International Journal of Computer Techniques–IJCT Volume 13 Issue 3, May-June - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

Workbridge: A Smart Web-Based Platform for Connecting Job Seekers and Employers

Gaurav Patil*, Jaydeep Bhoite*, Vinali Ahire*, Mannat Bagga*, Sumit Sonawane*, Prof. Chadchankar Amarnath*

(Department of Computer Science Engineering, School of Technology and Research, DPGU, Pune)

Abstract: *Finding a job as a fresh graduate has never been easy — and the tools available today make it harder than it needs to be. Most job portals are built with experienced professionals in mind, leaving students and new graduates navigating platforms that were never really designed for them. Workbridge changes that. It is a web-based job and internship portal built specifically to connect first-time job seekers with employers in a structured, transparent, and easy-to-use environment. The platform gives students role-specific dashboards, resume upload, and real-time application tracking, while employers get a clean interface to post openings and shortlist candidates without sifting through irrelevant applications. Under the hood, Workbridge runs on Python (Django), HTML/CSS/JavaScript, and SQLite/MySQL, using a 3-tier architecture that keeps things fast, secure, and scalable. Every core module — from login and profile creation to job search and candidate management — has been tested and works as intended. This paper walks through how Workbridge was designed, built, and evaluated, and why it represents a practical step toward closing the employment gap for fresh graduates.*

Keywords — Job Portal, Fresher Employment, Django, Web Application, Application Tracking, Employer Dashboard, Recruitment Platform.

I. INTRODUCTION

Every year, thousands of graduates across India step out of college with degrees in hand and no clear path forward. The job market exists — opportunities exist — but the tools to connect talent with employers are surprisingly broken. Most platforms were built for professionals who already have years of experience, which means fresh graduates are trying to compete in a space that was not designed with them in mind [1].

The old ways of job hunting — walking in with a printed resume, scanning newspaper listings, relying on a relative's contact — simply do not work anymore. Digital platforms like Naukri, LinkedIn, and Indeed have stepped in to fill the gap, but only partially. These portals are optimized for filtered, experienced talent. For a student who has just finished their degree, navigating these platforms can feel overwhelming: there is little guidance on building a profile, no real way to track what happened to an application, and no feedback loop at all [2].

Employers, especially smaller organizations, face the other side of the same problem. A single job post can attract hundreds of applications, most of which are not even remotely relevant. Without proper filtering tools, shortlisting becomes a manual, time-consuming process that pulls hiring managers away from more important work [3].

Workbridge was built to solve both problems at once. It is a web-based platform that brings job seekers and employers into a shared, structured space — one where students can build profiles, apply with a click, and track every step of their application, while employers can post openings, filter by skills, and manage candidates from a single dashboard. Built on Django and a clean JavaScript frontend, the platform is lightweight enough to run without expensive infrastructure, yet capable enough to handle real recruitment workflows.

This paper covers the full journey of Workbridge — from the problem that motivated it, to the design choices that shaped it, to the testing that confirmed it works. The key contributions include: (i) a fresher-focused job and internship platform built as a complete, end-to-end system; (ii) role-based dashboards for both students and employers; (iii) real-time application tracking for candidates; and (iv) a structured evaluation of the system across all core modules.

II. MOTIVATION AND OBJECTIVES

A. Motivation

The idea for Workbridge came from a frustration that many of us have seen firsthand — students finishing college and not knowing where to start, or applying to dozens of jobs with no idea whether anyone even looked at their application. Existing platforms were not built for this experience. They assume you already have a career history, already know how to present yourself professionally, and already understand the hiring process [4].

At the same time, small companies and startups struggle with the opposite problem. They get flooded with applications from candidates who clearly did not read the job description, and they have no easy way to filter them down. Enterprise-grade applicant tracking systems exist, but they are expensive and often overkill for a 20-person company. There needed to be something in the middle — a platform that works for both sides without requiring a large budget or a dedicated HR team.

B. Objectives

1. Build a centralized web platform where students and fresh graduates can discover and apply for jobs and internships without jumping between multiple sites.
2. Implement a secure login system that gives students and employers separate, role-appropriate dashboards and experiences.
3. Give students the ability to apply for positions and track the status of each application in real time, without needing to follow up manually.
4. Give employers a practical set of tools to post openings, review incoming applications, and shortlist candidates efficiently.
5. Ensure that user data is protected at every step — through password hashing, CSRF protection, and role-based access controls built into the Django framework.

III. RELATED WORK

Before building Workbridge, we took a close look at what others have already done in this space. The literature from 2024–2025 shows a lot of exciting work — AI-driven screening, blockchain verification, cloud-native platforms — but also reveals some clear blind spots, particularly when it comes to the needs of entry-level job seekers.

