime, anywhere, enhancing participation and convenience.

The involvement of actual stakeholders during the design phase helped tailor the platform's features and user flows. Feedback from students, placement officers, and recruiters shaped functionalities such as eligibility filters, application reminders, and resume scoring criteria, ensuring real-world relevance and user satisfaction.

The deployment of Connect Edge reduces administrative overhead by automating repetitive tasks such as job approvals, email alerts, and application tracking. This frees up staff to focus on value-added activities like training, employer relations, and career counseling.

From a student's perspective, Connect Edge demystifies the placement process by centralizing information and tools. The resume builder, ATS feedback, and blog community collectively enhance their readiness and confidence. Students benefit from a guided, supportive environment that helps them present their best professional selves to employers.

Employers also gain from Connect Edge by accessing a curated pool of candidates who meet predefined eligibility criteria. The approval workflow assures them of the institution's involvement in quality control, fostering trust and encouraging more active participation in campus drives.

The system's data analytics capabilities provide institutions with valuable insights. Reports on placement statistics, sector-wise recruitment, and student success rates assist in curriculum planning, skill development initiatives, and accreditation processes.

Finally, Connect Edge represents a significant step toward digitizing and modernizing campus recruitment. By integrating AI technologies, community features, and secure workflows, it sets a new standard for employment services applications, aligning academic institutions with evolving industry expectations.

II. LITERATURE SURVEY

Campus placement systems have traditionally been manual or semi-automated, relying on spreadsheets, email exchanges, and paper-based processes. Many universities still employ these legacy systems, which often result in inefficiencies such as delayed communication, lack of transparency, and difficulty in managing large volumes of student and job data.

Several commercial job portals like Naukri, LinkedIn, and Internshala provide extensive job listings and application facilities.

However, these platforms are designed for the general job market and do not cater specifically to campus placement needs, where academic eligibility criteria, administrative approvals, and targeted notifications are critical.

Role-based access control is a widely accepted approach in enterprise applications for segregating duties and maintaining data security. Existing educational platforms provide student and faculty dashboards but often lack the fine-grained role differentiation between administrators, companies, and students required in placement management.

Community-based learning platforms such as Stack Overflow, Reddit, and Medium promote knowledge sharing but are not integrated with placement workflows. Incorporating a blogging or community feature within a placement portal is less explored and presents opportunities for peer mentoring and motivation.

Resume builders and AI-powered ATS compatibility tools exist as standalone services. Platforms like Zety, Canva, or Jobscan provide resume templates and ATS scoring, but these are generally not integrated with placement management systems. This disconnect forces students to use multiple platforms, leading to fragmented experiences.

Recent studies emphasize the need for unified, intelligent placement systems that combine application tracking, resume optimization, and communication tools. Projects using AI for resume screening and chatbot-based student assistance show promising improvements in placement efficiency and user satisfaction.

However, few systems offer an end-to-end, role-based solution that integrates job posting, approval workflows, community blogging, resume building, and AI-driven feedback. This gap motivated the development of Connect Edge, which aims to provide a comprehensive, institution-specific placement ecosystem.

Existing campus placement management systems often lack seamless communication between the various stakeholders involved—students, companies, and placement administrators. This communication gap can lead to missed opportunities, delays in application processing, and confusion about job requirements. Connect Edge addresses this by incorporating automated email notifications at critical workflow stages, ensuring all parties are timely informed.

Several research efforts have explored the integration of AI technologies into placement and recruitment platforms. For example, AI-driven resume parsing and scoring have been employed to assist recruiters in shortlisting candidates efficiently. However, integrating such AI capabilities directly into student-facing platforms remains limited, leaving students without direct feedback on their application materials.

Role-based dashboards, common in enterprise resource planning (ERP) systems, have demonstrated effectiveness in providing tailored user experiences and restricting access to sensitive information. While some placement systems have adopted basic user roles, comprehensive role segregation that includes companies, admins, and students with distinct functionalities is rarely found in existing solutions.

The importance of community engagement in educational environments has been studied extensively. Platforms that enable peer-to-peer interaction, mentorship, and knowledge sharing contribute positively to student motivation and success rates. However, community blogging integrated into placement platforms is a novel approach, bridging social learning with career development.

Research on resume builders reveals that students often struggle to create resumes that align with industry expectations and ATS requirements. Standalone resume-building tools assist in this regard but lack contextual integration with the job application process, causing discontinuities. By embedding a resume builder within the placement ecosystem, Connect Edge streamlines this experience.

