International Organization of Research & Development (IORD)
ISSN: 2348-0831
Vol 13 Special Issue 02 | 2026

HireHub: A Flask-Based Job Portal For Efficient Job Seeker-Recruiter Matching

Eeshwari P. Gawande[1], Mansi J. More[2], Ruchi R. Kamatkar[3], Mayuri R. Rathod[4]
[1,2,3,4]*Computer Science & Engineering, Siddhivinayak Technical Campus Shegaon, Maharashtra, India*

DOI: 10.5281/zenodo.19539173

ABSTRACT

The rapid evolution of technology and the competitive job market have created a need for an automated placement system to replace traditional manual processes that cause inefficiencies, delays and miscommunication. This paper presents 'Hire hub', a web-based smart hiring and placement system designed to automate job posting, student eligibility verification, recruiter interaction and placement tracking. The system is built as a mini-ERP with three modules: Admin, Student and Recruiter, providing centralized job postings, profile management, application tracking and interview scheduling. Developed using Python-Flask for the backend, MySQL for database management and HTML/CSS/Bootstrap for the frontend, the system aims to reduce manual effort, improve transparency and generate accurate placement reports. The implementation demonstrates effective modular design and database integration for streamlined placement automation in computer engineering environments. Hire hub enhances the overall placement experience, benefiting students, recruiters and administrators alike, and serves as a scalable model for educational institutions seeking to modernize their placement processes.

Keywords: - Placement management system, Job portal, Recruitment automation, Student placement, Web application, Flask, MySQL, Python, HTML/CSS, Bootstrap, Placement cell, Campus recruitment, Student profile management, Recruiter portal

1. INTRODUCTION

The job market is getting tougher, and traditional placement processes are becoming outdated. Manual work, delays, and miscommunication are common issues. Hire hub is a web-based solution that automates job posting, student eligibility checks, recruiter interactions, and placement tracking. It's designed to make life easier for students, recruiters, and admins.

Hire hub has three main modules: Admin, Student, and Recruiter. Admins manage everything, students can apply for jobs and track progress, and recruiters can post jobs and shortlist candidates. The system uses Flask, MySQL, and HTML/CSS/Bootstrap, making it a robust and user-friendly platform.

1.1 Background of Hire Hub

The traditional placement process is plagued by inefficiencies. Manual job posting, student eligibility checks, and recruiter interactions lead to delays and miscommunication. Students struggle to find relevant job openings, while recruiters face challenges in identifying top talent. Existing systems lack automation, transparency, and scalability, highlighting the need for a smart hiring solution like Hire hub.

1.2 Problem Statement

- Inefficient placement processes due to manual job posting and student eligibility checks.
- Lack of transparency in job openings and application status.
- Difficulty for recruiters to identify top talent.
- Limited scalability of traditional placement systems.
- Poor user experience for students, recruiters, and admins.
- Data inconsistencies and wasted resources due to outdated methods.

1.3 Research Objective

The main objective of this research is to design and develop a web-based this System, the specific objectives are:

- Automate placement processes to reduce manual effort.
- Provide centralized job postings accessible to eligible students.
- Simplify recruiter interaction with candidates.
- Improve user experience for students, recruiters, and admins.
- Develop a scalable and secure placement system.

1.4 Scope of the Study

The scope of the study encompasses the development of Hire hub, a web-based placement management system, focusing on automating key processes like job posting, student eligibility checks, and recruiter interactions. It includes implementing three core modules – student, recruiter, and admin – and integrating with a MySQL database for efficient data management. The system's performance and usability will be thoroughly tested and evaluated to ensure a seamless experience.

2. LITERATURE REVIEW

Various studies highlight the limitations of traditional placement systems. Manual processes lead to inefficiencies, delays, and poor communication (Kumar et al., 2020). Existing systems often lack automation, transparency, and scalability, making it tough for students to find jobs and recruiters to find talent (Patel & Jain, 2019).

Research suggests web-based platforms can improve placement processes. Systems like these can automate tasks, enhance transparency, and provide better user experiences (Sharma & Gupta, 2021). Hire hub builds on this, using Flask and MySQL to create a user-centric placement solution.

