WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

ZYNCROLE AI - An Autonomous AI Career Intelligence and Personalized Job Matching Platform

Sini Prabhakar, Karunya Sri V, Harish R, Kavin P, Kabilesh P

Department of Artificial Intelligence and Data Science, Sri Shakthi Institute of Engineering and Technology

ABSTRACT

Choosing a suitable career and finding relevant job opportunities can be challenging for students, fresh graduates, and job seekers. Conventional job portals often rely on keyword-based searches and require users to manually compare job descriptions, skills, qualifications, and career goals. This can lead to mismatched applications, missed opportunities, and uncertainty about the skills needed to become job-ready.

ZyncRole AI is proposed as an autonomous AI career intelligence and personalized job matching platform whose central function is to collect and consolidate job postings from multiple authorized online platforms and job sources into one searchable system. The collected listings are standardized, checked for missing or outdated information, and processed before being matched with a candidate's profile. The platform also brings together career discovery, résumé/profile analysis, job recommendations, and skill-gap guidance in a unified workflow.

The platform is designed to combine multi-source job collection, data normalization, duplicate and freshness checks, natural language processing, semantic similarity, recommendation methods, and a user-friendly web interface. A dashboard can present consolidated job listings from different sources, ranked opportunities, source links, match explanations, skill-gap summaries, and application tracking. The system aims to reduce the need to visit many job portals separately and help users make more informed career decisions. Recommendations are decision support, not guaranteed employment outcomes.

Index Terms—Artificial Intelligence, Career Intelligence, Job Matching, Natural Language Processing, Recommendation System, Résumé Parsing, Skill-Gap Analysis, Personalized Career Guidance.

I. INTRODUCTION

Artificial Intelligence is increasingly being used to support decision-making in education, recruitment, and workforce development. At the same time, the job market contains a large and changing collection of roles with different qualification requirements, skill expectations, experience levels, and locations. Students and early-career applicants may find it difficult to determine which positions align with their current capabilities and long-term interests.

Traditional job-search methods commonly depend on keyword searches, filters, and manual résumé comparison. A job title may use different terminology from a candidate's résumé even when the underlying skills are related. Conversely, two roles with similar titles may require very different competencies. These limitations can make search results less relevant and place a significant time burden on applicants.

ZyncRole AI addresses this problem through a multi-source job aggregation and personalized career intelligence workflow. It gathers job postings from multiple authorized platforms or data providers, consolidates them in a common job database, and standardizes fields such as title, company, skills, experience, location, listing date, and application URL. It then combines candidate profile understanding, résumé information extraction, job-description analysis, matching-score generation, and actionable skill-gap recommendations. Instead of requiring applicants to search each platform separately, the system provides one place to discover and compare collected opportunities.

The platform follows a modular web architecture. The frontend enables users to manage profiles, explore recommendations, review match explanations, and track applications. A backend service handles profile processing, job data management, matching logic, and API requests. AI and natural language processing components extract structured information and compare candidate profiles with job requirements. A database stores user-approved profile details, job records, recommendations, and application status.

The main contributions of ZyncRole AI are multi-platform job-posting aggregation, a unified career discovery workflow, explainable job matching, practical skill-gap analysis, and personalized guidance that can be updated as a user's profile changes. The system is designed to support informed decisions, not to replace human judgment or make unsupported promises about hiring outcomes.

II. EXISTING SYSTEMS

Existing job portals provide searchable job listings, filters, employer information, and application facilities. These functions are useful, but candidates generally need to interpret job descriptions themselves and decide whether their skills meet the requirements. Keyword-based search may also overlook related skills expressed using different terms.

Career guidance websites and assessment tools can help users explore occupations, but they may operate separately from job-search platforms. As a result, a user may receive general career suggestions without a clear connection to current vacancies or the specific skills required for a target role.

Résumé builders and applicant tracking tools help organize candidate information, but they do not always provide a transparent explanation of how a candidate profile aligns with individual job requirements. Some recommendation systems provide rankings without clearly showing the factors that influenced the ranking.

The key limitations are fragmented job sources, the need to visit multiple portals, dependence on manual comparison, limited personalization, weak explanations for recommendations, and insufficient connection between job matching and skill development. ZyncRole AI proposes to aggregate listings from multiple authorized sources in one platform and combine them with transparent match factors and user-controlled profile information.