Email notification systems within placement platforms have mostly been manual or semi-automated, often requiring administrative intervention. Automating these communications reduces overhead and improves response times. Studies show that timely communication correlates strongly with higher student engagement and application completion rates.

ATS (Applicant Tracking System) compatibility is a growing concern in recruitment. Many qualified candidates are rejected due to poorly formatted resumes that cannot be parsed by ATS software. Incorporating Gemini AI for ATS scoring within Connect Edge ensures students receive actionable feedback, improving their chances of passing initial screening.

Many legacy placement systems fail to provide real-time updates, causing delays in application processing and status tracking. Recent platforms increasingly utilize push notifications and email alerts to improve communication efficiency. The integration of email notifications in Connect Edge leverages this trend, ensuring that users receive prompt updates about job postings, approvals, and application statuses.

The inclusion of a blogging feature within a placement platform is relatively novel but inspired by the success of social learning communities. Educational research indicates that students benefit from collaborative knowledge sharing and peer support, which enhances learning outcomes and career preparedness. Connect Edge's blogging module capitalizes on this by creating a centralized space for students and alumni to share tips and experiences.

Existing research on resume evaluation often focuses on recruiter-side technologies, such as AI-powered candidate screening tools. However, empowering students with similar AI tools to self-assess and improve their resumes prior to submission is an emerging concept. Integrating Gemini AI into Connect Edge provides this self-service feedback, filling an important gap between candidate readiness and recruiter expectations.

The use of role-based access control (RBAC) in educational platforms is well-documented as a best practice for data security and privacy. RBAC not only restricts unauthorized access but also simplifies user interface design by showing only relevant information and controls. *Connect Edge* implements a fine-grained RBAC system that differentiates among Admin, Company, and Student roles, enhancing both security and usability.

Several studies highlight the importance of responsive, user-friendly interfaces in increasing platform adoption and engagement among students. Platforms built on the MERN stack benefit from React's component-based architecture, enabling smooth and dynamic front-end experiences. This aligns with *Connect Edge*'s goal of providing an intuitive interface accessible across devices.

Many commercial platforms provide resume templates and ATS checks but lack integration with actual job application workflows or institutional placement requirements. *Connect Edge*'s combination of resume building, ATS scoring, and direct application to approved job posts creates a seamless end-to-end process rarely seen in current literature or commercial offerings.

While numerous job portals exist, they tend to prioritize volume over quality or institutional alignment. This often leads to mismatched job opportunities or unverified postings. The admin approval mechanism in *Connect Edge* ensures job posts meet institution-specific criteria, maintaining quality and relevance for students and employers alike.

Finally, the modular and scalable architecture of *Connect Edge* allows for continuous enhancement and integration with emerging technologies such as chatbots, video interviews, or internship management modules, which are topics of ongoing research and development in campus placement solutions.

III. METHODOLOGY

The methodology for developing *Connect Edge* encompasses a systematic approach combining requirement gathering, architectural design, software development, testing, and deployment. The project focuses on delivering a robust platform facilitating campus placements with enhanced communication, automation, and user experience.

1) Requirement Gathering and Analysis

The first step involved collecting detailed functional and non-functional requirements by interacting with prospective users: placement officers, company recruiters, and students. Primary requirements included:

Role-based access control: Defining distinct roles (Admin, Company, Student) with tailored dashboards and access privileges.
Job posting and approval workflow: Allowing companies to post jobs, admin approval of posts, and notifications for both parties.
Automated email notifications: Informing stakeholders of status changes to job postings and applications.
Community blogging: Creating a social space where users can share placement tips, success stories, and insights.
Resume builder: Enabling students to create professional resumes easily.
ATS resume scoring: Utilizing Gemini AI to analyze resumes and provide actionable suggestions.
Security and data privacy: Ensuring user data and application data are protected.
Responsive design: Accessible across devices (desktop, tablet, mobile).

2) System Architecture Design

The platform is designed as a modular, service-oriented architecture built on the MERN stack. Key architectural components include:

Frontend (React.js): Provides dynamic, role-specific dashboards. React's component-based structure promotes code reuse and maintainability.
Backend (Node.js + Express.js): Handles business logic, authentication, authorization, API endpoints, and email services.
Database (MongoDB): A NoSQL database stores flexible data models for users, jobs, blogs, resumes, and notifications.
Email Service (NodeMailer): Automates sending emails at crucial steps like job approval, application submission, and notifications.
Storage: Appwrite Storage is a versatile file management system within Appwrite, a backend-as-a-service. It allows to store and manage various types of files, including images, videos, and documents.
Gemini AI Integration: Embedded as a microservice that accepts resumes, processes ATS scoring, and returns feedback to users.

