



# International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

*(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)*



**Impact Factor: 8.699**

**Volume 14, Issue 4, April 2025**

# **Job Portal and Candidate Prediction using MERN Stack Application and Machine Learning Model**

**Dr. D. B. Kshirsagar, Rohan Kumatkar, Pratik Mule, Mayank Rane, Sanika Thorat, Akshada Wavhal**

Assistant Professor, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon,  
Maharashtra, India

Student, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon, Maharashtra, India

Student, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon, Maharashtra, India

Student, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon, Maharashtra, India

Student, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon, Maharashtra, India

Student, Department of Computer, SRES'S Sanjivani College of Engineering Kopergaon, Maharashtra, India

**ABSTRACT:** This study introduces the design and implementation of an online job portal built using the MERN stack (MongoDB, Express.js, React.js, Node.js), enhanced with machine learning techniques to predict suitable candidates for job roles. The platform streamlines the recruitment process by automating candidate screening, offering job recommendations, and increasing hiring efficiency. The Candidate Prediction Model uses machine learning to assess key factors like skills, experience, and education, ranking candidates according to their job fit. Job seekers can create profiles, search and apply for jobs, while recruiters can post vacancies, review applications, and access candidate recommendations. By incorporating predictive analytics, the system reduces hiring time, improves decision-making, and optimizes recruitment operations. The use of modern web technologies and machine learning ensures a scalable, user-friendly, and intelligent job-matching experience.

**KEYWORDS:** - 1. MERN Stack (MongoDB, Express.js, React.js, Node.js) 2. Machine Learning (ML) 3. Candidate Prediction 4. Machine Learning-driven Job Matching 5. Text Processing with Natural Language Processing (NLP)

## **I. INTRODUCTION**

The recruitment process is evolving rapidly, driven by the need for faster and more efficient hiring solutions. Traditional methods still depend on manual resume screening, subjective judgments, and ineffective matching, which can lead to poor hiring decisions, delays, and missed opportunities. As digital transformation accelerates, there is an urgent need for intelligent systems to optimize the recruitment process.

This research introduces the development of an ML-based job portal built on the MERN stack (MongoDB, Express.js, React.js, Node.js), integrated with machine learning and natural language processing (NLP) techniques. The system uses modern technologies to automate resume analysis, predict candidate success, and enhance job-candidate matching based on key factors like skills, qualifications, and experience.

The platform allows job seekers to create profiles, upload resumes, and receive tailored job recommendations, while recruiters can efficiently manage job listings, screen applicants, and view ML-generated candidate rankings. By combining the MERN stack's scalability with intelligent analytics, the system improves decision-making and reduces hiring time.

This research examines various machine learning methods to improve recruitment accuracy and efficiency:

- **Hire Prediction System:** Uses supervised learning models to predict a candidate's likelihood of being hired.
- **Suitability Measurement:** Implements cosine similarity to compare job descriptions with candidate resumes for skill-based matching.
- **Job-Candidate Classification and Ranking:** Uses cosine similarity to classify and rank resumes based on semantic relevance to job postings.



Through this integrated approach, the platform aims to deliver a more effective, unbiased, and scalable recruitment solution—benefiting organizations by improving hiring accuracy, and aiding job seekers by matching them with the right opportunities.

## **II. LITERATURE SURVEY**

### **1) Hire Prediction System Using Machine Learning: [1]**

This research focuses on developing a Windows-based application that predicts the likelihood of college students being hired or not, based on specific input parameters. The initial stage of the study involves gathering relevant data and designing an intuitive graphical user interface (GUI) for ease of use. In the final stage, machine learning algorithms are employed to analyze the data and determine the probability of a student securing a job offer.

### **2)Candidate Engagement Success Prediction Using Machine Learning and Natural Language Processing Techniques:[2]**

The uniqueness of this approach lies in the integration of diverse data types—structured information, textual content, and voice inputs—to build a more comprehensive candidate profile, thereby enhancing the accuracy of job success predictions. During the data preprocessing stage, beyond conventional techniques, the BERT model is utilized to convert textual data into machine-readable embeddings compatible with the chosen prediction algorithm. Following preprocessing and feature engineering, an XGBoost classifier is trained to perform binary classification, determining whether a candidate is likely to succeed or not.

