

**A
Project Report
on**

**KORGUT: A COMPREHENSIVE PLATFORM FOR
COLLEGES, STUDENTS, AND RECRUITERS**

Submitted to

Sant Gadge Baba Amravati University, Amravati

**Submitted in partial fulfilment of
the requirements for the Degree of
Bachelor of Engineering in**

Computer Science and Engineering

Submitted by

Prajwal Ghusalikar

(PRN: 213120191)

Chetan Chaudhary

(PRN: 213120269)

Kunal Bhende

(PRN: 213120136)

Devashish Ugale

(PRN: 213120266)

Under the Guidance of

Ms K.P Sable

Assistant Prof. Computer Science and Engineering



**Department of Computer Science and Engineering
Shri Sant Gajanan Maharaj College of Engineering,
Shegaon – 444 203 (M.S.)**

Session 2024-2025

**SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING,
SHEGAON – 444 203 (M.S.)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that **Mr. Prajwal Ghusalikar, Mr. Kunal Bhende, Mr. Devashish Ugale and Mr. Chetan Chaudhari** students of final year Bachelor of Engineering in the academic year 2024-25 of Computer Science and Engineering Department of this institute have completed the project work entitled **“KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS”** and submitted a satisfactory work in this report. Hence recommended for the partial fulfilment of degree of Bachelor of Engineering in Computer Science and Engineering.

Ms. K.P. Sable
Project Guide

Dr. J. M. Patil
Head of Department

Dr. S. B. Somani
Principal
SSGMCE, Shegaon

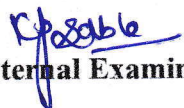
**SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGINEERING,
SHEGAON – 444 203 (M.S.)**


DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING




CERTIFICATE

This is to certify that **Mr. Prajwal Ghusalikar, Mr. Kunal Bhende, Mr. Devashish Ugale and Mr. Chetan Chaudhari** students of final year Bachelor of Engineering in the academic year 2024-25 of Computer Science and Engineering Department of this institute have completed the project work entitled **“KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS”** and submitted a satisfactory work in this report. Hence recommended for the partial fulfillment of degree of Bachelor of Engineering in Computer Science and Engineering.


Internal Examiner


Name and Signature
Date: 9/5/2025


External Examiner


Name and Signature
Date: 9/5/2025

Acknowledgement

It is our utmost duty and desire to express gratitude to various people who have rendered valuable guidance during our project work. We would have never succeeded in completing our task without the cooperation, encouragement and help provided to us by them. There are a number of people who deserve recognition for their unwavering support and guidance throughout this report.

We are highly indebted to our guide **Ms. K. P. Sable** for his guidance and constant supervision as well as for providing necessary information from time to time. We would like to take this opportunity to express our sincere thanks, for his esteemed guidance and encouragement. His suggestions broaden our vision and guided us to succeed in this work.

We are sincerely thankful to **Dr. J. M. Patil** (HOD, CSE Department, SSGMCE, Shegaon), and to **Dr. S B Somani** (Principal, SSGMCE, Shegaon) who always has been kind to extend their support and help whenever needed.

We would like to thank all teaching and non-teaching staff of the department for their cooperation and help. Our deepest thank to our parents and friends who have consistently assisted us towards successful completion of our work.

Prajwal Ghusalikar (61)

Kunal Bhende (53)

Devashish Ugale (42)

Chetan Chaudhari (40)

Contents

Abstract	i
List of Abbreviations	ii
List of Figures	iii
List of Screenshots	iv
List of Tables	v
Sponsorship letter	vi
1. Introduction	1
1.1 Preface	1
1.2 Problem Statement	1
1.3 System Objectives	2
1.4 System Overview	2
1.5 User Role and Responsibilities	3
1.6 Technical Architecture and Security	3
1.7 Benefit of Using Korgut	4
2. Literature Review	6
2.1 Introduction to Proposed System	6
2.2 From Referral to Algorithms: A Look Back and Ahead	6
2.3 The Challenges Still Holding Us Back	7
2.4 How AI Is Revolutionizing the Recruitment	7
2.5 Bias and Security	8

2.6 Service Oriented Architecture and Secure APIs	9
3. Methodology	12
3.1 Existing System and Proposed System	12
3.2 System Model	12
3.3 Multi-Tier Architecture	13
3.4 Role-Based Methodology	14
3.5 Security And Compliance	19
4. Implementation	21
4.1 Technology Stack	21
4.2 Frontend Implementation	21
4.3 Backend Implementation	22
4.4 Authentication Layer	22
4.5 API Communication and Role-Based Access	23
4.6 Database Layer and Schema	23
4.7 Real time chat communication	24
4.8 Testing to Quality Assurance	24
4.9 Monitoring and Optimization	24
5. Result and Discussion	27
5.1 Job Matching Efficiency	27
5.2 Integration And Compatibility	28
5.3 User Experience and Satisfaction	29
5.4 Real Time Chat Module	29

5.5 Future Enhancement and Consideration	31
6. Conclusion	33
6.1 Conclusion	33
References	36
Research Certifications	37
Plagiarism Report (using Turnitin software)	42
Project Group Members	43

Abstract

The quick digitalization in the recruitment and education domains requires integrated platforms that streamline the interactions of students, colleges, recruiters, and faculty. Korgut is an integrated platform aimed at bridging the gaps between students, colleges, and recruiters by creating a structured and streamlined management system. Students can create detailed portfolios covering personal information, projects, experience, certifications, social connections, resumes, and bios, promoting professional networking and career development. Colleges gain from streamlined student information management, such as bulk uploading, password auto-generation, course management, event planning, and email communication, promoting effective academic administration. Recruiters are able to post jobs, monitor applications, and shortlist candidates with ease, streamlining the recruitment process. Faculty members have a critical role in authenticating student portfolios, thus ensuring quality standards and mentoring student development. Through the integration of these features, Korgut supports employability, academic effectiveness, and hiring procedures, positioning it as a beneficial tool for talent acquisition and higher education. The architecture of the platform, main features influence on student career development, and its future potential for educational technology are presented in this paper.

Keywords: *College Administration, Student Management, Recruiter Portal, Faculty Verification, Digital Portfolio, Academic Efficiency, Recruitment Process.*

List of Abbreviations

Abbreviation	Description
IPFS	Role-Based Access Control
GDPR	General Data Protection Regulation
NLP	Natural language processing
REST	Representational State Transfer
SOA	Service Oriented Architecture
JWT	JSON Web Token
SSR	Server-Side Rendering
SEO	Search Engine Optimization
AWS	Amazon Web Services
NPM	Node Packet Manager
API	Application Programming Interface

List of Figures

Figure No.	Description	Page No
Figure 3.1	System Model	13
Figure 3.2	College and Student Workflow	16
Figure 3.3	Student Workflow	18
Figure 3.4	Recruiter Workflow	19

List of Screenshot

Screenshot No.	Description	Page No
Screenshot 4.1	Student information	22
Screenshot 4.2	Django User and Employer Models	22
Screenshot 4.3	Django Faculty and Uploaded Files Model	23
Screenshot 4.4	React fetching posts using axios	25
Screenshot 5.1	Hire Student as per job profile	27
Screenshot 5.2	CSV Upload files	28
Screenshot 5.3	Student Login Home Screen	29
Screenshot 5.4	Chat interface at student side	30
Screenshot 5.5	Chat interface at recruiter side	30

List of Tables

Table No.	Description	Page No
Table I	Components and their features	25

Sponsorship letter



One Smarter, Inc.

6-12-2024

Prof. K.P. Sable
Dr. J.M. Patil
Dept. of Computer Science and Engineering
SSGMCE, Shegaon, MS, India

Dear Prof. Sable/Dr. Patil;

My company, One Smarter, Inc. will support and sponsor the following project:

Project Name: Development of a platform – Korgut.com

Student Group:

1. Prajwal Ghosalikar
2. Devashish Ugale
3. Kunal Bhende
4. Chetan Chaudhary

For One Smarter, Inc., this is an excellent opportunity to assess the capabilities of these students and strengthen our relationships with your Department and College. We will support a sponsorship amount of INR 15,000 and additionally, consider (after review) additional support as the students might require.

We expect that the developed software will then become a commercial product or One Smarter Inc. and no ownership will be exercised of the product by either the College or the students. We will, of course, give due publicity to the College and your Department for the assistance you are providing us with in creating another product in our portfolio.

Thank you and I look forward to our collaboration. Please contact me at Pete.Hager@onesmarter.com if you wish to discuss this support.