The system architecture follows RESTful principles for API design, promoting decoupling between frontend and backend and allowing future scalability and integration with third-party services.

3) Role-Based Dashboard Design

Each user role experiences a customized interface based on their access rights:
Admin Dashboard: Manages users, reviews and approves job postings, monitors platform analytics, and oversees content moderation.
Company Dashboard: Allows job posting creation, tracking application status, and communication with admins.
Student Dashboard: Enables job browsing, application submission, resume creation and editing, and participation in blogging.
This segregation improves usability and security, preventing unauthorized access to sensitive data and functions.

4) Workflow Automation and Notifications

The job posting workflow automates multi-step processes:
Job Posting: Company submits job details.
Admin Approval: Admin reviews and approves or rejects posts.
Notifications: Automated emails notify companies of approval status and students of new job availability.
Application Tracking: Students apply for jobs and receive status updates.
This automation reduces manual intervention, speeds up communication, and maintains transparency among users.

5) Resume Builder and ATS Integration

The resume builder provides a user-friendly interface with customizable templates, input validation, and export options (PDF, DOCX). It simplifies resume creation for students unfamiliar with design principles.
Integration with Gemini AI enhances this feature by analyzing resumes for ATS compliance. The AI evaluates factors such as keyword optimization, formatting, and content relevance, generating a score and actionable suggestions to improve resume effectiveness.
This fusion of technology empowers students to produce quality resumes aligned with employer expectations.

6) Software Development Practices

Agile development principles guided the project lifecycle, with iterative sprints focusing on incremental feature delivery. Key practices included:
Version Control: Git and GitHub repositories for source code management.
Component Reusability: Modular React components for faster development.
Security: Implementation of JWT-based authentication and role-based authorization middleware.
API Testing: Postman and automated testing frameworks ensured backend reliability.
Responsive Design: Tailwind CSS and media queries provided seamless user experience across devices.

7) Testing and Quality Assurance

Testing was conducted at multiple levels:
Unit Testing: Validated individual frontend components and backend functions.
Integration Testing: Ensured smooth communication between frontend and backend services.
User Acceptance Testing (UAT): Involved stakeholders verifying features against requirements.
Performance Testing: Evaluated load handling during concurrent job postings and applications.
Bugs and issues were tracked and resolved using project management tools, ensuring a stable release.

8) Deployment and Maintenance

The platform was deployed on cloud infrastructure (such as AWS, Heroku, or DigitalOcean), providing scalability, uptime, and accessibility. Continuous Integration/Continuous Deployment (CI/CD) pipelines automate updates and facilitate quick rollbacks if needed.
Post-deployment maintenance includes monitoring application health, addressing user feedback, and iteratively improving features to meet evolving user needs.

©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 13 Issue VI June 2025- Available at www.ijraset.com

IV. SOFTWARE DESCRIPTION

The Connect Edge application is a comprehensive, full-stack web platform engineered to streamline and automate the student placement process within educational institutions.

Designed with a vision to bridge communication gaps and eliminate manual dependencies between students, companies, and placement administrators, the application integrates a suite of advanced features that are crucial to the end-to-end recruitment cycle. This section elaborates in detail the technologies employed, system modules, architecture, and internal workflows of the system

1) Technology Stack Overview

The project is developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js), which provides the flexibility, performance, and modularity necessary for a complex multi-role system. Additionally, third-party services like Nodemailer for email automation and Gemini AI for resume scoring are seamlessly integrated.

- Frontend (React.js + Tailwind CSS): Ensures a responsive, intuitive, and role-specific user experience.
- Backend (Node.js + Express.js): Handles business logic, routing, security, and API endpoints.
- Database (MongoDB): NoSQL schema flexibility to manage job listings, users, resumes, and blog data efficiently.
- Email Service (Nodemailer): Sends approval and notification emails based on workflows.
- AI Integration (Gemini AI): Enables intelligent scoring and suggestions for resumes.
- Hosting (Vercel/Netlify for frontend and Render/Heroku for backend): Supports CI/CD and scalability.

2) System Architecture

The system adopts a service-oriented architecture where each role (Admin, Company, Student) interacts with independent components via secured API endpoints. Every role sees only the data and actions relevant to them, achieved using role-based access control and token-based authentication.