2.1Existing Methods

- Manual job posting and student eligibility checks.
- Traditional placement cells with limited automation.
- Separate systems for job postings, student profiles, and recruiter interactions.

2.2 Limitations of Existing Solutions

- Inefficient processes lead to delays and miscommunication.
- Lack of transparency in job openings and application status.
- Difficulty for recruiters to identify top talent.
- Limited scalability and poor user experience.

2.3 Need for Proposed System

1. Automating the placement process help eliminate repetitive manual tasks such as data entry, and the resume screening, and interview scheduling. This increases efficiency, save time , and reduces human errors.

2. The centralized platform allows all job posting to be managed and viewed in one place. This ensures transparency by giving students and recruiter equal access to update and accurate information.

3. The system streamlines communication between recruiter and candidates through notification, messaging, and interview updates. This makes the hiring process smoother and more organized.

4. A well-designed system offers a user-friendly interface that is easy to navigate. It can also handle increasing numbers of users and data without affecting performance.

3. METHODOLOGY

The methodology involves developing Hire hub using a web-based approach with Flask (Python), MySQL, and HTML/CSS/Bootstrap. The system is designed to automate placement processes, provide centralized job postings, and simplify recruiter interactions. It includes three core modules: Admin, Student, and Recruiter. The development process involves feasibility study, requirements analysis, system design using ERDs and DFDs, implementation, and testing. Tools and technologies used include Flask, MySQL, and PyCharm/VS Code.

3.1 Development Approach

The development approach for Hire hub involves a structured process starting with a feasibility study to determine project viability, followed by requirements analysis to gather functional and non-functional needs. The system is then designed using ERDs, DFDs, and UML diagrams, and implemented using Flask, MySQL, and HTML/CSS/Bootstrap. The development phase is followed by thorough testing, including unit, integration, and system testing, before deploying the system on a local server (XAMPP) for evaluation and refinement.

The development process includes requirement analysis, system design, database creation, frontend development, backend integration, and system testing.

Table -1: Methodology in Table Form

Methodology Step	Description
Research Type	Development-based research
Data Source	User input data (student profiles, job posting, recruiter details)
Analysis Method	Functional testing and system validation
Tools Used	Flask, MySQL, HTML, CSS, Python, Bootstrap
Output	Web-Based Hire hub system with job posting, candidate management, and placement report features
How Done in Paper	System design, module development, database integration, authentication setup, and testing

3.2 Development Approach

The Hire hub system is developed using a web-based development approach. The application follows a modular structure where different components such as authentication, job management, student profiles, recruiter interactions, and placement reports are handled separately. This approach improves system organization and makes maintenance easier.

The development process includes requirement analysis, system design, database creation, frontend development, backend integration, and system testing.