### **3)AI-based suitability measurement and prediction between job description and job seeker profiles :[3]**

Selecting the right candidate for a job role is often a complex and demanding task, involving multiple detailed evaluations. Organizations frequently encounter difficulties in identifying candidates who meet the specific criteria outlined in the Job Description (JD). To address this, an AI-driven system has been designed to assess and predict the most appropriate candidates from a database of Candidate Resumes (CR). The system organizes information from both JD and CR into four distinct clusters—primary skills, secondary skills, descriptive adjectives, and adverbs. It then calculates the Jaccard similarity between these clusters to determine a suitability score for each candidate.

### **4)Job-Candidate Classifying and Ranking System-Based Machine Learning Method :[4]**

Identifying the most appropriate candidates for a job vacancy can become a repetitive and time-intensive process, particularly when dealing with a large number of applicants. Additionally, ensuring a fair and efficient shortlisting process can be challenging, and delays or human errors during screening may result in missing out on highly qualified candidates. This study introduces a machine learning-based approach to support human resource departments in efficiently classifying and selecting the best-fit candidates. The proposed model is designed to categorize applicants into two groups: (i) shortlisted and (ii) not suitable, thereby streamlining the recruitment workflow.

## **III. PROPOSED METHODOLOGY**

### **PROBLEM STATEMENT**

The traditional recruitment process is often time-consuming, biased, and inefficient, resulting in poor job-candidate matches, delayed hiring cycles, and increased operational costs. Recruiters struggle to sift through large volumes of resumes, while job seekers face challenges in finding roles that align with their skills and aspirations. There is a critical need for a smart, scalable, and data-driven solution that automates and optimizes the recruitment process for both employers and job seekers.

This paper addresses these challenges by developing an intelligent job portal using the MERN stack, integrated with machine learning and natural language processing (NLP). The proposed platform aims to enhance recruitment efficiency by automating candidate shortlisting, enabling ML-driven job matching, and providing actionable analytics and predictive insights—thereby improving the accuracy, speed.

## **IV. METHODOLOGY**

The Job Portal and Candidate Prediction System follows a structured and iterative approach to ensure efficiency, scalability, and accuracy in candidate-job matching. The methodology includes data collection, system design,

development, testing, and deployment in multiple phases.

#### 1. Research and Requirement Analysis :

- Conduct a literature review on ML-based candidate prediction.
- Define functional and non-functional requirements.

#### 2. System Development Approach (Incremental Model) :

- Requirement Gathering: Understand job seeker and recruiter needs.
- System Design: MERN stack architecture, AI-based recommendation model.

#### Implementation:

- ♣ **Frontend:** HTML, CSS, Javascript, React.js
- ♣ **Backend:** Node.js, Express.js
- ♣ **Database:** MongoDB
- ♣ **ML-Model:** Python (TF-IDF Vectorizer, Cosine Similarity, Scikit-Learn)
- ♣ **Testing:** Functional, performance, and security testing.

Technology/Library	Contribution Score (0–10)
Flask	8
HTML (Jinja2 templates)	6
docx2txt	5
PyPDF2	5
TfidfVectorizer	9
cosine_similarity	9
os	4

Fig. III.1. Comparative Contribution of technologies in ML table

The comparative contribution graph visually represents the relative importance of each technology or library integrated into the Resume Matcher system. These contribution scores (ranging from 0 to 10) are based on two main criteria: functional impact on the system's logic and frequency of use during the resume-job matching process.

#### Breakdown of Technology Contributions:

##### 1. TfidfVectorizer (9/10)

This stage plays a vital role in the system, as it converts textual information—such as resumes and job descriptions—into numerical feature vectors using Term Frequency-Inverse Document Frequency (TF-IDF), enabling effective analysis by machine learning models. It enables the algorithm to capture the significance of terms relative to both documents, making it central to text analysis and comparison.

##### 2. Cosine\_similarity (9/10)

Used in conjunction with TfidfVectorizer, cosine similarity computes the angle between vector representations of resume and job description. It quantifies their similarity, effectively producing the match or “accuracy” score. Without this function, scoring would not be feasible.

##### 3. Flask (8/10)

Flask serves as the web framework that supports user interaction. It handles routing (URL management), file uploads (resume and JD), and backend logic integration with the ML components. Flask also renders HTML templates, acting as the core of the user-facing system.

##### 4. HTML (Jinja2 templates) (6/10)

These templates are essential for the front-end structure. They define how data and forms are presented to users,

allowing dynamic rendering of results. Though not algorithmic, their usability and interface contribution are significant.