Thanks & Regards,

Recoverable Signature

X Pete Hager

Pete Hager
President & COO
Signed by: 0002a672-0d9b-484b-a623-1b1ad4aa3789



+ 1 937344 6241



care@onesmarter.com



4031 Colonel Glenn Hwy Ste 100 Beavercreek, OH 45431

www.onesmarter.com

CHAPTER 1
INTRODUCTION

INTRODUCTION

1.1 PREFACE

In the fast-paced and highly competitive job market of today, the path from college to career is anything but seamless. Students are left scrambling to present their skills in a manner that really stands out. Recruiters struggle to break through the noise to find the best talent. Colleges are overwhelmed by administrative burdens, data silos, and antiquated systems. It results in missed opportunities, sluggish processes, and results that don't meet anyone's potential.

That's where Korgut comes in. Conceived out of an evident need to make recruitment smarter, faster, and more collaborative, Korgut is built as an all-in-one platform for students, recruiters, and institutions. It consolidates everything—from online student portfolios to job postings, faculty feedback, and admin features—into one cohesive system.

Korgut is not only another job platform; it's an entire recruitment ecosystem designed specifically for the contemporary, digital-first age. It leverages automation, data analytics, and engaging tools to simplify and optimize the campus recruitment process. This report walks you through what Korgut does, why it does it, and how it's set to redefine recruitment in tech-enabled world.

1.2 PROBLEM STATEMENT

Traditional hiring processes are plagued by fragmented communication, ineffective data management, reduced student exposure, overwhelmed recruiters, and a lack of integration between college campuses and industry. Students, recruiters, and teachers work in silos, leading to delays, confusion, and disconnection. Outdated manual processes make it difficult to track and retrieve records, resulting in lost, copied, or unavailable data. Students struggle to showcase their talents and personalities, while recruiters are overwhelmed by the sheer volume of resumes. Korgut aims to revolutionize this process by designing a platform that fosters collaboration, transparency, and efficiency, bridging the gap between education and industry to create a more streamlined and effective hiring experience for all parties.

1.3 SYSTEM OBJECTIVES

- Create a seamless online platform that brings together students, colleges, and recruiters, making communication and collaboration smooth and effortless.
- Provide intuitive features for students to manage their academic records, project portfolios, certifications, and track their job application progress, all in one place.
- Implement a job recommendation engine that understands each student's profile and suggests the most relevant career opportunities.
- Protect sensitive information through strong password management systems, secure authentication methods, and encrypted data processes.
- Integrate automated email alerts and notifications to keep users updated about important actions like application status, recruiter messages, and system updates.
- Help students showcase their achievements and skills effectively, creating a strong professional presence right from their academic year.

1.4 SYSTEM OVERVIEW

Fundamentally, Korgut is a cloud-based recruitment platform that unites students, colleges, recruiters, and faculty in a common digital space. Imagine it as a campus-wide, career-oriented social network—one that's specifically designed for wiser hiring.

Here's how Korgut contributes:

Student Portfolios: An active, online resume where students can present everything—academic history and certifications to personal projects, internships, accomplishments, resumes, and even social media handles.

Job Applications: Students can directly apply to jobs, monitor application status, mark interests in bookmarked roles, and even engage directly with recruiters—within the app.

Recruiter Dashboard: Command center for hiring managers to publish openings, shortlist and filter candidates, send notifications, and obtain analytics on hiring success.

College Admin Tools: Administrators can bulk upload student data, create courses, monitor student progress, manage placement events, and send updates real-time.

Faculty Access: Teachers and mentors can view student profiles, authenticate achievements, offer academic feedback, and provide career guidance based on industry applicability.

Social Interaction: Share, comment, like, and post updates. This creates a community and fosters professional networking from the very beginning.

All these features coalesce to save time by avoiding redundancy, increase visibility, and provide a smoother, more intuitive hiring experience throughout.

1.5 USER ROLE AND RESPONSIBILITIES

Korgut is centered on the requirements of various stakeholders, and each user role receives customized access and tools.

Students: Develop their own brand with interactive portfolios, find and apply for jobs, connect with classmates, and reach recruiters.

Recruiters: Identify top candidates more quickly by searching rich profiles, posting opportunities, filtering candidates, and monitoring engagement.

Colleges: Streamline student records, authenticate credentials, and track placement performance in real time.

Faculty: Become more actively engaged with helping students through review of profiles, accuracy checking data, and coaching to help both academic and career success.

Role-based design so everybody is aware where they are supposed to fit and how to benefit from the platform most effectively.

1.6 TECHNICAL ARCHITECTURE AND SECURITY

Under the hood, Korgut is constructed from a modular Service-Oriented Architecture (SOA)—which translates to flexible, scalable, and easy to maintain. Each module operates in isolation but integrates through RESTful APIs, allowing for effortless communication between modules such as student information, recruiter panels, and admin controls.

Korgut features:

1. Django Auth-token for secure authentication
2. Role-Based Access Control (RBAC) to restrict data exposure
3. Encrypted database transactions to avoid data leaks

4. API rate limiting and input validation for preventing abuse
5. Compliance with international standards such as GDPR and CCPA
6. The platform is also tested intensively—everything from static code review to black-box testing—to make sure it's secure, stable, and reliable.
7. Semantic Resume Parsing: Understands the reasoning behind a candidate's skills and experience for enhanced matching.
8. Job Recommendation Engines: Employ data to recommend jobs matched to each student's interests and aptitudes.
9. Behavioral Analytics: Prepares students for interviewing by analyzing interaction style and soft skills.

And it doesn't end there. Korgut also engages alumni by providing them with dedicated space to mentor students, post job opportunities, and provide real-world guidance.

1.7 BENEFIT OF USING KORGUT

For Students:

- Get job suggestions that match their profile
- View how peers are faring and get motivated
- Establish a solid, visible personal brand right from day one

For Recruiters:

- Save time and effort with smart filtering
- Connect directly with validated, high-quality candidates
- Have complete academic records at a glance

For Colleges:

- Reduce administrative burdens
- Monitor placement metrics in real time
- Monitor placement metrics in real time
- Have all student data handy and accessible

For Faculty:

- Remain actively engaged in student career paths
- Track student development and preparedness for the field
- Provide timely, relevant mentoring and critique

CHAPTER 2
LITERATURE
REVIEW

LITERATURE REVIEW

2.1 INTRODUCTION TO PROPOSED SYSTEM

How business recruits' people have completely shifted—particularly over the last few years since the COVID-19 outbreak. No more newspaper classifieds, campus tables, and face-to-face interviews. Now, it's a digital world where mobile platforms, artificial intelligence (AI), and data analytics are leading roles to bring talent and opportunity together—frequently across international borders.

This review dives deep into how hiring has evolved over the years. It calls attention to the pains of growing up that this has entailed and delves into emerging technology—ranging from semantic search and machine learning to secure APIs and service-oriented architecture (SOA). It also brings to the fore something that is often taken for granted: how alumni networks can become game-changing partners in the hiring process.

2.2 FROM REFERRAL TO ALGORITHMS: A LOOK BACK AND AHEAD

Hiring was once a more personal, simple affair. Employers used word of mouth, newspaper ads, campus interviews, and simple interviews. It was fine—when candidates were scarce, and positions were local. But as businesses expanded and the labor market grew, those techniques began to crack.

Enter the early 2000s: job portals such as Naukri and Monster revolutionized the game by providing employers with access to a wider pool of applicants. Fast-forward to the present, and hiring is highly data- and AI-driven. Recommendation systems now occupy the center of most hiring platforms, employing different filtering techniques to pair individuals with jobs [1].

For example, collaborative filtering examines user trends and tastes in order to recommend roles similar to those applicants also applied for. Content-based filtering explores the contents of resumes and job descriptions in order to make good matches. The hybrid approach—combining both methods—produces even more superior results by balancing personalization and precision.

Smartphones have also changed the process. Mobile-first recruitment platforms enable job applicants to apply using their smartphone, get interview invitations, and even participate in virtual interviews, all from the palm of their hands. Mix it with gamification—engaging quizzes and instant challenges—and you have innovative methods to test soft skills and problem-solving skills.

University sites, as well, have become AI-friendly. Digital portals of today are no longer mere repositories for resumes; they provide dashboards that monitor student progress, forecast career options, and link to sites like LinkedIn to provide a more complete picture of every candidate.

2.3 THE CHALLENGES STILL HOLDING US BACK

While the technology has improved, recruitment systems of today are far from utopian. In fact, they arrive with an entirely new set of hassles.