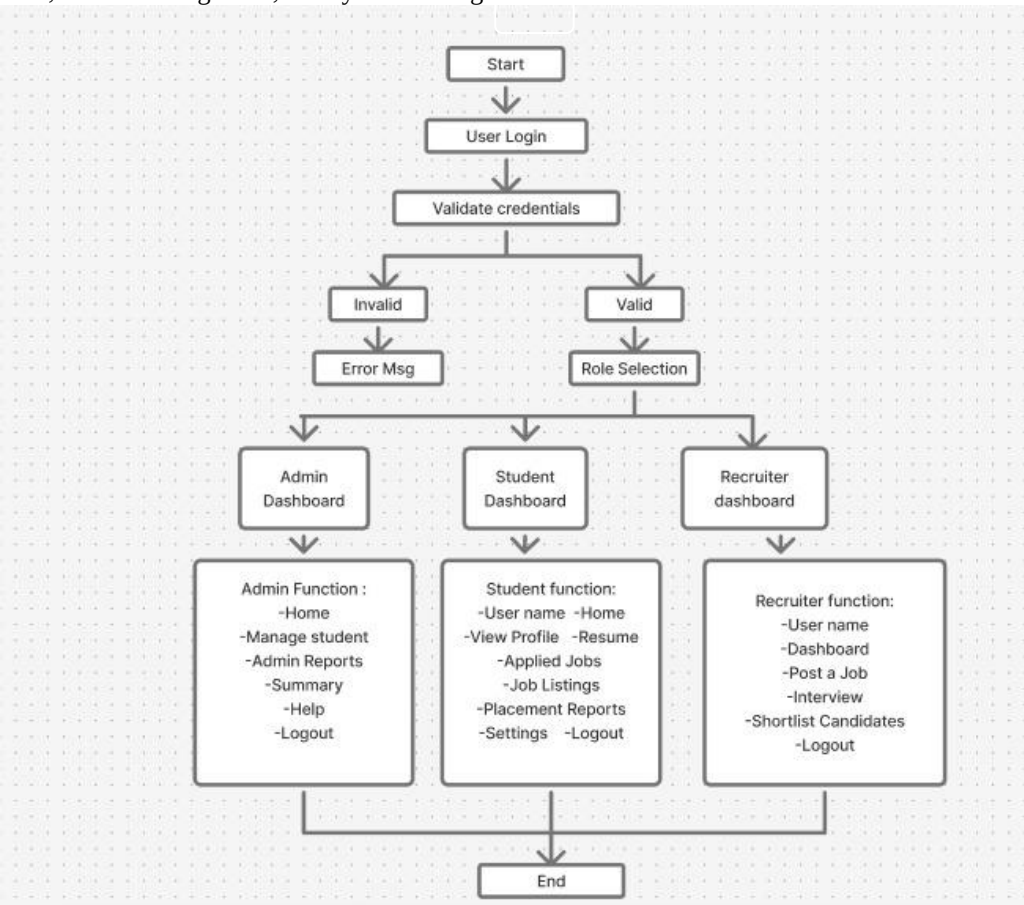


Fig-2: Flowchart – System Architecture

Algorithm – System Architecture:

Step 1: User Login
1. User enters credentials
2. System checks credentials against database
3. If valid, redirect to respective module

Step 2: Admin Module
1. Admin logs in
2. Admin dashboard displayed
3. Admin performs operations (e.g., manage students, manage recruiters, etc.)
4. System updates database accordingly

Step 3: Student Module
1. Student logs in
2. Student dashboard displayed
3. Student applies for job
4. System checks eligibility and updates database
5. Student tracks status

Step 4: Recruiter Module
1. Recruiter logs in
2. Recruiter dashboard displayed
3. Recruiter posts job

4. System updates database
5. Recruiter shortlists candidates
6. System notifies students

3.3 Implementation Modules

Here's a breakdown of the implementation modules for Hire hub:

1. Admin Module
- Manage students (manage everything about student report)
- Manage recruiters (manage everything about recruiter report)
- Manage job postings (manage all job posting)

2. Student Module
- Registration and login
- Profile management (update, view)
- Job application and tracking
- Resume upload and management

3. Recruiter Module
- Registration and login
- Job posting and management
- Candidate shortlisting and scheduling interviews
- Communication with students

4. Database Module
- MySQL database design and implementation
- Tables: students, recruiters, jobs, applications

5. Frontend Module
- User interface design (HTML, CSS, Bootstrap)
- Responsive design for accessibility

6. Backend Module
- Flask (Python) implementation
- API integration for database interactions
- Authentication and authorization

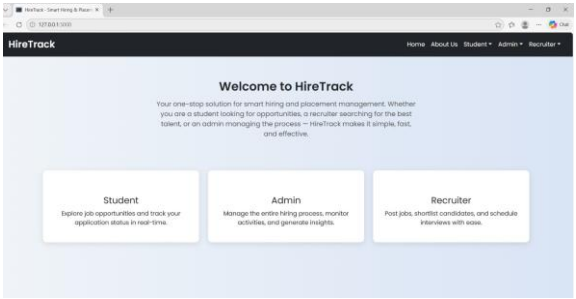
3.4 System Testing

System testing ensures Hire hub functions as expected, meeting requirements and identifying bugs.