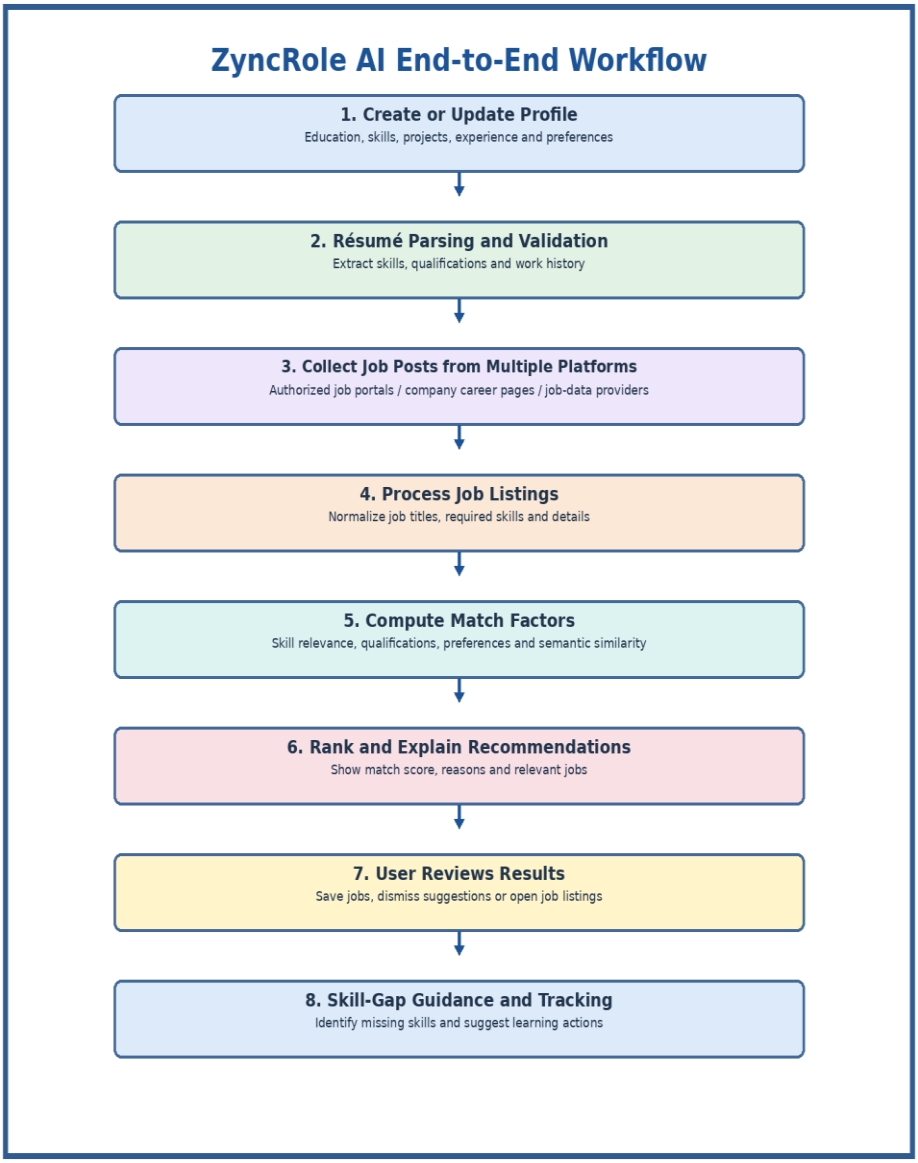


Figure 2. End-to-end workflow of the proposed platform.

Table 1. Comparison of existing approaches and the proposed system.

Existing Approach	Common Limitation	ZyncRole AI Improvement
Keyword-based job portals	May miss related skills expressed differently	Uses normalized skills and semantic similarity
Separate career guidance tools	Advice may not connect to current vacancies	Connects career guidance with job recommendations
Manual résumé comparison	Time-consuming and inconsistent	Automates profile-to-job comparison
Opaque recommendation lists	Limited explanation for rankings	Shows match factors and possible skill gaps

III. SYSTEM DESIGN AND OBJECTIVES

A. System Architecture

The proposed system is organized into three main layers, supported by a multi-source job collection pipeline that gathers listings from authorized job platforms and providers before the AI matching stage.