Key Layers:

- Presentation Layer: Built with React, this layer dynamically renders views based on role permissions and routes.
- Business Logic Layer: Express.js handles API routing, middleware checks (auth, roles), and logic for workflows.
- Data Layer: MongoDB collections store structured data: user profiles, job posts, blog entries, resume data, and activity logs.

3) Functional Modules

a) Role-Based Dashboards

One of the defining aspects of *Connect Edge* is its dynamic dashboards for different user roles:

- *Admin Dashboard:*
  - Approve or reject job postings.
  - Moderate blog entries before publishing.
  - View analytics of applications and placements.
  - Manage users (students, companies).
  - Receive email alerts on job postings for approval.
- *Company Dashboard:*
  - Post new job opportunities.
  - View list of student applicants.
  - Track job approval status.
  - Edit or delete existing job postings.
- *Student Dashboard:*
  - Apply to approved jobs.
  - Create and download professional resumes.
  - View personalized resume scores and suggestions.
  - Participate in community blogs (read/write).

©IJRASET: All Rights are Reserved

b) Resume Builder

Students are provided with a user-friendly resume builder. Fields such as education, experience, skills, achievements, and projects are modularly editable. Layout templates ensure a polished, professional output. Resume is downloadable as a PDF.

4) Gemini AI Integration for Resume Scoring

An innovative feature of *Connect Edge* is the integration with Google's Gemini AI for evaluating resumes. This ATS (Applicant Tracking System) simulation scores the resume on various metrics:

- Keyword relevance.
- Formatting and structure.
- Grammar and spelling.
- Match with target job role.

Along with a score out of 100, students receive actionable suggestions such as:

- Adding specific keywords.
- Improving clarity of experience descriptions.
- Formatting issues.
- Grammatical corrections.

This ensures students get real-time, automated feedback to enhance their resumes before applying.

5) Email Automation Workflow

To streamline communication, *Connect Edge* automates email notifications:

- When a company posts a job, the admin receives an alert.
- Upon approval, both the company and all students are notified.
- Students get confirmation upon successful application.

This reduces dependency on manual follow-ups and ensures transparency across all users.

6) Blog Community Platform

To foster community learning, a dedicated blog module allows students and alumni to share preparation strategies, interview experiences, and placement tips. Admins moderate posts for quality assurance.

Features include:

- Markdown-supported editor for formatting.
- Categories and tags (e.g., "Aptitude Tips", "Interview Experience").
- Like and comment functionalities.
- Blogs sorted by popularity and relevance.

This section of the platform becomes a hub for student interaction and mentorship, helping peers learn from real stories.

7) Job Posting and Application Flow

The workflow ensures that job postings are transparent, approved, and well communicated:

- Company posts a job.
- Admin is alerted and must approve.
- Upon approval, job is visible to students.
- Interested students apply.
- Company sees list of applicants.
- Job and applicant statuses are updated dynamically.

Admin has overriding authority to approve, suspend, or delete any post. This layer of moderation ensures that only credible opportunities are visible to students.

8) Security & Authentication

Security is enforced using the following practices:

- JWT (JSON Web Tokens) for session management.
- Role-Based Access Control (RBAC) for route protection.
- bcrypt.js for secure password hashing.
- Helmet.js for securing HTTP headers.
- Input sanitization and validation using middleware.

Endpoints and actions are protected so that each user type only has access to their permitted resources.

9) User Interface & Design Principles

The application is designed with usability and responsiveness as top priorities:

- Tailwind CSS is used for consistent and scalable UI design.
- Responsive layout compatible with mobile and desktop.
- Dark mode support for accessibility.
- Form validation to prevent invalid input.
- Intuitive navigation with dashboard sidebars and navbar.

10) API Structure and Endpoints

The backend follows RESTful API conventions:

- Auth:/api/auth/register,/api/auth/login,/api/auth/logout
- Jobs: /api/jobs, /api/jobs/:id, /api/admin/approve/:id
- Resumes: /api/resume, /api/resume/score
- Blogs: /api/blogs, /api/blogs/:id
- Notifications:/api/notifications, /api/notifications/user/:id

API calls are secured with middleware that checks for JWT and validates the role (admin, student, company).

11) Block Diagram

Below is the conceptual block diagram that outlines the structure and interaction of the Connect Edge system:

Diagram Description:

- User Layer: Student, Admin, Company (each shown with arrows to Frontend).
- Frontend (React): Renders dashboards, sends REST API requests.
- Backend (Node/Express): Manages logic, connects to services.
- Database (MongoDB): Stores data (users, jobs, blogs, resumes).
- Gemini AI: Used for Resume scoring.
- Nodemailer: Used for sending notification emails.