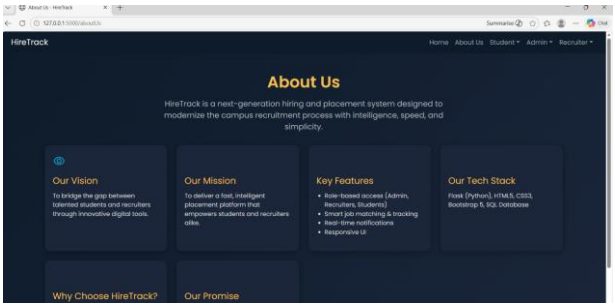
Testing Types:

1. Unit Testing: Verify individual components (e.g., login, job posting)
2. Integration Testing: Test interactions between modules (e.g., student applying for job)
3. System Testing: Validate overall system functionality and performance
4. User Acceptance Testing (UAT): Ensure system meets user expectations

3.5 Result



Fig(1): Home Page



Fig(2): About Page

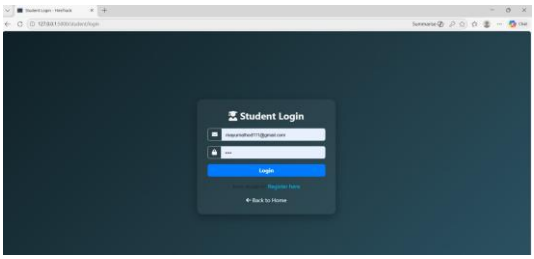


Fig (3): Student Login

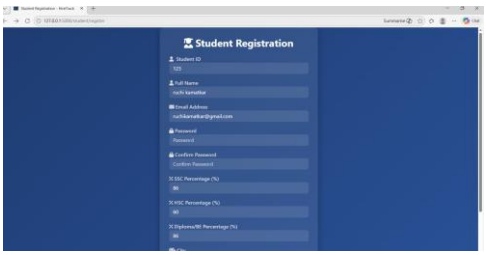


Fig (4): Student Registration

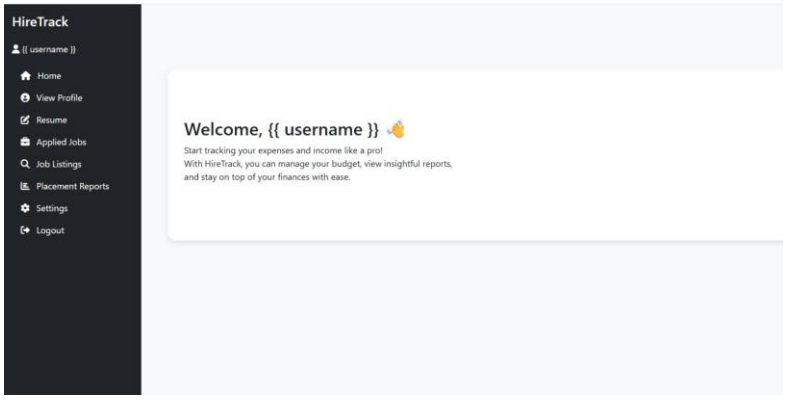


Fig (5): Student Dashboard

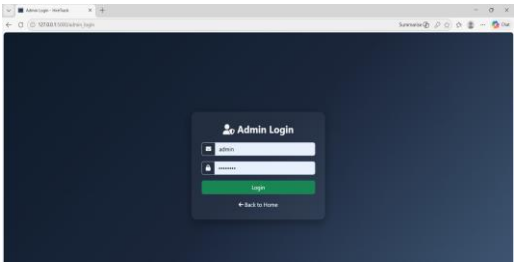


Fig (6): Admin Login

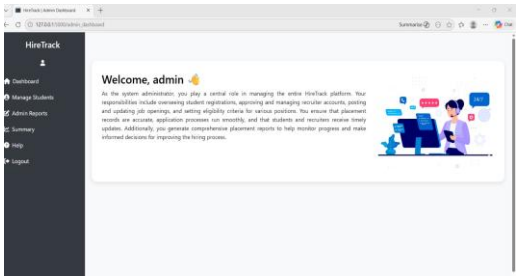


Fig (7): Admin Dashboard

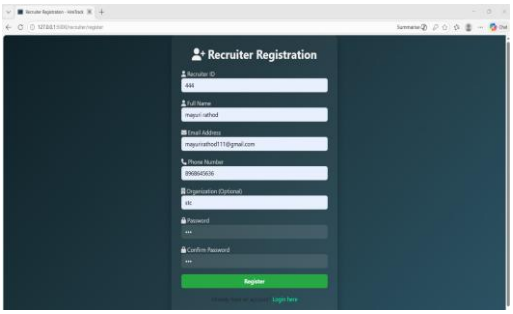


Fig (7): Recruiter Registration

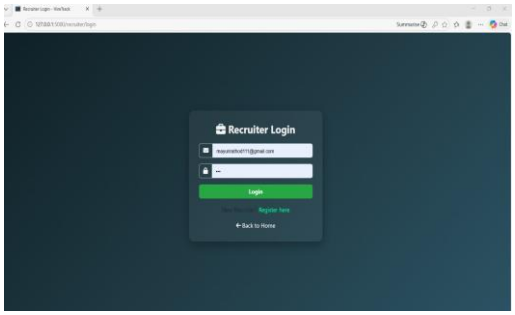


Fig (8): Recruiter Login

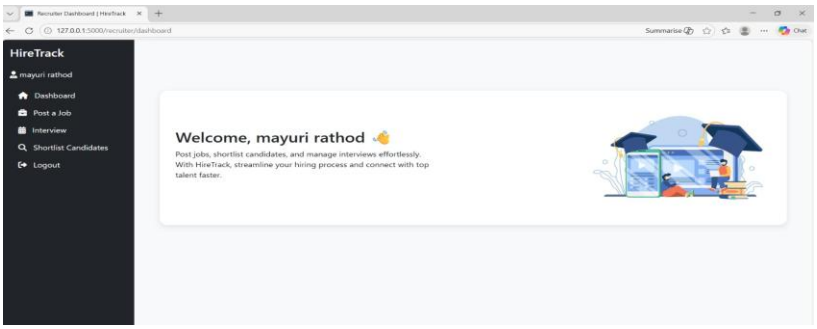


Fig (9): Recruiter Dashboard

Test Scenarios:
1. Admin login and dashboard functionality
2. Student registration and profile management
3. Recruiter job posting and candidate shortlisting
4. Job application and tracking
5. Report generation

Testing Tools:
1. Manual testing
2. Automated testing (e.g., Selenium)

4. CONCLUSIONS