#### 5. docx2txt (5/10)

This library extracts text from .docx files, which are commonly used for resumes. While its job is singular—parsing DOCX—its role is critical for text preprocessing.

#### 6. PyPDF2 (5/10)

Similar to docx2txt, PyPDF2 is used to extract text from PDF resumes. Its utility ensures compatibility with multiple file formats, increasing the system's robustness and user-friendliness.

#### 7. os (4/10)

The os module is used for file path management, directory handling, and safe access to local resources. Though relatively simple, its inclusion ensures system stability and cross-platform support.

### 3. Machine Learning Model for Candidate Prediction :

- **Data Collection:** Job seeker profiles, resumes, job postings.
- **Preprocessing:** Data cleaning, skill extraction, categorization.
- **Training and Testing:** Model evaluation using accuracy metrics.
- **Integration:** ML-driven job recommendations and candidate ranking.

#### FLOWCHART :-

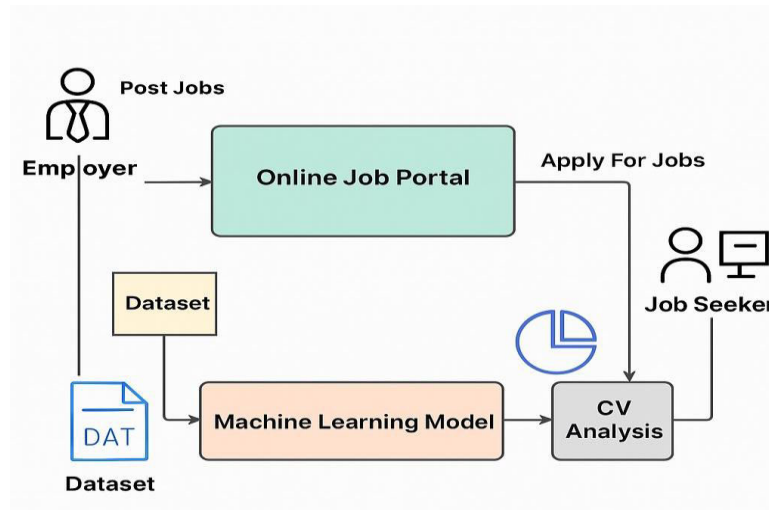


Fig III.2. Online JOB Portal Design

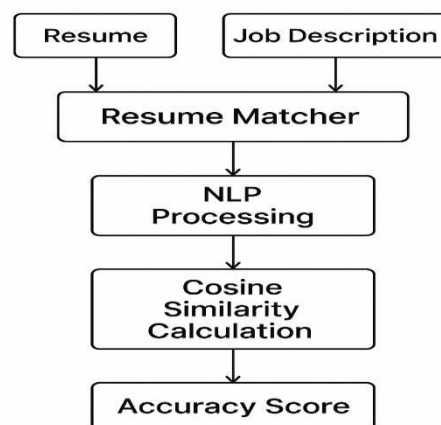


Fig.III.3. Resume Matcher With JD Design

### 4. Implementation Details: Fig. 1)

The implementation of the ML-powered job portal system begins with selecting the appropriate technology stack. The platform's frontend is built with React.js, providing a dynamic and responsive user interface. The backend utilizes Node.js and Express.js to manage API routes, handle user authentication, and facilitate communication with the database. MongoDB is employed as the primary database for storing user profiles, job postings, and application data, chosen for its scalability and adaptability. For machine learning and natural language processing tasks, the system utilizes **Python** libraries such as scikit-learn, pandas, XGBoost, and transformers (e.g., BERT) for model development, while NLP libraries like spaCy and NLTK aid in resume and job description analysis.

The system accommodates two categories of users: job seekers and employers. Job seekers are able to register, set up and update their profiles, upload resumes, and apply for available positions. Employers can post job vacancies, view

applicants, and receive AI-driven candidate recommendations. Uploaded resumes are parsed using Python-based tools like pdfminer and docx2txt, extracting structured information such as skills, experience, and education. This information is then standardized and analyzed using various machine learning models.