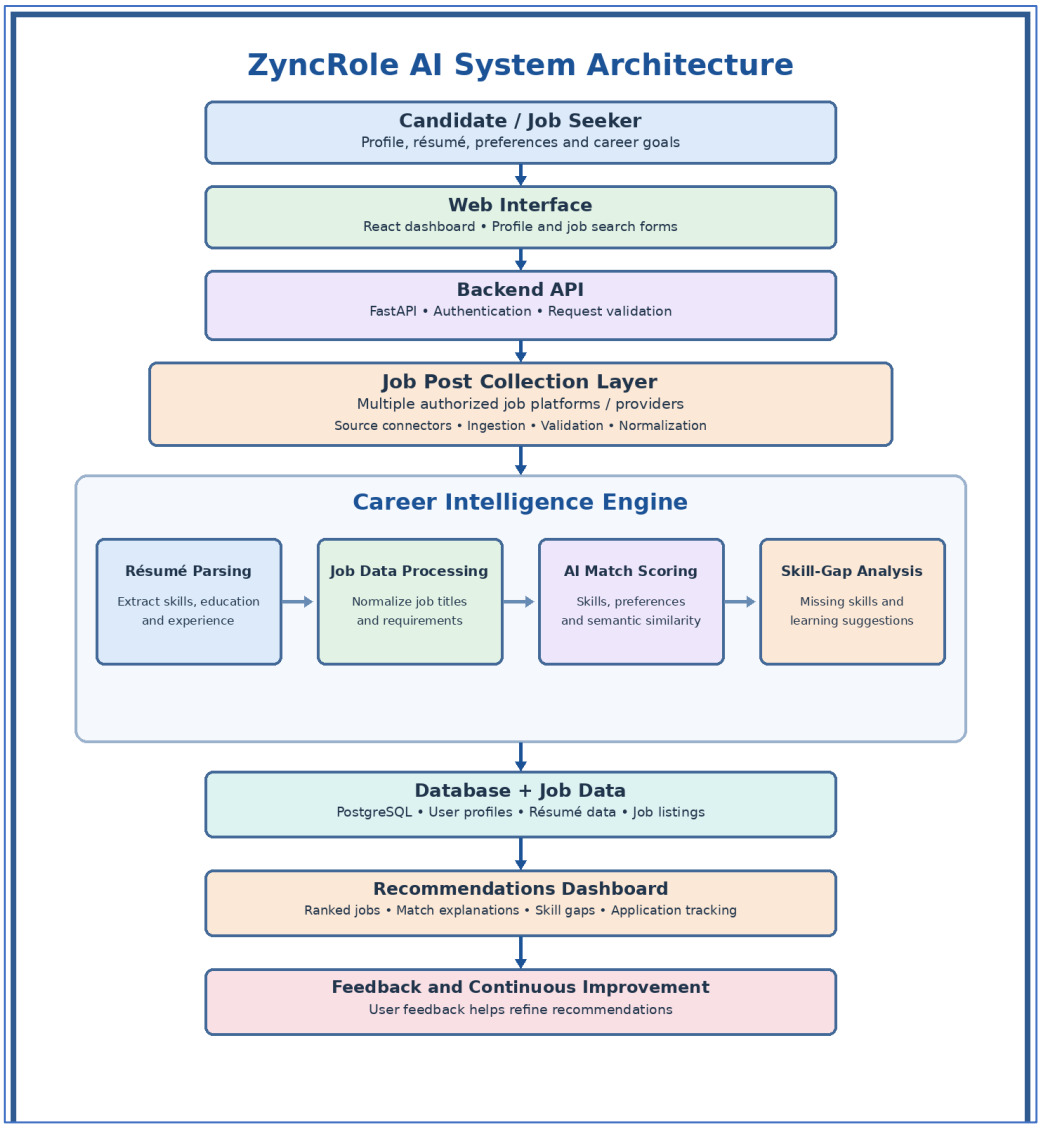


Figure 1. Proposed system architecture of ZyncRole AI.

1. Input and Profile Layer

This layer collects the information required to personalize recommendations. Users can enter education, skills, projects, certifications, experience, preferred roles, locations, work arrangements, and career interests. If résumé upload is supported, the parser extracts relevant text and proposes structured fields for the user to review before saving.