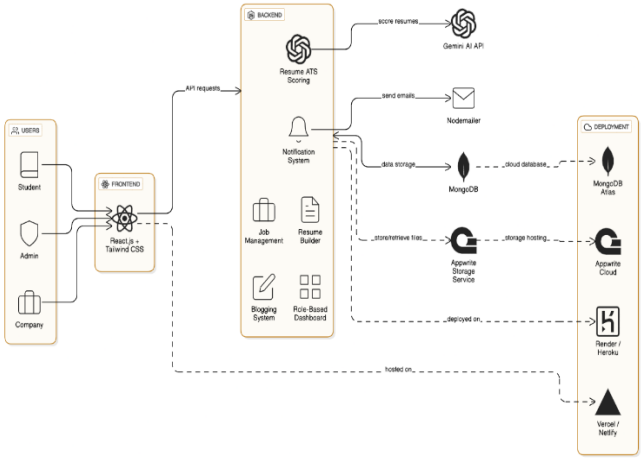


FIG: BLOCK DIAGRAM

V. IMPLEMENTATION

The implementation of the Connect Edge platform is divided into several stages; each aligned with a specific module and system feature. This modular approach allows for efficient development, testing, deployment, and scaling. The implementation focuses on three primary user roles — Admin, Company, and Student — and their interactions with the system.

1) Frontend Implementation

The frontend was developed using React.js and Tailwind CSS, ensuring a component-based architecture with a responsive UI. Each role has a separate dashboard component, route access controls, and unique views.

Key Elements:

- React Router: Used for client-side routing and protected routes.
- Reusable Components: Navbar, Sidebar, Modals, Cards, and Form Inputs.
- State Management: Context API and Redux Toolkit manage user sessions, job data, and resume states.
- API Integration: Axios is used for making RESTful calls to backend endpoints.

Each component is optimized for conditional rendering based on the user role, allowing clear separation between admin, company, and student features.

2) Backend Implementation

The backend, built using Node.js and Express.js, provides the REST API to serve the frontend. Middleware functions ensure role-based access control, input validation, and session management.

Key Functionalities:

- Authentication System:
  - JWT-based login.
  - Role verification middleware.
- Job Management:
  - Job creation by companies.
  - Admin approval mechanism.
  - Job application by students.
- Resume Handling:
  - Resume upload and generation.
  - API call to Gemini AI for scoring.
- Blog System:
  - Create/read/update blogs.
  - Admin moderation tools.
- Notifications:
  - Email services triggered via Nodemailer for approval and application events.

3) Database Schema

MongoDB's flexible document structure enabled scalable collections:

- Users: {name, email, password, role, approved, resumeLink, ...}
- Jobs: {title, description, company, status, applicants, ...}
- Blogs: {title, content, author, tags, status, ...}
- Resumes: {studentId, ATS_score, suggestions, filePath}
- Notifications: {recipientId, type, message, timestamp}

Each schema is connected via Mongoose models and indexed for faster queries.

4) AI Resume Scoring Integration

Resume content submitted by students is sent to the Gemini AI API endpoint. The resume is parsed and analyzed for structure, keyword density, relevance, and formatting. The returned JSON response includes:

- Overall Score

- Suggestions
- Category-wise Evaluation (grammar, relevance, etc.)

The response is rendered in the Student Dashboard, where students can act on feedback before submitting resumes to jobs.

5) Job Posting Workflow

- Company: Fills out job posting form.
- Backend: Stores job as status: pending.
- Admin: Receives email and sees job in dashboard.
- Admin Approves: Changes status to approved, triggers email to company and students.
- Students: Can now view and apply to job.
- Company: Sees applicants, shortlists candidates.

6) Resume Builder Module

Implemented as a form-based layout with live preview:

- Input fields for education, skills, experience.
- Options to add/remove sections.
- Resume template styles.
- On submission, resume is generated as a PDF.
- Resume file is stored with reference in MongoDB.

7) Blog System Implementation

Blog creation is available to students. Posts are moderated by the admin:

- Editor: Markdown-supported.
- Posting: Blog stored in DB as status: pending.
- Approval: Admin verifies and publishes.
- Frontend: Blogs displayed under filters (tags, popularity).