The platform incorporates various ML models to improve recruitment accuracy. A Candidate Prediction System assesses candidates based on their experience, education, and skill alignment to predict the likelihood of being hired. The Suitability Scoring System utilizes cosine similarity to assess the alignment between job descriptions and candidate resumes, focusing on skill-based compatibility. Additionally, the Job-Candidate Ranking model employs TF-IDF and cosine similarity to rank candidates based on their suitability for each job role. Furthermore, the system incorporates a recommendation engine that proposes job opportunities to users based on their profile details, skills, and previous application history.

Recruiters benefit from a dashboard that displays ML-ranked candidates, allowing them to filter applicants and monitor job application trends. Admins can view analytics on hiring patterns and model performance. Security protocols like JWT authentication, input sanitization, and secure file handling are implemented for data protection. The system is designed for scalability, utilizing containerized services and cloud platforms like Docker and AWS to manage large user bases and data volumes. The combination of the MERN stack and ML technologies provides a robust and intelligent recruitment platform that streamlines the hiring process and improves job-candidate matching.

### **Fig. 2)**

The architecture diagram represents a Resume Matching System using Natural Language Processing (NLP) and Cosine Similarity to assess how well a resume matches a job description. The system starts by accepting resumes in formats like .pdf or .docx and job descriptions as input. Text extraction is done using libraries such as docx2txt for Word documents and PyPDF2 or pdfminer.six for PDFs. After text extraction, the Resume Matcher module processes the documents.

NLP preprocessing steps, including tokenization, stopword removal, lowercasing, and lemmatization, are applied to standardize the text data. This processed text is then converted into numerical vectors using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. The cosine similarity algorithm is used to calculate the similarity between the resume and the job description vectors, producing a score between 0 and 1 that represents their semantic alignment. A higher score indicates a stronger match between the candidate's profile and the job requirements. This final accuracy score is used to rank candidates, helping recruiters quickly identify the most suitable applicants. The system thus streamlines hiring by intelligently automating the job-candidate matching process.

### **1. NLP Processing (Text Vectorization with TF-IDF)**

This is the first major step after text extraction.

```
vectorizer = TfidfVectorizer().fit_transform([job_description] + resumes)
```

```
vectors = vectorizer.toarray()
```

- **What it does:** TfidfVectorizer is an NLP tool that converts raw text into numerical vectors based on **word importance** in the document.
- It processes:
  - The **job description**
  - All the **uploaded resumes**
- The result is a **TF-IDF matrix**, where:
  - Each row = a document (JD or a resume)
  - Each column = a word/token
  - Each cell = a TF-IDF score (importance of that word in that document)

### **2. Similarity Calculation (Cosine Similarity Matrix) :**

After the textual content has been vectorized, the **cosine similarity** algorithm is used to measure how similar the resume is to the job description.

#### **Cosine Similarity**

- Cosine Similarity calculates the cosine of the angle between two vectors.
- Its value ranges from **-1 to 1**:

- 1 → vectors are aligned in the same direction (very similar)  
0 → vectors are orthogonal (no similarity)  
-1 → vectors are pointing in opposite directions.

The formula used is:

$$\text{Cosine Similarity} = \frac{A \cdot B}{||A|| \cdot ||B||}$$

After the textual data is vectorized, cosine similarity is applied.

`similarities = cosine_similarity([job_vector], resume_vectors)[0]`

This compares the **vector of the job description** (job\_vector-A) to each **resume vector** (resume\_vectors-B) using this formula.

#### Accuracy Score Calculation:

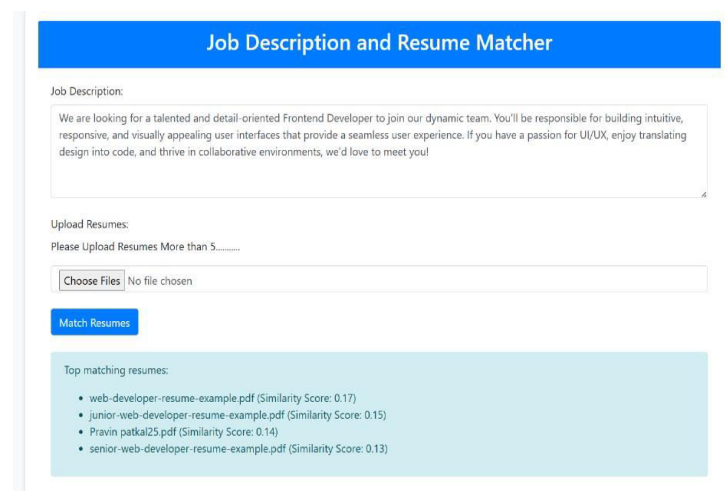
The final output is an **Accuracy Score** (or **Matching Score**) based on the cosine similarity value. This score indicates the degree of alignment between the resume and the job description. It can be interpreted in different ways:

**Above 0.8:** Strong match

**0.5–0.8:** Moderate match

**Below 0.5:** Weak match

This score can then be displayed to recruiters as a ranking metric or used in filtering candidates for further screening.