2. AI Career Intelligence Layer

This layer processes candidate profiles and job descriptions collected from multiple sources. A job aggregation service retrieves listings through permitted APIs, feeds, licensed datasets, or other authorized methods. Source adapters map different platform formats into a common schema; preprocessing then normalizes fields and checks for duplicate, incomplete, or expired listings. Natural language processing identifies skills, qualifications, responsibilities, and experience requirements. A matching engine compares candidate information with job requirements and preferences. A skill-gap module highlights important requirements not yet demonstrated in the profile and can suggest learning priorities.

3. Application and Visualization Layer

This layer provides the web dashboard and communicates with backend APIs. It displays ranked job recommendations, match explanations, missing-skill summaries, career suggestions, saved jobs, and application status. Results should be presented clearly so users can understand why a role was recommended.

B. System Workflow

User creates a profile or uploads a résumé for review.

The platform extracts and organizes education, skills, experience, and preferences.

Job descriptions are processed to identify role requirements and relevant skills.

The matching engine compares candidate profiles with available job records.

Jobs are ranked using skill relevance, qualifications, preferences, and other configured factors.

The dashboard explains matching strengths and highlights possible skill gaps.

The user reviews recommendations, saves relevant jobs, and tracks application progress.

The profile and recommendations can be updated as the user gains skills or changes career goals.

C. Objectives

Provide personalized job recommendations based on a candidate's profile and preferences.

Reduce manual effort involved in comparing résumés with job descriptions.

Use natural language processing to recognize related skills and job requirements.

Explain key factors that contribute to each match rather than presenting a score alone.

Identify potential skill gaps and recommend practical learning priorities.

Support career exploration for students, fresh graduates, and other job seekers.

Provide a unified dashboard for saved jobs and application tracking.

Protect personal information and keep users in control of profile edits and application decisions.

IV. SYSTEM DESIGN

A. High-Level Design (HLD)

At a high level, the system contains a web frontend, backend API, multi-source job collection service, career intelligence services, job-data store, and user database. The collection service connects to multiple authorized job platforms or providers using permitted APIs, feeds, licensed datasets, or other authorized access methods. Source-specific adapters transform incoming listings into a shared schema. The backend validates and stores the records, applies freshness and duplicate checks where possible, and retrieves relevant jobs for matching. The frontend sends profile updates and search requests to the backend. AI components process the profile and job descriptions and return recommendations with explanations for the dashboard.

The architecture is modular so that résumé parsing, job-description processing, matching, skill-gap analysis, and application tracking can be developed and tested separately. External job data, where used, must be obtained through permitted APIs, licensed datasets, or other authorized sources.

B. Low-Level Design (LLD)

The profile module validates and normalizes user-provided fields. The résumé parser extracts text and identifies possible entities such as skills, education, certifications, and work experience. Extracted details should be shown to the user for confirmation because parsing may be incomplete or incorrect.

The job processing module standardizes job titles, required skills, experience levels, location, and work arrangements. The matching module computes a compatibility estimate using configurable factors such as skill overlap, semantic similarity, qualification fit, experience relevance, and user preferences. The system can apply weights to these factors, but the weights and resulting scores must be tested and explained rather than treated as objective measures of a candidate's worth.

The recommendation output module returns ranked jobs with reasons for the ranking, matching skills, unmet requirements, and profile improvement suggestions. The application tracking module stores user-selected statuses such as saved, planned, applied, interview, or closed. Access control and validation should be applied to personal and application data.