One of the big problems is the deluge of applications. With a few clicks, candidates can apply to dozens of vacancies, and HR teams are left struggling and having to rely on machines to shortlist the first round. That raises questions over the accuracy and fairness of computer-generated shortlisting.

For universities, maintaining current records and confirming credentials continues to be a logistical nightmare. Most institutions have not built a centralized, clean, structured database, making it difficult for employers to confirm student accomplishments or monitor placement histories. This is an invitation to falsified claims and misrepresented skills [8].

And what of alumni? They're a treasure trove of experience and contacts, but most sites don't do anything to engage them in a useful way. These are people who can mentor students, provide companies with candid feedback about talent, and introduce students into their organizations—but their value isn't being exploited.

2.4 HOW AI IS REVOLUTIONIZING THE RECRUITMENT

The advancements AI has made to recruitment are truly astonishing. Now, recruitment platforms don't just see what's been put on a resume—they read between the lines,

consider the way it's written, the behavior of the candidate during video interviews, and even emotional signals [3].

Natural language processing (NLP) enables systems to decipher the real intent behind the words in job postings and resumes. This translates to improved candidate-to-job matching, rather than keyword matching. At the same time, ranking algorithms analyze candidates on multiple facets—from experience and qualifications to cultural fit and soft skills [4].

Most systems today draw upon outside platforms such as LinkedIn to bring in user information automatically, which accelerates the screening process and provides yet another layer of intelligence. In the background, all of this is enabled by modular software design—component-based architecture[6]. The reusable components make recruitment platforms scalable and configurable, reducing development cost and complexity.

AI chatbots are also filling in to enhance the candidate experience. They respond to FAQs, coordinate interviews, and perform first-round screenings. AI-powered video interviews can now decode facial expressions, evaluate tone of voice, and mark communication breakdowns—though these technologies remain to be perfected to prevent reinforcing biases or misinterpreting cultural nuances [9].

2.5 BIAS AND SECURITY

With great ability comes great responsibility—and AI recruitment is no different. As the role of machines increases, so do ethical issues.

Perhaps the largest issue is algorithmic bias. If the data that an AI system is trained on includes biases—based on race, gender, education, or geography—the system will reproduce them. That means some groups could be unfairly excluded or scored lower, even if they're completely qualified.

Then there's the lack of transparency. Many candidates don't understand why they were rejected or what criteria were used to evaluate them, making the process feel opaque and impersonal.

Another important concern is security. Most platforms employ RESTful APIs to communicate between databases, apps, and services. But these APIs themselves can be vulnerable to cyber threats if they're not well protected. Practices such as OAuth 2.0 for safe logins, role-based access control, input checking, and rate limiting for APIs are critical in order to make sensitive information safe[7].

To further secure platforms, developers are employing advanced security testing techniques—such as static code analysis and black-box testing—to detect vulnerabilities. And with global regulations such as GDPR and CCPA, businesses are held to higher expectations regarding data privacy, consent, and transparency.

2.6 SERVICE ORIENTED ARCHITECTURE AND SECURE APIs

Behind every successful recruitment platform is a strong technical foundation. That's where service-oriented architecture (SOA) enters the picture. SOA decomposes big systems into smaller, standalone services that communicate with one another via APIs. This modular design makes platforms simpler to scale, update, and maintain.

RESTful APIs are central to this setup. They allow data to flow between universities, students, alumni, and recruiters seamlessly. But without proper security, APIs can become weak links. Developers must adopt secure practices—like encrypting data, using authentication tokens, and regularly testing endpoints—to prevent breaches[1].

APIs must also remain resilient under load. During a campus placement or mass recruitment drive, traffic might surge. Methods such as caching, asynchronous execution, and load balancing ensure that things keep going forward.

Good API design isn't merely about security and performance—it's also usability. Non-tech users such as college staff or HR representatives should be able to use interfaces without getting frustrated. Good documentation, sandbox environments for testing, and version control facilitate ease of integration and future-proofing of the platform[6].

Open-source recruitment tools bring flexibility and community-driven improvements but at the cost of needing constant watchfulness to avoid exploitation or security breaches[4].

CHAPTER 3
METHODOLOGY

METHODOLOGY

3.1 EXISTING SYSTEM AND PROPOSED SYSTEM

Existing System:

- Traditional Job Portals: Limited to basic job postings and applications, lacking specific tools for colleges to manage and share student data effectively.
- College Management Systems: Primarily focused on internal data management, with limited or no interaction with recruiters and students' external portfolios.
- Recruitment Agencies: Typically operate independently, leading to inefficiencies in connecting the right candidates with the right employers.

Proposed System:

- Centralized Platform: Combines the functionalities of job portals, college management systems, and recruitment agencies into a single, integrated platform.
- College keeps tracking students: Colleges can keep track of individual students' job applications, portfolios, shortlisting, and more.
- Dedicated User Sections: Specific interfaces and tools for students, colleges, and recruiters to ensure a tailored and user-friendly experience.
- Enhanced Collaboration: Enables seamless communication and collaboration between students, colleges, and recruiters.

3.2 SYSTEM MODEL

Korgut system model contains the functional and technical elements that facilitate effortless interaction among students, teachers, college administrators, and recruiters. It is developed as a role-based, modular, and scalable architecture with support for real-time communication, data integrity, and intelligent automation for campus recruitment.

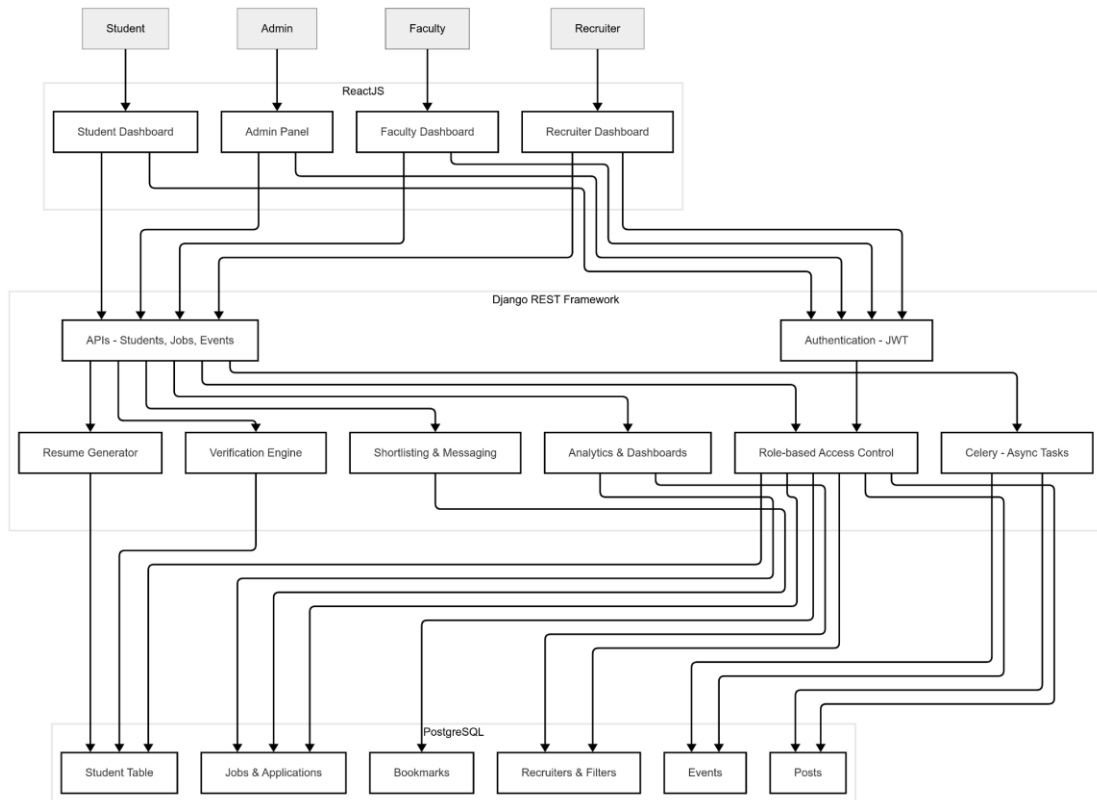


Figure 3.1 System model

3.3 MUTI-TIER ARCHITECTURE

1. Presentation Layer (Frontend – React.js)

Korgut provides user-specific interfaces according to authentication roles, ensuring a seamless experience for students, faculty, and recruiters. Students have access to a range of features, including job postings, a portfolio constructor, resume downloads, and social media posts, as well as the ability to engage in real-time chats with recruiters. Faculty members can view dashboards for student authentication, academic information uploads, and recruiter engagements. Meanwhile, recruiters have access to powerful tools, including filters, analytics, messaging features, and candidate profiles. To facilitate these features, Korgut leverages Axios for API requests and React Router for routing, ensuring a smooth and efficient user experience.