**Job Description and Resume Matcher**

Job Description:

We are looking for a talented and detail-oriented Frontend Developer to join our dynamic team. You'll be responsible for building intuitive, responsive, and visually appealing user interfaces that provide a seamless user experience. If you have a passion for UI/UX, enjoy translating design into code, and thrive in collaborative environments, we'd love to meet you!

Upload Resumes:

Please Upload Resumes More than 5.....

Choose Files No file chosen

Match Resumes

Top matching resumes:

- web-developer-resume-example.pdf (Similarity Score: 0.17)
- junior-web-developer-resume-example.pdf (Similarity Score: 0.15)
- Pravin patkal25.pdf (Similarity Score: 0.14)
- senior-web-developer-resume-example.pdf (Similarity Score: 0.13)

Fig.III.4. Resume Matcher with JD

#### How Matching Happens with JD:

- Input:** Job description and multiple resume files.
- Extract:** Text from PDF, DOCX, or TXT resumes.
- Vectorize:** All text (JD + resumes) into TF-IDF vectors.
- Compare:** Compute cosine similarity between the job description vector and each resume vector.
- Rank & Choose:** Resumes with the highest similarity scores are selected as the best matches.
- Display:** The top matching resumes and their similarity scores are presented on the HTML page.

## V. RESULTS

- Improved Job Matching** The system efficiently pairs job seekers with suitable opportunities by evaluating their skills, experience, and preferences. Enabling employers to find better candidates and reducing mismatches.



2. **Enhanced Candidate Screening:** Automated resume screening filters out unqualified candidates, saving HR teams time. Predictive analytics assess candidates' potential, factoring in past performance, experience, and skills.
3. **Streamlined Hiring Process:** Automation of tasks like resume review and interview scheduling accelerates the recruitment process, allowing employers to make quicker decisions with a more organized workflow.
4. **Customized Recommendations for Job Seekers:** Job seekers receive tailored job suggestions based on their profiles, boosting their chances of securing suitable roles. The system also provides insights to help candidates improve their profiles and employability.
5. **Bias Reduction:** The system's data-driven approach reduces hiring biases, promoting fairness in recruitment. Objective evaluation criteria ensure candidates are assessed based on qualifications and fit.

**Job Description 1:-** We are seeking a skilled and meticulous Frontend Developer to join our innovative team. The role involves designing and developing user interfaces that are intuitive, responsive, and visually engaging to ensure a smooth and user-friendly experience. If you have a passion for UI/UX, enjoy translating design into code, and thrive in collaborative environments, we'd love to meet you!

Resume	Cosine Similarity Score
Resume 1	0.14
Resume 2	0.15
Resume 3	0.17
Resume 4	0.13

Fig. IV.I. Similarity score table

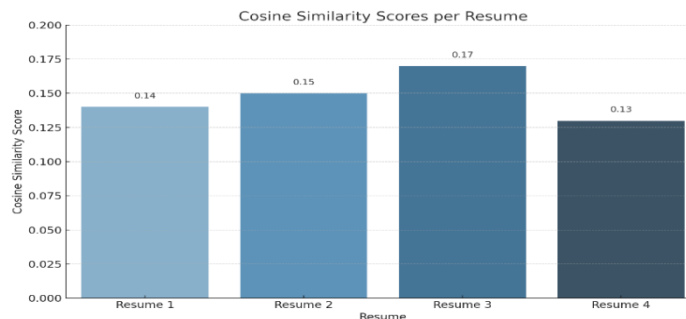


Fig. IV.I. Similarity score chart

**Job Description 2 :** We are seeking a highly motivated and analytical **Data Scientist** to join our data team. You will be responsible for turning raw data into actionable insights through advanced analytics, statistical modeling, and machine learning techniques. If you are passionate about **Python**, data-driven problem-solving, and deploying **ML** solutions, we'd love to hear from you!