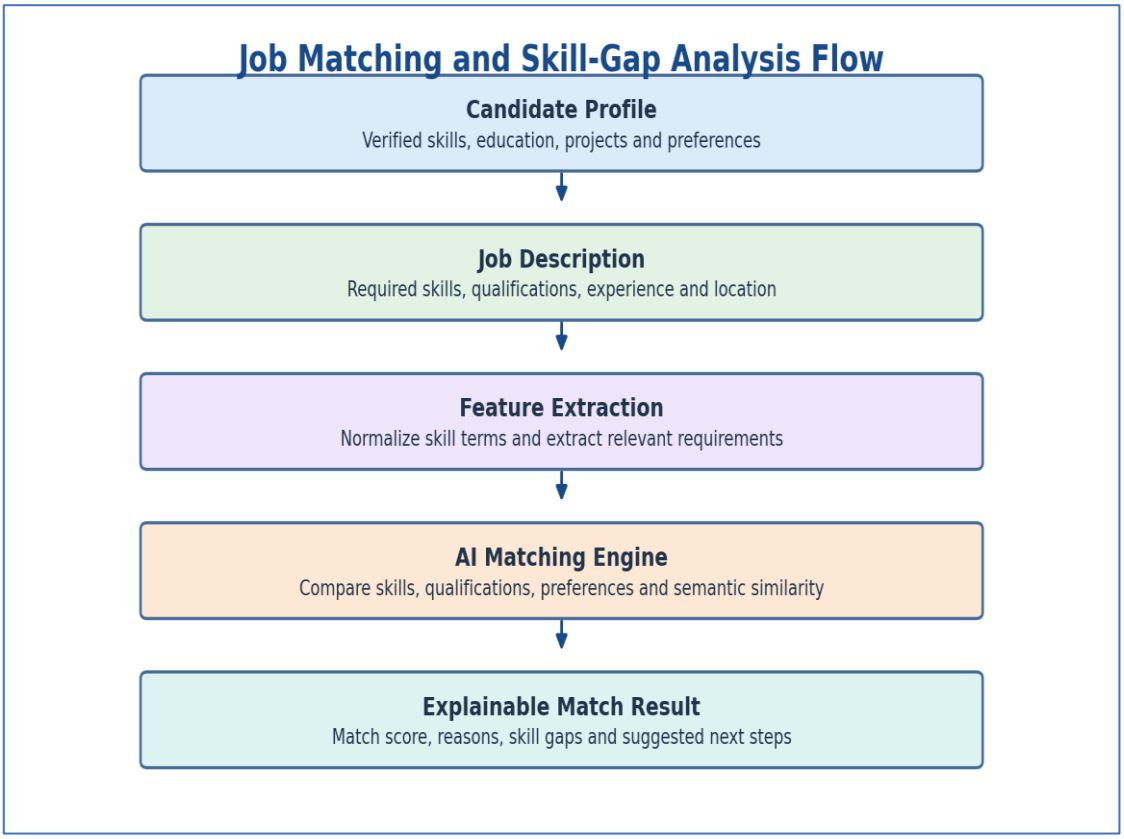


Figure 3. Matching and skill-gap analysis process.

V. METHODOLOGY

A. Data Collection

The system uses candidate information supplied by the user and job postings collected from multiple authorized online platforms and job-data providers. Each source may expose different fields and formats, so source-specific connectors retrieve the available listing details and pass them to a common ingestion pipeline. Candidate data may include skills, education, projects, certifications, experience, interests, preferred job titles, location, and work mode. Job data may include title, employer, description, required skills, qualifications, experience, location, listing date, source platform, and application link.

B. Multi-Platform Job Collection and Aggregation

This module is a core part of ZyncRole AI. It gathers available job postings from multiple authorized job portals, company career pages, public job feeds, or licensed job-data providers. Because each source may use a different format, a connector/adapter is used for each source and maps the available fields into a common structure: job title, company, description, skills, qualifications, experience, location, date posted, source name, and application URL. The ingestion pipeline validates records, records source information, and identifies likely duplicates or stale listings when the necessary data is available. The consolidated database then supplies current, searchable job records to the AI matching and ranking modules. Integrations must use permitted APIs, feeds, licensed datasets, or other authorized methods and respect each source's terms and access restrictions.

B. Data Preprocessing

Preprocessing improves consistency across postings collected from different platforms before matching. The pipeline maps source-specific fields to a common format, normalizes text, removes duplicate job records when identifiable, standardizes skill names, identifies missing fields, and flags listings that may be outdated. It also separates relevant sections of résumés and job descriptions.

Normalize capitalization, spacing, and common abbreviations.

Extract résumé text and candidate-confirmed profile fields.

Identify skills, qualifications, experience, and role responsibilities.

Remove duplicate or expired listings when the source data supports this.

Validate data types and handle missing information without inventing values.

C. Natural Language Processing

Natural language processing is used to analyze free-text résumés and job descriptions. A first version may use keyword and phrase extraction with a curated skill dictionary. A more advanced version may use text embeddings to compare the semantic meaning of candidate experience and job requirements. Extracted information should be checked for accuracy and should not be interpreted as verified credentials.