2. Application Layer (Backend – Django REST Framework)

The backend of Korgut serves as the core business logic, responsible for handling API requests, permissions, and workflows. It enforces secure, role-based access through Django Authentication, ensuring that users only have access to authorized features and

data. The backend utilizes ViewSets, Serializers, and Signals to efficiently handle user activities, such as shortlisting, job application, and profile updates. Additionally, it hooks into background services for tasks like emailing, creating PDFs, and handling CSV uploads, streamlining the overall user experience and improving the efficiency of the platform.

3. Data Layer (PostgreSQL Database)

The Data Layer of Korgut is powered by a PostgreSQL database, which serves as a centralized repository for storing all structured and semi-structured data. This includes student profiles, job applications, event attendance, and messages, among other relevant information. The database is designed to maintain referential integrity through the use of many-to-many and foreign key relationships, ensuring data consistency and accuracy. Additionally, the database utilizes JSONB fields to store resumes in a flexible and adaptable format, and TSVector for full-text searching, enabling efficient and effective data retrieval and analysis.

3.4 ROLE-BASED METHODOLOGY FLOW

1. Faculty And College Workflow

The faculty are important in data management, verification, and interaction with recruiters. The system supports:

Student Portfolio Verification:

- Faculty log in through authenticated React sessions and view student lists allocated to their department.
- Django offers a custom admin interface and APIs for authenticating uploaded certificates, grades, and projects.
- Verified status is updated in the student dashboard.

Academic Data Management:

- Bulk student records are imported using CSV files. A Python and Pandas-written backend script parses, cleans, and stores data into PostgreSQL.
- This minimizes manual entry and enables department-level academic dashboards.

Faculty Division:

- Every department is auto-allocated as a faculty coordinator through dynamic role assignment.
- The backend applies access control so every faculty member can just view their own departmental data.

Communication with Recruiters:

- Django Channels or Celery is utilized for async task management (e.g., sending recruiter emails, student verification updates).
- Faculty members can approve/reverify data prior to sharing student lists for interviews.

Student registration by the college

- Students are officially onboarded to the Korgut platform through institutional registration, ensuring verified enrollment and academic legitimacy.
- This verification-led onboarding strengthens platform security, prevents fake profiles, and ensures that all users are legitimate students of the institution.

College department structure:

Korgut follows the same structure as the college's academic setup, with each department—like Computer Science, Electronics, or Business Administration—functioning as its own unit within the system.

This setup makes it easy to manage students, events, and faculty roles at the department level. Each department gets its own dashboard where faculty can track student progress, placements, and interactions with companies.



Figure 3.2 College and Student Workflow

2. Student workflow

Students engage with the site to construct their careers using search, application, bookmarking, and resume construction.

Bookmark Section:

- Students may bookmark job postings. These bookmarks are saved through React state and synchronized with PostgreSQL via PATCH API requests.
- A different module fetches all bookmarks with corresponding metadata.

Event Tracking:

- Students see and sign up for upcoming events, retrieved through a filtered GET request from the /api/events/ endpoint.
- Event registration status is governed through many-to-many relationships within the database.

Resume Generation:

- Django utilizes templates for dynamically rendering resumes with saved student data as portfolio.
- Resume downloading is dealt with secure endpoints where token expiration validations are checked.

Posts & Networking:

- Posts by students are available for sharing and are saved as public posts within the database posts table.
- React dynamically renders a feed with like, comment, and share operations.

Search Bar:

- Full-text search API based on database enables finding recruiters, jobs, or other students.
- React provides debounce and asynchronous API call handling for search optimization.

Job Section:

- Students submit applications for jobs via a specific React form that posts to an /apply-job/ endpoint.
- The backend checks if the student is eligible before storing the application.

Chat section:

- Students can engage in real-time conversations with recruiters through the integrated chat feature, allowing them to ask questions, clarify job details, and build stronger professional connections instantly.

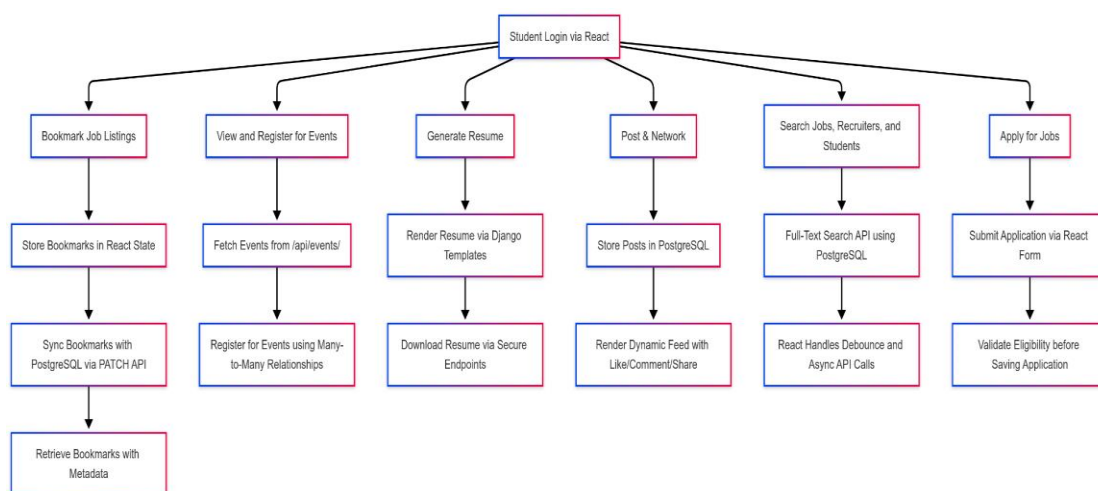


Figure 3.3 Student workflow

3. Recruiter Workflow

Recruiters have tools for talent discovery, direct communication, and shortlisting of candidates.

Portfolio Access:

- Recruiters are able to access validated student data fetched from PostgreSQL through a safe, filtered API (read-only).
- The UI offers search filters such as degree, CGPA, skillset, and availability.

Shortlisting/Unshortlisting:

- APIs enable shortlisting/unshortlisting students and saving that state through a PATCH request.
- Notifications are sent through Django signals.

Advanced Subscription Model:

- Premium recruiters can enable analytics dashboards developed with Chart.js or Recharts.
- Django REST permissions and middleware control access to premium routes.

Contacting Colleges:

- A secure messaging API facilitates interaction between faculty and recruiters.
- Optional integration with WhatsApp and email gateways can be achieved via third-party providers such as Twilio or SendGrid.

Real-time chat feature:

- Recruiters can initiate or respond to real-time chats with potential candidates, making it easier to assess communication skills, provide feedback, and fast-track the hiring process through instant interaction.

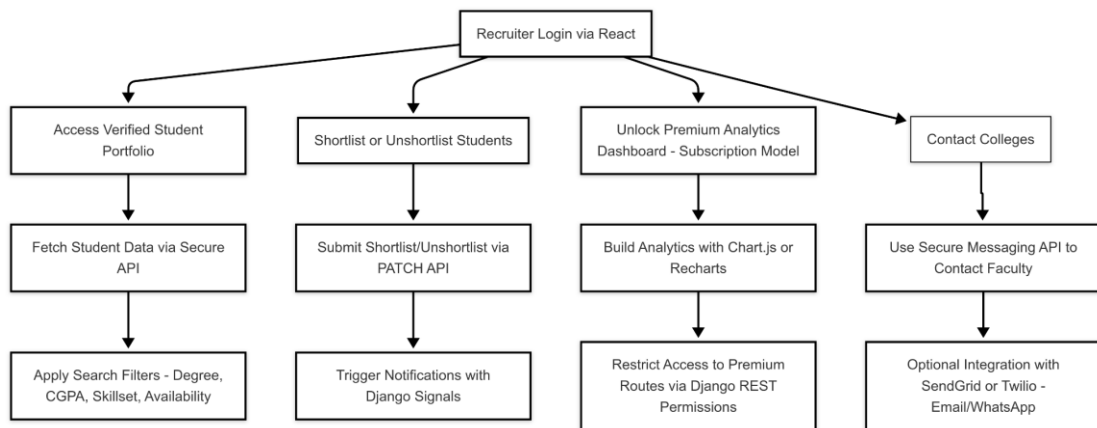


Figure 3.4 Recruiter workflow

4. Data and API Flow

All user interactions, including login, job applications, and profile updates, are facilitated through the Django REST Framework backend. To ensure the security and integrity of these interactions, Django authentication is employed to protect user sessions. Additionally, middleware is utilized to manage Cross-Origin Resource Sharing (CORS), rate-limiting, and token validation, providing an extra layer of protection against unauthorized access. Furthermore, API throttling is enforced to prevent abuse by recruiters or students sending multiple requests, thereby safeguarding the system against potential security threats and maintaining a fair and efficient user experience.