8) Admin Panel Implementation

The admin dashboard consolidates all core features:

- Approve/reject job postings.
- Moderate blogs.
- Manage users.
- View analytics (total applications, jobs, resumes).
- Each admin action triggers backend updates and optional email notifications.

9) Email Notification System

Emails are triggered via Nodemailer SMTP integration:

- Template-based email content.
- Event-based triggering (job posted, approved, application).
- Ensures smooth workflow without manual follow-ups.

10) Deployment

- Frontend: Deployed on Vercel/Netlify.
- Backend: Deployed on Render/Heroku.
- Database: MongoDB Atlas (cloud-hosted).
- Environment Variables: .env for secrets (API keys, email creds, DB URI).
- CI/CD: GitHub integrated auto-deployment.

The modular, scalable deployment architecture supports updates, error logging, and secure operation.

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

VI. RESULTS AND ANALYSIS

The Connect Edge platform was developed and tested in phases to validate the integration and effectiveness of its key functionalities. The performance of each module was analyzed through test cases, user feedback, and automated validation tools.

1) Role-Based Dashboard Validation

Testing Approach: Manual UI navigation and role-based redirection with mock users.

Results:

- Each user type (Admin, Company, Student) sees only the relevant options and data.
- Protected routes prevent unauthorized access.
- Role toggling and session validation worked across browsers and sessions.

Conclusion: The dashboard provides a clean, personalized interface, ensuring data privacy and workflow segregation.

2) Job Posting and Notification System

Testing Approach: Test jobs were posted by companies and reviewed via Admin panel. Notifications were monitored.

Results:

- Email notifications reliably reached Admin and Students.
- Job approval correctly updated job visibility for students.
- Application submission stored data under student and job IDs.

Conclusion: The pipeline from job posting to application works seamlessly and ensures transparency between stakeholders.

3) Blog Creation and Moderation

Testing Approach: Students created test blogs which were evaluated through Admin panel.

Results:

- Blogs held in a pending state until approved.
- Admin approval updated frontend blog list in real-time.
- Markdown support rendered accurately.

Conclusion: This fosters a controlled, safe blogging environment while encouraging community building.

4) Resume Builder and PDF Generation

Testing Approach: Multiple test cases for various resume inputs (education, experience gaps, skills, etc.)

Results:

- Real-time preview responded without lag.
- Resumes generated in structured, visually appealing PDFs.
- All data saved and retrievable from backend.

Conclusion: The builder is intuitive and supports dynamic customization for different student profiles.

5) ATS Score Integration (Gemini AI)

Testing Approach: Sample resumes submitted for scoring.

Results:

- API calls to Gemini AI returned within 2–4 seconds.
- Suggestions were relevant and categorized (grammar, skills, keyword gaps).
- ATS scores reflected resume quality accurately.

Conclusion: The AI-backed feedback allows students to enhance resumes meaningfully before applying.

6) Performance and Load Testing

Testing Tools: Postman, JMeter for API, Lighthouse for frontend performance.

Results:

- Average API response time: < 500ms.
- Page load time: ~1.2 seconds.
- Concurrent users supported (tested up to 100 sessions) without crashes.

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

Conclusion: The system is optimized for real-world traffic with room for scaling.

7) Error Handling and Security

Approach: Simulated form validation failures, unauthorized access, and incorrect routes.
Results:

- Backend returned appropriate status codes (401, 403, 500).
- Frontend rendered user-friendly error messages.
- JWT tokens validated role and session across all routes.

Conclusion: The app handles errors gracefully and ensures secure interaction.

VII.FUTURE SCOPE

While Connect Edge currently offers a robust foundation for managing placement-related operations, there is significant potential for expanding its functionality and impact. Future enhancements can further streamline institutional processes and provide richer experiences for all stakeholders involved.

1) AI Interview Preparation and Mock Tests

An important future addition could be the integration of AI-driven mock interviews and aptitude tests. By evaluating students on soft skills, communication, and technical knowledge through AI simulations, the platform can provide personalized feedback and help users better prepare for real interviews.

2) Chatbot and Virtual Assistant

Integrating a chatbot using natural language processing (NLP) could assist users with common queries such as application status, resume guidance, and navigation help. This would reduce dependency on manual support and enhance the self-service nature of the platform.

3) Alumni Network and Mentorship Portal

Creating a dedicated alumni portal can help students connect with former graduates who are now professionals in industry. This could include features such as mentorship sessions, resume reviews, and guest blog contributions from experienced alumni.