D. Job Matching and Ranking

Each job can be evaluated against the candidate profile using a configurable matching function. A conceptual score may combine skill relevance, qualification alignment, experience fit, location or work-mode preference, and semantic similarity. The score is a ranking aid, not a probability of being hired. The platform should show which factors influenced the result and allow users to adjust preferences.

For example, a candidate interested in data analytics may be matched with a junior analyst role when the profile includes Python, spreadsheets, data visualization, and relevant projects. If the listing requires SQL or a particular reporting tool that is not present in the profile, the platform can identify it as a possible development area.

E. Skill-Gap Analysis

The skill-gap module compares the skills evidenced in a candidate profile with the requirements of selected target roles. It groups requirements into demonstrated skills, related skills, and skills not yet evidenced. The system can recommend a prioritized learning plan based on the user's target role, current profile, and the importance of requirements across selected listings.

F. Evaluation

The platform should be evaluated using a curated set of candidate profiles and job descriptions reviewed by human evaluators. Relevant measures may include Precision@K, Recall@K, ranking quality, duplicate detection rate, parsing accuracy, response time, and user feedback. Evaluation must use documented test data and should avoid claiming measured performance until experiments have actually been conducted.

G. End-to-End Workflow

Collect or enter candidate profile data with user consent.

Parse and normalize candidate information.

Ingest and validate authorized job descriptions.

Extract skills and requirements from each job listing.

Calculate match factors and rank relevant opportunities.

Generate explanations and possible skill-gap recommendations.

Display recommendations and allow users to save or dismiss them.

Gather feedback and update the profile and ranking preferences when requested.

VI. SOFTWARE IMPLEMENTATION

The software implementation is planned as a full-stack web application. The exact framework and services may be adjusted to the development team's resources and project requirements.

A. Development Environment

Frontend: React.js, HTML, CSS, and JavaScript for the interactive dashboard.

Backend: Python with FastAPI or Flask for profile, matching, and application APIs.

AI/NLP: Python libraries for text processing, skill extraction, semantic similarity, and recommendation logic.

Data handling: Pandas and NumPy where tabular processing is required.

Database: PostgreSQL or another suitable database for user profiles, job listings, matches, and application status.

Development tools: Visual Studio Code, Git, and GitHub for coding and version control.

Deployment: A suitable cloud or local hosting environment with secure configuration.

B. System Integration

The frontend communicates with the backend through REST APIs. A multi-source job collection service retrieves postings from authorized platforms and providers, while source adapters normalize their different formats into a shared job schema. The backend validates and stores listings, tracks source and application URLs, and checks duplicates or freshness when possible. It then coordinates the profile parser, matching engine, and database. Matching results are returned with a score breakdown and explanation so the interface can show more than a ranked list.

Table 3. Proposed technology stack.

Layer	**Suggested Technology**	**Purpose**
Frontend	React.js, HTML, CSS, JavaScript	Profile forms and recommendation dashboard
Backend API	Python with FastAPI or Flask	Validation, business logic and API endpoints
AI / NLP	Python NLP libraries and text embeddings	Skill extraction and semantic comparison
Data processing	Pandas and NumPy	Cleaning and processing structured data
Database	PostgreSQL	Profiles, job records, matches and application status
Version control	Git and GitHub	Collaboration and source-code history

Conceptual workflow: Multiple Authorized Job Platforms / Providers → Source Connectors → Job Ingestion, Validation and Normalization → Consolidated Job Database ← User Profile / Résumé → AI Matching and Skill-Gap Engine → Ranked Recommendations Dashboard.

VII. SYSTEM MODULES

A. User Profile Module

Supports profile creation and editing, education and skill entry, experience and project details, career interests, and preferred job conditions. Users should be able to correct extracted résumé information and control which profile details are retained.

B. Résumé Parsing Module

Extracts text from supported résumé formats and identifies candidate details. The module should preserve uncertainty, ask the user to confirm extracted fields, and avoid treating a résumé statement as independently verified information.

C. Job Data Module

Collects and consolidates job postings from multiple authorized platforms and job-data providers. Source connectors retrieve available listing details; the ingestion pipeline maps fields into a common schema, validates records, and checks for duplicates or expired listings where possible. The module stores and updates job descriptions and standardizes job title, company, required skills, experience, location, work mode, source platform, listing date, and application link. Listing freshness and source attribution should be displayed where available.