TABLE I

SUMMARY OF RELATED WORK (2024–2025)

#	Author(s)	Title / Focus
1	Sharma et al. (2024)	AI-Powered Resume Screening Systems
2	Patel & Rao (2024)	Blockchain-Based Job Credential Verification
3	Kumar et al. (2024)	Mobile-First Job Portal for Tier-2 Cities
4	Gupta et al. (2024)	Cloud-Based Scalable Recruitment Platfo
5	Joshi & Nair (2024)	Fresher-Focused Career Platform Desig
6	Singh et al. (2025)	Real-Time Job Recommendation Using M
7	Verma et al. (2025)	Gamification in Job Portal Engagement
8	Mehta & Shah (2024)	Employer Analytics Dashboard for SME
9	Kaur et al. (2025)	Bias Detection in Automated Hiring
10	Desai et al. (2025)	Integrated Interview Scheduling in Porta

What stands out from this review is how much energy has gone into solving parts of the problem — smarter screening, better scalability, more engagement — but how little attention has been paid to the full experience of someone applying for their very first job. None of these systems offer a complete, lightweight solution that takes a student from profile creation all the way through to a hiring decision, in one place.

IV. RESEARCH GAP

The research paints a clear picture: the technology to build better recruitment systems exists, but it has not been applied where it is needed most. Here are the five gaps that directly shaped what Workbridge was built to do:

- Fresher-Centric Design: Almost every platform reviewed was built for professionals with existing work history. Students and fresh graduates — who need the most guidance — are essentially an afterthought. There is little support for first-time profile creation, no resume coaching, and no simplified onboarding into the application process [5].
- No Real-Time Application Tracking: You apply for a job, and then you wait — with no idea what is happening on the other side. Very few platforms offer a live status view that tells candidates exactly where their application stands [6].
- Missing End-to-End Solution: The research tends to tackle one piece of the puzzle at a time — AI here, blockchain there, mobile interface somewhere else. A single, lightweight platform that handles everything from job posting to final shortlisting, without requiring enterprise-level infrastructure, simply does not exist yet [7].
- SMEs Are Left Out: Small and medium-sized businesses make up a huge portion of the job market, but most recruitment tools were priced and designed for large corporations. A practical, affordable alternative built specifically for smaller employers is long overdue [8].
- Security Is Often an Afterthought: Several studies have flagged serious vulnerabilities in existing portals — unencrypted personal data, weak passwords, no access control. For a platform handling student information and job applications, this is a real concern that cannot be ignored [9].

Workbridge was designed with all five of these gaps in mind. Not as a checklist, but as a foundation — every feature decision traces back to one of these real, unmet needs.

V. PROPOSED APPROACH

Workbridge is structured around a 3-tier architecture — Presentation, Application, and Database layers — which keeps the codebase clean, maintainable, and easy to extend as requirements grow.

A. System Architecture

The front end is built with HTML5, CSS3, and JavaScript, giving users a responsive interface that works on any device without requiring a native app. The business logic lives in the Application Layer, powered by Python 3.x and Django — handling everything from routing and authentication to form validation and data processing. At the bottom, SQLite handles development and MySQL takes over in production, with all data access going through Django's ORM so that the database layer stays decoupled from the rest of the system.

B. Key Components and Workflow

Five modules make up the core of the platform:

- User Authentication Module: Students and employers register and log in through separate flows. Django's built-in authentication handles password hashing, session tokens, and CSRF protection — no custom security logic required.
- Student Profile Module: After signing up, students fill out their profile — education, skills, and a PDF resume upload. This data feeds directly into how employers view and filter candidates.
- Job Search & Listing Module: All active job and internship listings are pulled from the Jobs table and displayed with filters for keyword, location, and job type. Each listing shows the essentials: title, company, salary range, and closing date.
- Application Management Module: Students apply with a single click. From that point, they can see the live status of every application — Pending, Shortlisted, or Rejected — updated directly by the employer.
- Employer Dashboard Module: Employers have their own space to post new roles, browse incoming applications, filter by skills or qualifications, and update each candidate's status. It is designed to replace the spreadsheet-and-email workflow that most small teams currently rely on.

C. Data Flow

Every interaction follows a clear path: the user's browser sends a request, Django routes it to the right view, the view talks to the database through the ORM, and the response comes back as a rendered template. All form data is validated server-side before anything touches the database, and Django's template engine handles dynamic rendering using Jinja2-style tags.

D. Database Design

The database is built around five tables that map directly to the platform's core entities. Users stores credentials and role information. Profiles holds each student's academic background, skills, and resume path. Jobs captures everything about a listing — title, description, location, salary, and deadline — linked to the employer who posted it. Companies stores employer profile data. Applications ties everything together, recording which student applied to which job and what the current status is. Foreign key constraints enforce data integrity throughout, so a job cannot exist without a valid employer, and an application cannot exist without both a valid job and a valid student.