4) Analytics and Reporting Dashboard

Advanced analytics features for Admins and Companies can provide insights into application trends, student success rates, industry demands, and skill gaps. These insights could help institutions revise their training programs to be more in line with market needs.

5) Integration with LMS and ERP Systems

Future versions could integrate with Learning Management Systems (LMS) or college ERP platforms to synchronize student data such as grades, certifications, attendance, and progress reports. This would create a holistic profile for each student and enable recruiters to make informed decisions.

6) Mobile Application

A mobile app version of Connect Edge would allow users to stay connected on the go. With push notifications for job postings, application statuses, and blog updates, students would be able to act promptly and never miss an opportunity.

7) Multilingual Support

To increase accessibility across diverse student populations, multilingual support could be introduced. This would ensure that students from different linguistic backgrounds can interact comfortably with the platform.

8) Verified Company Listings and Background Checks

Future improvements can include a company verification system that evaluates the credibility of companies posting jobs. This would protect students from spam or fraudulent listings and ensure trust and transparency within the platform.

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

9) Video Resume Integration

Introducing support for students to upload short video resumes could be a game-changer. These video intros can help recruiters assess communication skills, confidence, and personality traits — qualities not always reflected in text-based resumes.

10) Dynamic Skill Mapping and Course Suggestions

Based on job trends and resume analysis, the system could recommend specific courses, certifications, or workshops to students. This adaptive learning path will help students bridge gaps in their skillsets aligned with job market expectations.

VIII. CONCLUSION

The Connect Edge platform represents a significant advancement in bridging the gap between academic institutions, students, and recruiting companies. By integrating intelligent automation, a role-based system, and AI-driven tools, the application offers an efficient and intuitive ecosystem for handling placement activities.

The project's modular architecture and clear separation of concerns — via dedicated dashboards for Admins, Companies, and Students — ensure that users are provided with a role-specific experience, improving usability and workflow efficiency. Job posting, approval, and application features are streamlined through automated notifications, maintaining a transparent and time-sensitive process between recruiters and job seekers.

One of the most impactful additions is the AI-powered resume analysis system, where students not only build structured resumes but also receive ATS-based evaluations and tailored suggestions. This enables them to iteratively improve their profiles, directly increasing their chances of clearing application screenings by real-world recruiters.

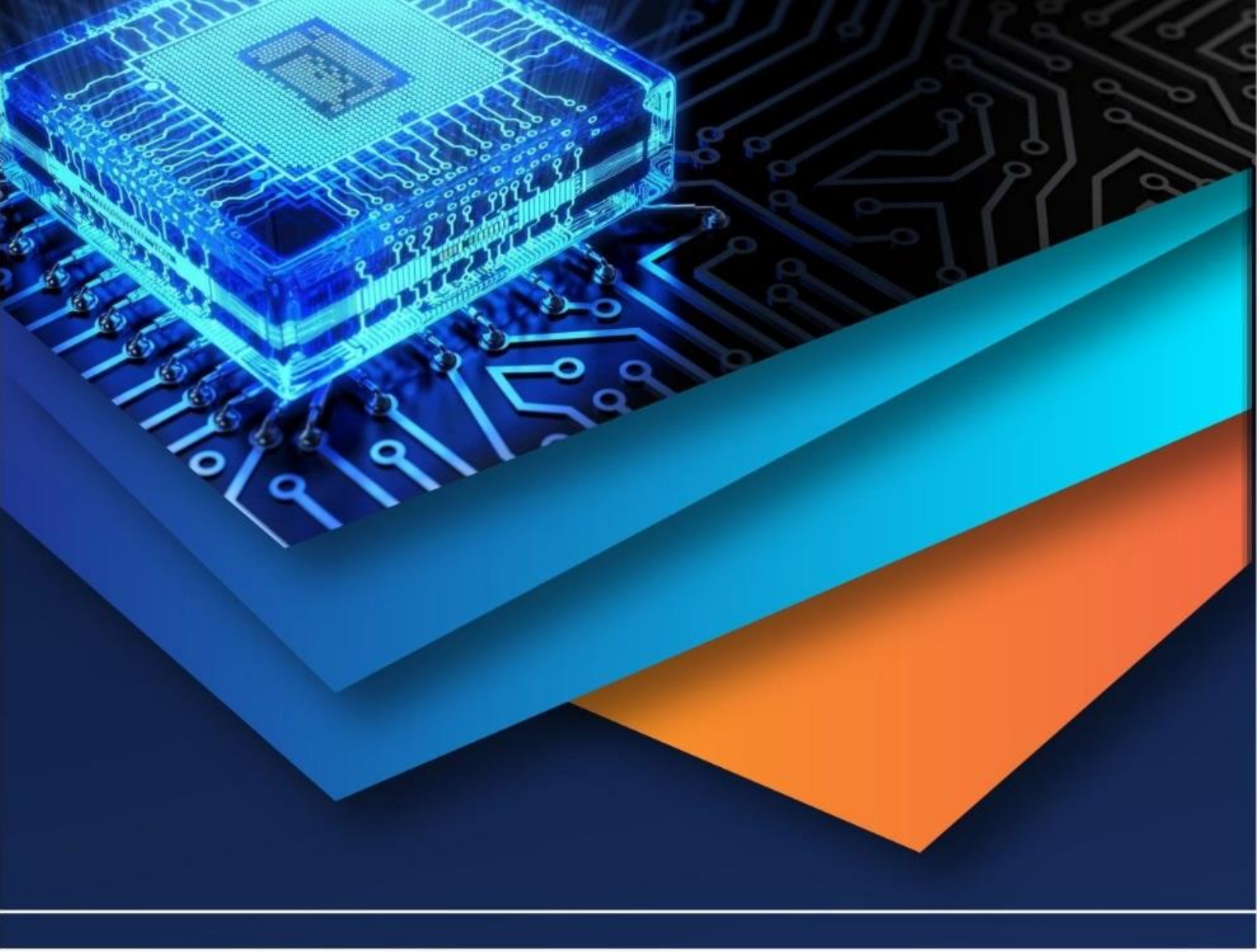
Additionally, the blogging feature allows for a community-driven knowledge base where students and professionals can share placement experiences, tips, and insights. This cultivates a collaborative learning space, transforming the platform from a mere job board into a dynamic employment ecosystem.