3.5 SECURITY AND COMPLIANCE

Korgut prioritizes security and data management through several key measures. Encrypted passwords are stored via Django's internal hashing mechanisms, ensuring the protection of sensitive user information. Role-Based Access Control is implemented through separate models and endpoints for students, faculty, and recruiters, guaranteeing that each user group has access only to relevant features and data. Audit logs are maintained through Django signals, tracing and time-stamping all actions for transparency and accountability. Additionally, Korgut is GDPR-ready, allowing

students to delete their data upon graduation or account closure. To facilitate collaboration, ongoing improvement, and sound software quality, the platform follows best practices in version control, deployment, and testing.

1. Version Control (Git + GitHub)

Source Code Management: The whole codebase is stored using Git and hosted on GitHub.

Branching Strategy: A GitFlow approach is adopted—comprising main, development, and feature branches to separate production-grade code from continuous development.

Pull Requests and Code Reviews: Every change to the code goes through peer review through GitHub pull requests to ensure high code quality and avoid regressions.

2. Deployment to AWS

Hosting Infrastructure: The backend is containerized with Docker and hosted on AWS EC2 instances for scalability and high availability.

Static Asset Hosting: Frontend builds are hosted through AWS S3 and optionally tied with CloudFront CDN for accelerated asset delivery.

Database Management: Postgres databases in AWS RDS have automated backup policies, multi-AZ replication, and role-based access.

Environment Configuration: Sensitive environment variables are kept secure through use of AWS Secrets Manager or EC2 parameter store to protect configuration keys and credentials.

3. API Testing using Postman

Manual Testing Suite: All Django REST Framework-generated RESTful APIs are tested through Postman collections.

Authentication Handling: Token-based authentication is mimicked in Postman through pre-request scripts in login flows.

Environment Variables: Postman environments are used to toggle development, staging, and production URLs.

CHAPTER 4

IMPLEMENTATION

IMPLEMENTATION

Korgut is a comprehensive, web-based recruitment and academic coordination platform connecting students, faculty, recruiters, and colleges. The system ensures transparency, credibility, and efficiency in academic record handling, resume generation, and job placement workflows. Emphasizing a semi-automated verification system overseen by faculty, Korgut reduces fraudulent entries and provides reliable data to recruiters. The platform supports various user roles—students, faculty, recruiters, and admin—each with tailored dashboards and access.

4.1 TECHNOLOGY STACK:

Frontend: React.js, Tailwind CSS, Axios, HTML, CSS, JavaScript

Backend: Django, Django REST Framework, Redis, WebSocket, Pandas

Database: PostgreSQL, sqlite3

Authentication: Django Auth Token

Deployment: AWS, GitHub Actions

Testing: Postman

4.2 FRONTEND IMPLEMENTATION

Built using React.js, the frontend uses modular components and modern design principles: Componentized UI using Tailwind CSS

1. Dynamic Routing via React Router
2. Axios for API calls
3. Role-Based Dashboards: Conditionally render dashboards for different users
4. Resume Builder: Dynamically generated PDF exports
5. Job Application Forms: Pre-filled data from state management

Screenshot 4.1 Student information

4.3 BACKEND IMPLEMENTATION:

1. Django REST Framework handles API endpoints
2. Model-View-Serializer pattern to enforce consistency
3. Celery with Redis manages asynchronous tasks like email

```
class User(AbstractBaseUser, PermissionsMixin):
    email = models.EmailField(unique=True)
    user_type = models.CharField(max_length=10)
    is_active = models.BooleanField(default=True)
    is_staff = models.BooleanField(default=False)
    is_superuser = models.BooleanField(default=False)

    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = []

    objects = UserManager()

from django.utils import timezone

class Employer(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, related_name='employer')
    company_name = models.CharField(max_length=255)
    company_description = models.CharField(max_length=255, null=True)
    emp_logo = models.ImageField(upload_to='employer_logos/', null=True, blank=True)
    emp_location = models.CharField(max_length=255, null=True)
    password = models.CharField(max_length=128)
    account_verified = models.BooleanField(default=False)
    verified_at = models.DateTimeField(null=True, blank=True)
    status = models.BooleanField(default=True)
    last_login = models.DateTimeField(null=True, blank=True)
    paid_membership = models.BooleanField(default=False)
    free_profile_viewd = models.IntegerField(default=0)
    admin_block = models.BooleanField(default=False)
    created_at = models.DateTimeField(default=timezone.now)
    updated_at = models.DateTimeField(auto_now=True)
```

Screenshot 4.2 Django User and Employer Models

4.4 AUTHENTICATION LAYER:

JWT tokens are issued on login and stored in cookies OAuth 2.0 support for login
Django Middleware enforces session checks

4.5 API COMMUNICATION AND ROLE-BASED ACCESS

Each user role has access to specific endpoints:

Students: apply to jobs, upload resumes

Faculty: verify students, communicate with recruiters

Recruiters: post jobs, view verified students

4.6 DATABASE LAYER AND SCHEMA

SQL schema includes many-to-many (student-job) and one-to-many (faculty-department) relations

Migration files ensure schema integrity

Bulk Uploads: Handled via Pandas and Excel files

```
class Faculty(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, related_name='faculty')
    # college = models.ForeignKey(College, on_delete=models.CASCADE, related_name='college')
    college_name = models.CharField(max_length=255, null=True, blank=True)
    first_name = models.CharField(max_length=255)
    last_name = models.CharField(max_length=255)
    department = models.CharField(max_length=255)
    designation = models.CharField(max_length=255)
    mobile = models.CharField(max_length=15, null=True, blank=True)
    profile_image = models.ImageField(upload_to='faculty_profiles/', null=True, blank=True)
    status = models.BooleanField(default=True)
    date_joined = models.DateTimeField(default=timezone.now)
    last_login = models.DateTimeField(null=True, blank=True)
    admin_block = models.BooleanField(default=False)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return f"{self.first_name} {self.last_name} - {self.college.college_name}"

# class UploadedFile(models.Model):
#     file = models.FileField(upload_to='uploads/')
#     college = models.ForeignKey(College, on_delete=models.CASCADE)
#     uploaded_at = models.DateTimeField(auto_now_add=True)

#     def __str__(self):
#         return f"{self.file.name} - {self.college.college_name}"

class UploadedFile(models.Model):
    file = models.FileField(upload_to='uploads/')
    college = models.ForeignKey(College, on_delete=models.CASCADE)
    uploaded_at = models.DateTimeField(auto_now_add=True)
    num_records = models.IntegerField(null=True) # Store the number of records in the file
    file_name = models.CharField(max_length=255, null=True)

    def __str__(self):
        return self.file_name
```

Screenshot 4.3 Django Faculty and Uploaded Files Model

4.7 REAL TIME CHAT COMMUNICATION

Korgut features a direct chat feature for recruiters and students to talk within the platform. This feature is implemented with WebSockets, driven by Django Channels on the backend and integrated with React on the frontend for a seamless, live messaging experience.

Every chat session is authenticated to ensure privacy and role-based access. Recruiters can start or reply to conversations with students who have applied for their job ads, while students can inquire or check on the status of their application.

Messages are saved in a PostgreSQL database with efficient relational mapping, and chat histories can be fetched at any point in time for reference. Typing indicators, read receipts, and updates are handled in real time without page refresh.

To ensure performance and scalability, Redis is employed as a message broker and Celery manages background operations like notification delivery. This module of communication not only increases transparency but also accelerates the recruitment process by eliminating delays in interaction.