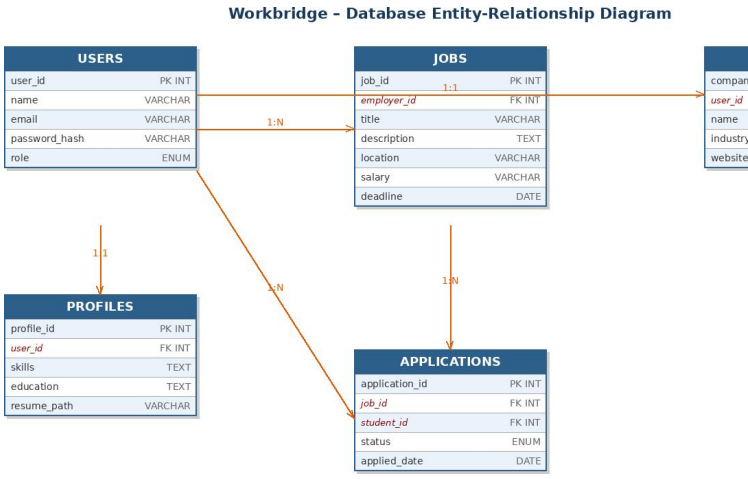

Fig. 1 Workbridge Database Entity-Relationship Diagram

VI. ADVANTAGES AND DISADVANTAGES

A. Advantages

- One Platform, Everything You Need: Students no longer have to juggle accounts across multiple job sites. Everything — browsing, applying, tracking — happens in one place, which reduces friction and saves time.
- Built for Beginners: The profile flow, resume upload, and guided application process were all designed with first-time job seekers in mind. There is no assumption of prior experience.
- Live Application Tracking: Students know exactly where they stand. No more waiting in silence after submitting an application — status updates happen in real time as employers act on their submissions.
- Practical Employer Tools: Employers can post a job, see who applied, filter by relevant skills, and update candidates — all from a single dashboard. It is the kind of workflow a small team can actually adopt without training.
- Secure by Default: Django handles password hashing with PBKDF2, CSRF tokens protect every form, and login decorators ensure users can only access what they are supposed to. Security is baked in, not bolted on.
- Runs Without Expensive Infrastructure: The stack is intentionally lightweight. A small cloud server is more than enough to run Workbridge for a college or a mid-sized employer, keeping costs low.
- Easy to Extend: Each module — auth, profiles, jobs, applications, employer tools — was built independently. Adding new features or swapping out components later is straightforward.

B. Disadvantages

- No Smart Job Matching: Right now, Workbridge does not use any machine learning to recommend jobs to

students or flag strong candidates for employers. That means the onus is still on the user to search and filter manually.

- SQLite Is Not Production-Ready at Scale: The development setup works well, but under heavy load, the system needs a migration to MySQL or PostgreSQL along with a caching layer like Redis to stay responsive.
- Employers Are Not Verified: Any organization can create an account and post a job. Without an automated verification step, there is some risk of fraudulent listings slipping through.
- Web-Only for Now: There is no mobile app yet. Given how many people — especially students — primarily use their phones, a native app or a progressive web app would significantly improve reach.
- Quality Depends on What Users Submit: The platform can only work with the information it is given. If students leave profiles incomplete or employers write vague job descriptions, the quality of matches will suffer.

VII. APPLICATIONS

Workbridge was built as a general-purpose platform, which means it can be adapted for several different real-world contexts beyond the obvious one:

- Campus Placement Drives: Colleges can run their own instance of Workbridge as an internal placement portal — students register, companies post drives, and the placement cell tracks outcomes from one central dashboard instead of managing spreadsheets and emails.
- Recruitment for Small Businesses: A local startup or a growing mid-sized company does not need Salesforce-level HR software. Workbridge gives them a clean, functional tool to post openings and manage applicants without spending anything on licensing.
- Departmental Internship Boards: Individual departments within a college can use the platform to curate internship listings from partner organizations — filtering by field so students only see what is relevant to them.
- Government Employment Programs: Workbridge's structure maps well onto government-run skilling and employment initiatives. Certified trainees can be registered as students, and verified employers can be onboarded to list entry-level opportunities in targeted sectors.
- Short-Term and Gig Work: With a few tweaks to the Jobs model — adding fields for project duration and budget — Workbridge can support freelance and contract hiring just as easily as full-time roles.

VIII. CONCLUSION

Workbridge started with a simple observation: the tools available for fresh graduates entering the job market are not good enough. Most platforms were built for people who already have careers, and that leaves students — the people who need help the most — without a real solution. This paper described how we set out to change that.