In conclusion, Hire Hub makes the placement process easier and more organized for everyone involved. It brings students, recruiters, and administrators onto one common platform where job postings, communication, and reports can be managed smoothly. By reducing manual work and improving transparency, the system saves time and increases efficiency. In the future, adding features like AI-based candidate matching and a mobile app could make Hire Hub even more powerful and user-friendly.

5. ACKNOWLEDGEMENT

The authors would like to sincerely thank Prof. Raut Mam for her constant guidance, encouragement, and valuable suggestion throughout the development of the hire hub system. He supports and supervision played a significant role in the successful completion of the research work. The team also extended support required to implement and test the project successfully.

6. REFERENCES

[1]. M. Grinberg, Flask Web Development: Developing Web Applications with Python, O'Reilly Media, 2018.
[2]. P. DuBois, MySQL Cookbook, O'Reilly Media, 2014.
[3]. A. Ron, Python Crash Course, No Starch Press, 2019.
[4]. W. Richard, Bootstrap 4 Cookbook, Packet Publishing, 2017.
[5]. R. Nixon, Learning Web Design and Development, O'Reilly Media, 2018.
[6]. 10. S. Holzner, Python Programming for Beginners, McGraw-Hill, 2020.
[7]. Bootstrap. (n.d.). Bootstrap v5.3 Documentation. Retrieved from https://getbootstrap.com
[8] DuBois, P. (2014). MySQL Cookbook. O'Reilly Media.
[9] Grinberg, M. (2018). Flask Web Development. O'Reilly Media.Ronacher, A. (2010). Flask Documentation. Pallets Projects.
[10] Silberschatz, A., Korth, H., & Sudarshan, S. (2019). Database System Concepts. McGraw Hill.