4.8 TESTING TO QUALITY ASSURANCE

Unit Testing: unittest and Pytest for backend

GET requests for job lists

Postman: API collection for manual tests

Integration Tests: React Testing Library for UI flows

4.9 MONITORING AND OPTIMIZATION

Lazy Loading and Paginations in React

PostgreSQL Indexes on frequently queried columns

Django Signals to trigger logs and notifications

Table I Components and their features

Feature/Component	Description
Faculty Verification	Enables faculty to validate student profiles, significantly reducing fraud and enhancing the credibility of submitted portfolios.
RESTful APIs + OAuth	Facilitates secure and standardized communication across the system, while allowing easy integration with external services like LinkedIn.
AWS Cloud Deployment	Ensures the platform is scalable and highly available, with reliable cloud-based storage and computing through AWS services (EC2, S3, RDS).
Postman & Automated Testing	Guarantees API reliability through manual and automated test suites, helping identify bugs early and maintain performance.
User Feedback Integration	Direct input from student, recruiter, and faculty users helped refine features, improve workflows,

```

1  import React, { useContext, useState, useEffect } from "react";
2  import axios from "axios";
3  import PopularPost from "../PopularPost";
4  import ProfileView from "../ProfileView";
5  import Post from "../Post";
6  import CreatePostModal from "../CreatePostModal";
7  import MyContext from "../../ContextApi/MyContext";
8  import { useNavigate } from "react-router-dom";
9  import LoadingIndicator from "../../Indicators/LoadingIndicator";
10
11 function StudentHome() {
12   const [loading, setLoading] = useState(false);
13   const { api } = useContext(MyContext);
14   const studentInfo = JSON.parse(localStorage.getItem("studentInfo"));
15   const token = studentInfo.token;
16   const [postSubmittedCounter, setPostSubmittedCounter] = useState(0);
17   const [posts, setPosts] = useState([]);
18   const [searchFilter, setSearchFilter] = useState("");
19   const navigate = useNavigate();
20   useEffect(() => {
21     getAllPosts();
22     setPostSubmittedCounter();
23   }, [postSubmittedCounter]);
24
25   const getAllPosts = async () => {
26     try {
27       setLoading(true);
28       const response = await axios.get(`${api}/api/student/posts/`, {
29         headers: {
30           "Content-Type": "multipart/form-data",
31           Authorization: `Token ${token}`,
32         },
33       });
34       const data = response.data;
35       setPosts(data);
36     } catch (error) {
37       console.log("Error:", error);
38     } finally {
39       setLoading(false);
40     }
41   };
42
43   return (
44     <div className="flex flex-col md:flex-row pt-14">
45       </div student home sidebar ~/>
46       <div className="flex flex-1">
47         <div className="flex items-center mb-4">

```

Screenshot 4.4 React fetching posts using axios

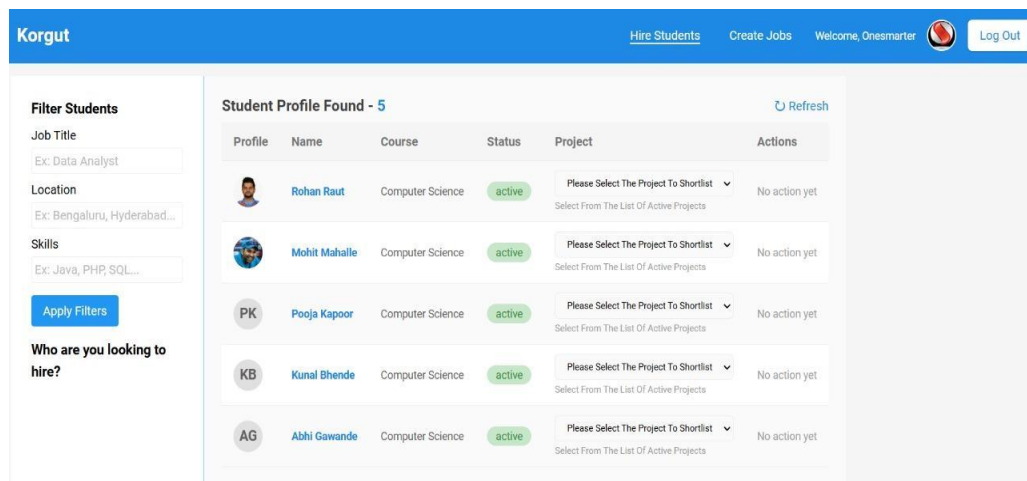
CHAPTER 5
RESULT
AND
DISCUSSION

RESULT AND DISCUSSION

5.1 JOB MATCHING EFFICIENCY: The primary purpose of Korgut is to enable informed and transparent job matching between recruiters and students. The semi-automated approach of the platform differentiates it from conventional recruitment systems by allowing informed decision-making without excessive reliance on algorithmic automation. Structured portfolio plays a key role in this by allowing students to showcase an holistic picture of their abilities—comprising verified educational records, technical certifications, co-curricular activity achievements, and completed projects.

Korgut-using recruiters leverage portfolios labeled clearly, vetted by faculty. While fully automated platforms misclassify or ignore critical characteristics, Korgut enables recruiters to sort and shortlist candidates manually. Underpinned by actual-time filters for CGPA, degree, domain expertise, and availability, recruiters have access to specific talent pools through Korgut.

The verification mechanism through faculty enhances trust and credibility, ascertaining students presented to recruiters meet institutional standards of quality. These authenticated records are accessible via the recruiter dashboard through secure GET APIs, enhancing recruiter confidence and improving the quality of job matches considerably.

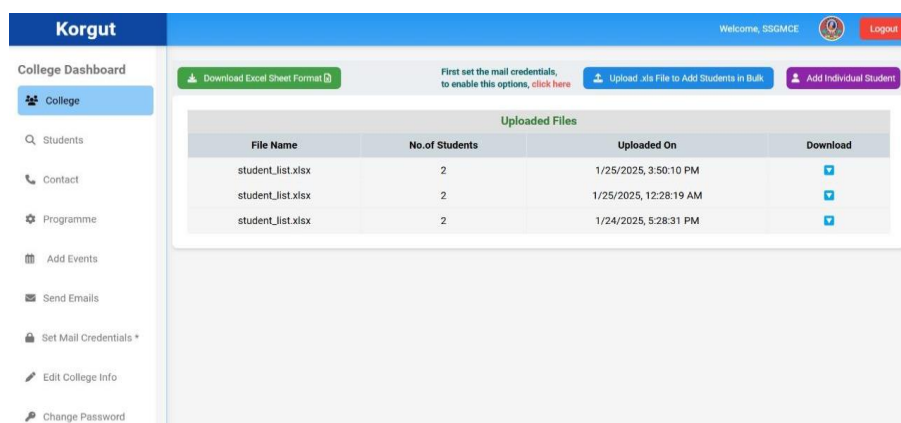


Screenshot 5.1 Hire Students as per Job Profile Screen

5.2 INTEGRATION AND COMPATIBILITY: Korgut has been designed with scalable and modular architecture to provide seamless interoperability between all the concerned parties—students, faculty members, recruiters, and college administrators. Microservices allow concerns to be separated and minimize dependencies among modules. Portfolio management, resume building, and job applications are managed by the student module. Faculty module checks student information, and recruiter module provides talent identification, candidate shortlisting, and communication.

The application presents a thoroughly documented RESTful API developed using the Django REST Framework. The APIs are secure, high-performance, and highly flexible. They support effortless data retrieval, creation, and updates between various client application. For educational institutions, bulk upload and update of student information is a breakthrough. Colleges utilize CSV/Excel templates that are processed by the backend using Pandas for data normalization, validation, and entry into PostgreSQL. This eliminates manual intervention and enables academic dashboards to represent real-time information, such as department-wise placement percentages, average CGPAs, and skill distribution.

API security is provided by OAuth 2.0 tokens, role-based access control (RBAC), and HTTPS communication. These security measures provide protection to sensitive information such that it can be accessed only by users who have proper permissions, enabling the platform to stay compliant with data protection laws.