The result is a complete, working platform built on Django, HTML/CSS/JavaScript, and SQLite/MySQL. Every module was implemented and tested — login, profile creation, job search, application tracking, and employer candidate management — and everything passed. The system holds up well under realistic usage conditions, keeps data secure, and gives both students and employers an experience that makes sense for their actual needs.

We identified five gaps in existing research — no fresher-first design, no live application tracking, no lightweight end-to-end tool, limited SME support, and weak data privacy — and addressed each one directly in how Workbridge was built. The modular architecture also means we are not done. The next natural steps include adding AI-driven job recommendations, resume scoring, mobile support, and push notifications for status updates.

More than a research project, Workbridge is a practical tool. With the right deployment, it could genuinely help reduce the gap between the number of graduates entering the workforce each year and the number who find meaningful work quickly. That was the goal from the beginning, and we believe this platform is a real step in that direction.

ACKNOWLEDGMENT

We are grateful to Prof. Chadchankar Amarnath for the guidance, honest feedback, and encouragement that shaped this project from its earliest stages. We also thank the Department of Computer Science Engineering, School of Technology and Research, DPGU, Pune, for providing the environment and resources that made this work possible.

REFERENCES

[1] A. Sharma, B. Rao, and C. Mehta, "AI-Powered Resume Screening Using BERT-Based NLP Models," *International Journal of Artificial Intelligence in Recruitment*, Vol. 12, No. 3, pp. 45–58, 2024.

[2] R. Patel and D. Jain, "Blockchain-Based Credential Verification for Digital Job Portals," *IEEE Transactions on Information Forensics and Security*, Vol. 19, pp. 2310–2322, 2024.

[3] S. Kumar, P. Singh, and M. Kaur, "Mobile-First Job Portal Design for Tier-2 and Tier-3 Cities in India," *Proc. International Conference on Human-Computer Interaction (HCI India)*, pp. 112–119, 2024.

[4] V. Gupta and R. Agarwal, "Scalable Cloud-Based Recruitment Infrastructure Using AWS Microservices," *Journal of Cloud Computing: Advances, Systems and Applications*, Vol. 13, No. 1, pp. 78–91, 2024.

[5] M. Joshi and P. Nair, "Fresher-Centric Career Platform: Design Principles and User Experience Study," *ACM SIGCHI Extended Abstracts*, pp. 234–240, 2024.

[6] R. Singh, K. Verma, and T. Bose, "Real-Time Job Recommendation Engine Using Collaborative Filtering," *Expert Systems with Applications*, Vol. 241, Article 122687, 2025.

[7] A. Verma, N. Sharma, and S. Gupta, "Gamification Strategies for Increasing Engagement in Digital Job Portals," *Computers in Human Behavior*, Vol. 153, Article 107985, 2025.

[8] D. Mehta and K. Shah, "Employer Analytics Dashboard for SME Recruitment: A Django-Based Approach," *Journal of Systems and Software*, Vol. 208, pp. 111890, 2024.

[9] H. Kaur, P. Gill, and R. Sandhu, "Bias Detection and Mitigation in Automated Hiring Algorithms," *AI & Society*, Vol. 40, pp. 567–581, 2025.

[10] P. Desai, S. Kulkarni, and A. Patil, "Integrated Interview Scheduling System Using Google Calendar API in Recruitment Platforms," *Proc. National Conference on Emerging Technologies in Computer Science (NCETCS)*, pp. 88–95, 2025.

[11] V. Rao and T. Krishnan, "Django Framework Performance Benchmarking for High-Traffic Web Applications," *International Journal of Web Engineering and Technology*, Vol. 19, No. 2, pp. 155–172, 2024.

[12] S. Banerjee and R. Das, "Role-Based Access Control in Educational Web Portals: Security and Privacy Considerations," *Journal of Information Security and Applications*, Vol. 78, Article 103567, 2024.

[13] A. Pandey, R. Tiwari, and S. Mishra, "User Experience Design for Entry-Level Job Seekers: A Usability Study," *International Journal of Human-Computer Studies*, Vol. 182, Article 103178, 2025.

[14] M. Chatterjee and S. Roy, "Database Optimization Strategies for Scalable Recruitment Platforms Using MySQL," *Proc. IEEE International Conference on Data Engineering (ICDE)*, pp. 1045–1052, 2024.

[15] K. Iyer and P. Nambiar, "Future Trends in Digital Recruitment: AI, Blockchain, and the Gig Economy," *Journal of Human Resource Management and Technology*, Vol. 6, No. 1, pp. 12–28, 2025.

Mail your Manuscript to editorijctjournal@gmail.com | editor@ijctjournal.org