Resume	Cosine Similarity Score
Resume 1	0.30
Resume 2	0.19
Resume 3	0.20
Resume 4	0.26

Fig. IV.II. Similarity score table



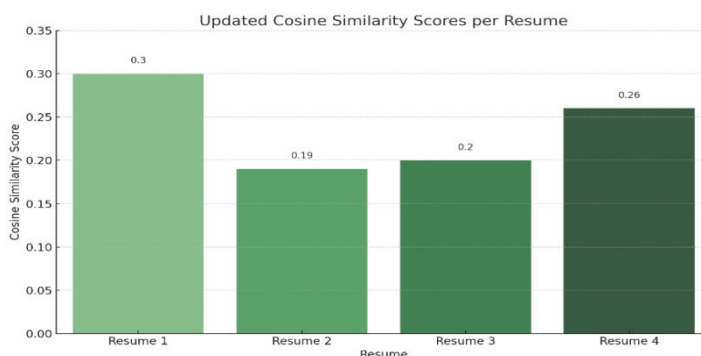


Fig. IV.II. Similarity score chart

## V. CONCLUSION

The Online Job Portal with Candidate Prediction System provides a comprehensive solution to modernize the job recruitment process. By leveraging machine learning, the system enhances job matching, improves candidate screening, and optimizes the hiring process for both employers and job seekers. Overall, the application of machine learning in the job portal ecosystem significantly contributes to transforming how recruitment handled.

## REFERENCES

1. Arunthathi S, Logeshwari T, Anuratha V, "A Research Paper on Online Job Portal System", Department Computer Engineering and Technology, Available:<https://www.ijniet.org/wpcontent/uploads/2024/05/391.pdf>.
2. Sowmya Mathukumalli, "Job Search Portal by SOWMYA MATHUKUMALLI", Department of Computer Science. Available:<https://core.ac.uk/download/pdf/77979433.pdf>.
3. S.Sivakumar1,A.Ramkumar2,S.Sankar Ganesh.
4. Prof. A.Ishwariya, "Online job portal web application using MERN stack", B.E student, Dept. of CSE, Available:<https://www.ijedr.org/viewpaperforall.php?paper=IJSDR2303014>
5. Alam Sher Khan1, Hussain Ahmad2, Muhammad Zubair Asghar3\* Furqan Khan Saddozai4, Areeba Arif5, Hassan Ali Khalid6, "Personality Classification from Online Text using Machine Learning Approach Institute of Computing and Information Technology. [Online]. Available:<https://thesai.org/Publications/ViewPaper?Volume=11&Issue=3&Code=IACSA&SerialNo=58>.
6. Nuthalapati Leena, Normala Sandhya, Reddem Sai Archana Reddy, Shaik KasimSaheb', "Personality prediction through cv analysis" Student, Department of Computer Science. [Online]. Available:<https://www.studocu.com/row/doc/ument/university-of-sargodha/websystemandtechnologies/personality-prediction-through-cv-analysis/4627607>
7. Friedrich-Alexander-University, Schöller Endowed Chair for Information Systems, Erlangen-Nuremberg, Germany; [jessica.ochmann@fau.de](mailto:jessica.ochmann@fau.de), [sven.laumer@fau.de](mailto:sven.laumer@fau.de)
8. Assistant Professor, Department of Computer Science, Maharaja Agrasen Institute of Technology, Delhi, New Delhi, India 2,3,4 UG Student, Department of Computer Science, Maharaja Agrasen Institute of Technology, Delhi, New Delhi, India
9. [https://ijariie.com/AdminUploadPdf/Online\\_Job\\_management\\_ijariie20527.pdf?srsId=AfmBOoonK5W0A\\_12BYjzEpcgVpPFGsnqLBJ0wmpygU7E4g4OsIknNIUu](https://ijariie.com/AdminUploadPdf/Online_Job_management_ijariie20527.pdf?srsId=AfmBOoonK5W0A_12BYjzEpcgVpPFGsnqLBJ0wmpygU7E4g4OsIknNIUu)
10. [https://www.irjmets.com/uploadedfiles/paper/issue\\_5\\_may\\_2023/39951/final/fin\\_irjmets1684846640.pdf](https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2023/39951/final/fin_irjmets1684846640.pdf)



INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  [ijirset@gmail.com](mailto:ijirset@gmail.com)



[www.ijirset.com](http://www.ijirset.com)

Scan to save the contact details