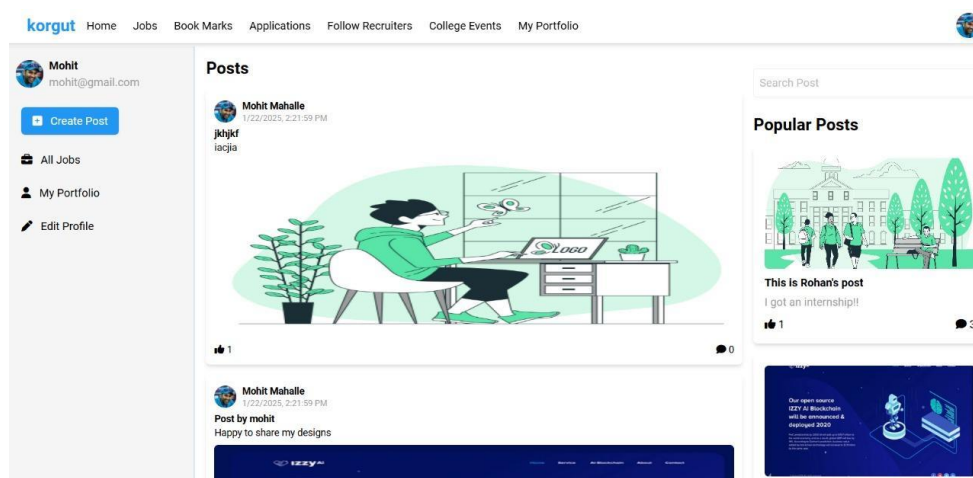
File Name	No. of Students	Uploaded On	Download
student_list.xlsx	2	1/25/2025, 3:50:10 PM	Download
student_list.xlsx	2	1/25/2025, 12:28:19 AM	Download
student_list.xlsx	2	1/24/2025, 5:28:31 PM	Download

Screenshot 5.2 Add Students in bulk using CSV file Upload

5.3 USER EXPERIENCE AND SATISFACTION: Korgut focuses on user-oriented design and transparency of operations. The platform is based on a cutting-edge tech stack that comprises React.js for frontend and Django for backend, which provides a seamless and responsive user interface for all user segments. Students are enriched with the following features:

1. Dynamic resume creation through Django templates.
2. Event tracking and registration through a separate dashboard.
3. Real-time status monitoring of applications
4. Job bookmarking and alert notifications.

It is the responsibility of faculty members to affirm student information. The faculty members view their departmental student records via secure login, authenticate uploaded documents, and make academic updates. The participation of faculty members minimizes the danger of fraudulent applications as well as enhances the integrity of candidate profiles.



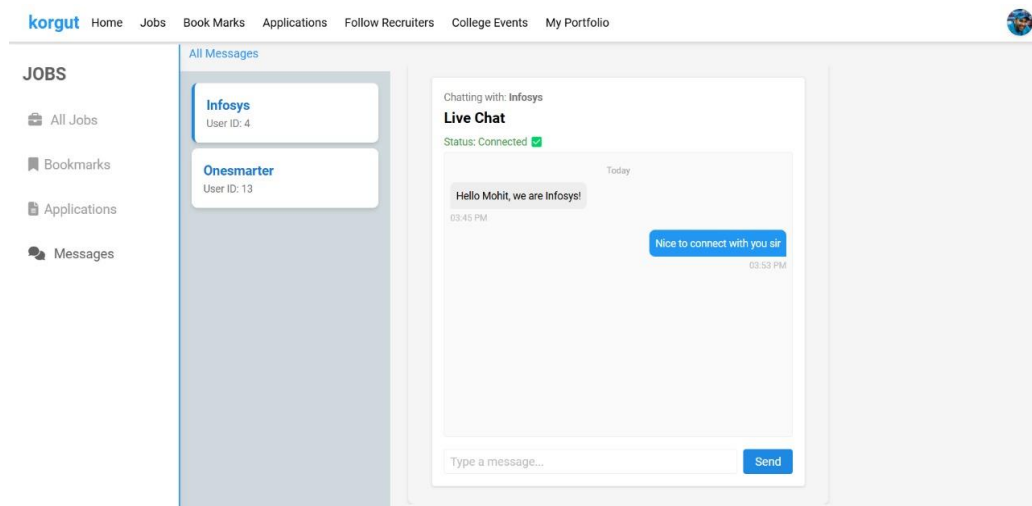
Screenshot 5.3 Student Login Home Screen

5.4 REAL-TIME CHAT MODULE

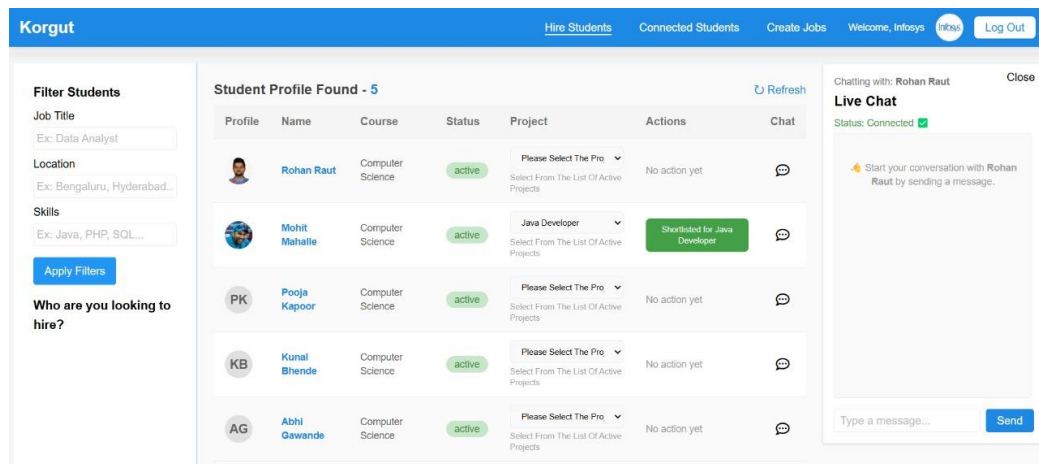
Korgut also has a feature of real-time chat that allows immediate messaging between students and recruiters, making the process of communication smoother. Through this feature, students can quickly clarify doubts, follow up on submissions, and establish professional relationships. For recruiters, it makes candidate interaction easier through instant response, interview scheduling, and individualized interaction.

Technically, the chat is developed with Django Channels and WebSockets for real-time updates, secure data processing with PostgreSQL, and background task management with Redis and Celery. The feature seamlessly integrates with the React frontend, providing a seamless messaging experience.

This channel of communication has increased user satisfaction and improved the speed and transparency of the recruitment process, and hence it is an integral part of the Korgut platform.



Screenshot 5.4 Chat interface at student side



Screenshot 5.5 Chat interface at recruiter side

5.5 FUTURE ENHANCEMENT AND CONSIDERATION

While Korgut offers a robust feature set and has achieved high adoption rates during initial pilots, several enhancements are envisioned:

AI-Assisted Job Matching: Leveraging machine learning to suggest jobs based on user behavior, skills, and recruiter preferences.

Automated Faculty Review: OCR and document recognition could aid in partially automating certificate validation workflows.

Real-Time Analytics: Dashboards with data visualizations (using Recharts or Chart.js) to help faculty and recruiters track placement trends and skill gaps.

Expanded Integrations: Inclusion of third-party job platforms like Handshake, Naukri, or Indeed for multi-channel job distribution.

Mobile App Development: A cross-platform mobile version for students and recruiters to enhance accessibility.

CHAPTER 6
CONCLUSION

6.1 CONCLUSION

The creation of Korgut marks a meaningful shift in how campus recruitment and academic coordination can work in the digital age. Instead of fragmented processes and outdated systems, Korgut brings together students, faculty, and recruiters into one well-integrated platform. It tackles persistent issues head-on—things like unverified student data, poor communication between stakeholders, and the absence of a central, reliable system. By taking a modular and thoughtfully designed approach, the platform manages to be secure, user-friendly, and ready to grow—handling everything from resume generation and portfolio verification to real-time job applications and recruiter engagement.

What truly sets Korgut apart is how it strikes a balance between automation and human oversight. Faculty play a key role by verifying student achievements, certifications, and project work before these details are shared with recruiters. This process not only builds trust but also aligns the system with academic credibility. Unlike platforms that rely purely on AI to make decisions, Korgut allows human judgment to play a central role. At the same time, the system is built with the future in mind—ready to incorporate smart features like AI-powered job matching and intelligent shortlisting as it evolves.

On the technical side, Korgut’s architecture is built for scale and efficiency. It takes inspiration from microservices, with each module—whether for students, faculty, recruiters, or admins—designed to work independently but cohesively through a RESTful API layer. The frontend, developed in React.js, is smooth, responsive, and intuitive. On the backend, Django REST Framework ensures secure and consistent data management. PostgreSQL handles complex relationships in the database, supporting real-world academic and recruitment scenarios. Bulk uploads, role-based access controls, and fine-grained permissions all contribute to a system that’s both powerful and flexible. Plus, with tools like SendGrid, Twilio, and OAuth 2.0 integration, the platform offers seamless communication and secure user authentication.