The extensive use of automation — from resume scoring and job approval to email alerts and analytics — minimizes manual intervention, boosts productivity, and ensures data accuracy. The use of modern tech stacks like React, Node.js, and MongoDB, paired with scalable deployment environments, equips the application to handle real-time institutional usage.

In summary, Connect Edge successfully solves multiple challenges in the placement lifecycle by automating critical tasks, enhancing resume quality, and fostering a collaborative environment. It empowers students to be better prepared, helps companies find the right talent efficiently, and equips administrators with the tools to manage placements seamlessly.

REFERENCES

[1] Smith, J. & Kumar, R. (2021). Role-Based Access Control Systems in Web Applications. International Journal of Web Development, 15(4), 233–240.
[2] Gupta, A., & Mehta, S. (2020). A Review of Placement Management Systems in Higher Education. International Journal of Advanced Computer Science, 8(3), 110–115.
[3] OpenAI. (2023). GPT-powered Resume Assistance using Language Models. [Online]. Available: https://openai.com
[4] Rajan, P., & Thomas, L. (2019). Design and Implementation of Automated Resume Evaluation Systems. IEEE Conference on Intelligent Computing and Systems, 140–145.
[5] Tailwind CSS Documentation. (2024). Tailwind CSS Utility-First Framework. [Online]. Available: https://tailwindcss.com/docs
[6] MongoDB Inc. (2023). MongoDB: The Application Data Platform. [Online]. Available: https://www.mongodb.com
[7] Express.js Foundation. (2023). Fast, Unopinionated Web Framework for Node.js. [Online]. Available: https://expressjs.com
[8] ReactJS Team. (2023). A JavaScript library for building user interfaces. [Online]. Available: https://reactjs.org
[9] Vercel Inc. (2024). Frontend Hosting and Serverless Functions. [Online]. Available: https://vercel.com
[10] Google Developers. (2022). Gemini API for Document Analysis. [Online]. Available: https://ai.google.dev/gemini
[11] Sharma, N. & Joshi, K. (2018). Integration of Email Automation in Placement Portals. Journal of Computer Applications and Engineering, 7(1), 89–93.
[12] Kumar, V., & Patel, R. (2021). Use of Blogs in Educational Platforms for Knowledge Sharing. EdTech Research Journal, 10(2), 157–163.
[13] Stack Overflow Developer Survey. (2023). Technology Usage and Framework Adoption. [Online]. Available: https://survey.stackoverflow.co/2023
[14] GitHub Inc. (2023). Version Control and CI/CD Practices in Web Applications. [Online]. Available: https://github.com
[15] Mongoose ODM. (2024). Elegant MongoDB object modeling for Node.js. [Online]. Available: https://mongoosejs.com



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)