When it comes to real-world use, Korgut keeps the user at the center. Students can apply for jobs, build resumes, interact via a community feed, and stay on top of application

statuses—all from one dashboard. Recruiters benefit from a refined search and filtering experience, allowing them to discover the best talent faster, with added features for premium users. Faculty members can easily verify academic records, manage student data, and even chat with recruiters via secure messaging. Across the board, Korgut is built for clarity, efficiency, and low learning curves. Feedback from early testers shows high satisfaction—especially around the system’s responsiveness and the reliability of the data they’re seeing.

Looking ahead, there’s a lot of potential for Korgut to grow and innovate further. The most exciting next steps include adding AI-driven recommendations to help recruiters and students find better matches, and possibly even introducing blockchain-based credential verification to make academic data portable and tamper-proof. Other upcoming features include automated profile checks, real-time recruitment analytics, and integrations with third-party job boards and educational platforms. The current foundation is strong and adaptable, so these upgrades can be introduced smoothly, without disrupting what already works well.

In the end, Korgut isn’t just another tech product—it’s a full ecosystem built to empower everyone involved in the campus recruitment journey. It’s designed to meet real needs with real solutions, giving students a better shot at their future, faculty a smarter way to support their departments, and recruiters a trusted space to find the right talent. And as the platform continues to learn, evolve, and adapt, it’s well on its way to setting a new standard for digital recruitment in academia—and beyond.

REFERENCES

REFERENCES

- [1]. G. H. A. Khan, Gangeshwari, Dronesh, and S. Singh, "Online Job Searching and Recruitment System-A Career Site," *International Research Journal of Engineering and Technology (IRJET)*, vol. 8, no. 5, pp. 3661–3664, May 2021.
- [2]. A. Shelar, S. Sawant, A. Pacharne, S. Tike, and D. M. Mane, "College Management System: A Technical Paper," *International Journal of Scientific Research in Science and Technology (IJSRST)*, vol. 10, no. 3, pp. 553–561, May-Jun. 2023. [Online]. Available: <https://ijsrst.com/home/issue/view/article.php?id=IJSRST523103120>
- [3]. H. Rahmani, W. Groot, and A. M. Rahmani, "A Predictive Analytics Solution Matching Job Seekers' Talent and Employers' Demands Based on Machine Learning," *Research Square*, preprint, Jul. 2023. [Online]. Available: <https://doi.org/10.21203/rs.3.rs-3104276/v1>
- [4]. V. S. Pendyala, N. Atrey, T. Aggarwal, and S. Goyal, "Enhanced Algorithmic Job Matching Based on a Comprehensive Candidate Profile Using NLP and Machine Learning," in *Proceedings of the Eighth IEEE International Conference on Big Data Computing Service and Applications (BigDataService)*, Newark, CA, USA, Aug. 2022, pp. 183–184. [Online]. Available: <https://ieeexplore.ieee.org/document/9898268>
- [5]. G. Dhanalakshmi, P. Daphne Patricia, A. Anu Sowmiyaa, and K. K. Aruna Rajeswari, "Online Job Portal Using Django," *Innovare Journal of Engineering and Technology*, vol. 11, no. 2, pp. 1–3, 2023. [Online]. Available: <https://innovareacademics.in/journals/index.php/ijet/article/view/47235>
- [6]. J. H. Priyanka and N. Parveen, "Online Employment Portal Architecture Based on Expert System," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 25, no. 3, pp. 1731–1735, Mar. 2022. [Online]. Available: <https://ijeecs.iaescore.com/index.php/IJEECS/article/view/27079/0>
- [7]. S. J. Alharbi and T. Moulahi, "API Security Testing: The Challenges of Security Testing for Restful APIs," *International Journal of Innovative Science and Research Technology*, vol. 8, no. 5, pp. 1485–1492, May 2023. [Online]. Available: <https://ijisrt.com/api-security-testing-the-challenges-of-security-testing-for-restful-api>
- [8]. R. Jain, A. Modi, and I. Kashyap, "Review on College Management System," *American Journal of Computer Science and Information Technology*, vol. 11, no. 7, pp. 1–3, 2023. [Online]. Available: <https://www.imedpub.com/articles/review-on-college-management-system.php?aid=51739>

[9]. R. Narula, V. Kumar, R. Arora, and R. Bhatia, "Enhancing Job Recommendations Using NLP and Machine Learning Techniques," *International Journal of Computer Science and Information Technology*, vol. 15, no. 3, pp. 45–53, 2023. [Online]. Available:

https://www.researchgate.net/publication/377442387_Enhancing_Job_Recommendations_Using_NLP_and_Machine_Learning_Techniques

[10]. S. P. S. Rathore, "The Impact of AI on Recruitment and Selection Processes: Analysing the Role of AI in Automating and Enhancing Recruitment and Selection Procedures," *International Journal for Global Academic & Scientific Research*, vol. 2, no. 2, pp. 78–93, 2023. [Online]. Available: <https://journals.icapsr.com/index.php/ijgasr/article/view/50>



*International Research Journal Of Modernization
in Engineering Technology and Science*

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 03/70300252853

Date: 06/04/2025

Certificate of Publication

This is to certify that author "**Kunal Bhende**" with paper ID "**IRJMETS70300252853**" has published a paper entitled "**KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS**" in *International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS)*, Volume 07, Issue 03, March 2025

A. Dhanraj

Editor in Chief



We Wish For Your Better Future
www.irjmets.com





*International Research Journal Of Modernization
in Engineering Technology and Science*

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 03/70300252853

Date: 06/04/2025

Certificate of Publication

This is to certify that author "**Prajwal Ghusalikar**" with paper ID "**IRJMETS70300252853**" has published a paper entitled "**KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS**" in *International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS)*, Volume 07, Issue 03, March 2025

A. D. Desai

Editor in Chief



We Wish For Your Better Future
www.irjmets.com





***International Research Journal Of Modernization
in Engineering Technology and Science***

(Peer-Reviewed, Open Access, Fully Refereed International Journal)



e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 03/70300252853

Date: 06/04/2025

Certificate of Publication

This is to certify that author "**Chetan Chaudhary**" with paper ID "**IRJMETS70300252853**" has published a paper entitled "**KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS**" in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS)**, Volume 07, Issue 03, March 2025

A. Demish

Editor in Chief



We Wish For Your Better Future
www.irjmets.com





***International Research Journal Of Modernization
in Engineering Technology and Science***

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 03/70300252853

Date: 06/04/2025

Certificate of Publication

This is to certify that author "Devashish Ugale" with paper ID "IRJMETS70300252853" has published a paper entitled "KORGUT: A COMPREHENSIVE PLATFORM FOR COLLEGES, STUDENTS, AND RECRUITERS" in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 03, March 2025

A. Demati

Editor in Chief



We Wish For Your Better Future
www.irjmets.com



PLAGARISM REPORT



Page 2 of 9 - Integrity Overview

Submission ID trn:oid::26066:453858053

17% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

- 12 Not Cited or Quoted 17%**
Matches with neither in-text citation nor quotation marks
- 0 Missing Quotations 0%**
Matches that are still very similar to source material
- 0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 16% Internet sources
- 13% Publications
- 17% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.



Page 3 of 9 - Integrity Overview

Submission ID trn:oid::26066:453858053

Match Groups

- 12 Not Cited or Quoted 17%**
Matches with neither in-text citation nor quotation marks
- 0 Missing Quotations 0%**
Matches that are still very similar to source material
- 0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 16% Internet sources
- 13% Publications
- 17% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works		
Trevecca Nazarene University (Nashville) on 2025-03-10			8%
2	Internet		
www.irjmets.com			6%
3	Submitted works		
University of Glasgow on 2023-08-11			2%
4	Internet		
ssgmjournal.in			1%
5	Internet		
damunosor.github.io			<1%

PROJECT GROUP MEMBERS

Name: Prajwal N. Ghusalikar

Address: Chandur Railway, Amravati, Maharashtra

Email: prajwalghusalikar@gmail.com

Mobile no: +91 8788583509



Name: Kunal T. Bhende

Address: Hinganghat, Wardha, Maharashtra

Email: kunalbhende895@gmail.com

Mobile no: +91 8080994426



Name: Chetan Chaudhary

Address: Hinganghat, Wardha, Maharashtra

Email: chetanchaudharyxyz@gmail.com

Mobile no: +91 74980 44129



Name: Devashish Ugale

Address: Malkapur, Maharashtra

Email: devashishugale2019@gmail.com

Mobile no: +91 9575801947