D. Matching and Recommendation Module

Compares candidate profiles with job requirements, ranks suitable listings, and produces understandable explanations. Users should be able to filter by role, location, experience level, and work arrangement and dismiss recommendations that are not relevant.

E. Skill-Gap and Career Guidance Module

Identifies requirements that are not evidenced in the current profile and suggests learning priorities. Recommendations should be practical, relevant to selected career goals, and framed as guidance rather than mandatory requirements.

F. Dashboard and Application Tracking

Presents recommended jobs, match explanations, saved opportunities, and application statuses. Users can organize their job search and update progress without needing separate spreadsheets.

G. Privacy, Security, and Fairness

Table 2. Core modules, inputs and outputs.

Module	Input	Output
User Profile	Education, skills, experience, preferences	Structured candidate profile
Résumé Parser	Uploaded résumé text	Extracted fields for user confirmation
Job Data	Authorized job listings	Normalized job records
Matching Engine	Candidate profile + job requirements	Ranked jobs with match factors
Skill-Gap Analysis	Target role requirements	Missing-skill summary and learning priorities
Application Tracker	User status updates	Saved jobs and application progress

The platform should collect only the information needed for its functions, protect stored data, use access controls, and provide a way to correct or delete profile information. Matching should avoid using sensitive personal characteristics as ranking factors. The system should be tested for inconsistent recommendations and should clearly state that a match score does not guarantee selection.

VIII. RESULTS AND DISCUSSION

A. Evaluation Setup

A planned evaluation can use representative job descriptions and sample candidate profiles covering different education levels, skills, and career interests. Human reviewers can judge whether the top-ranked results are relevant, whether the explanations are understandable, and whether extracted résumé details are accurate. The evaluation dataset, criteria, and results should be documented.

B. Evaluation Measures

Precision@K: proportion of the top K recommendations judged relevant.

Recall@K: proportion of relevant opportunities retrieved within the top K results.

Parsing accuracy: correctness of extracted profile and job fields.

Ranking quality: agreement between system ranking and human relevance judgments.

Response time: time required to produce recommendations for a profile.

Usability feedback: user assessment of clarity, relevance, and ease of use.

C. Expected Outcomes

The intended outcome is a unified career platform that collects and consolidates job postings from multiple authorized sources, then helps users discover relevant listings, understand why each listing may fit, and identify skills that could strengthen their applications. The integrated workflow is expected to reduce repetitive searching across separate portals and manual comparison. These are design goals and must be validated through implementation and testing before quantitative performance claims are made.

Table 4. Proposed evaluation plan (results must be measured during testing).

Evaluation Area	Suggested Measure	What It Checks
Recommendation relevance	Precision@K / Recall@K	Whether relevant jobs appear near the top
Résumé extraction	Field-level parsing accuracy	Correctness of extracted candidate details
Ranking quality	Human relevance review	Whether ranking aligns with reviewer judgments
System efficiency	Response time	Time taken to return recommendations
Usability	User feedback questionnaire	Clarity and ease of using the platform

D. Discussion and Limitations

Recommendation quality depends on the completeness and freshness of job data and on the accuracy of candidate profiles. Résumé parsing may miss context, and semantic similarity may associate terms that are not equivalent in practice. Job requirements can also be incomplete or overly broad. To reduce these issues, the platform should explain its results, allow profile correction, show job-source details, and provide feedback controls.

The platform does not guarantee interviews or employment. A high matching score should not be treated as a hiring decision, and a lower score should not prevent a user from applying. The user should remain responsible for reviewing job details and deciding whether to submit an application.

IX. CONCLUSION

ZyncRole AI is proposed as an autonomous AI career intelligence and personalized job matching platform whose central function is to collect and consolidate job postings from multiple authorized online platforms and job sources into one searchable system. The collected listings are standardized, checked for missing or outdated information, and processed before being matched with a candidate's profile. The platform also brings together career discovery, résumé/profile analysis, job recommendations, and skill-gap guidance in a unified workflow.

The system combines a web interface with backend services, natural language processing, a matching engine, and structured data storage. By presenting match factors and possible skill gaps, the platform aims to help students and job seekers plan their next steps more confidently. Future development can improve semantic matching, expand authorized job-data integrations, provide more personalized learning pathways, and strengthen privacy, fairness, and evaluation procedures.

ACKNOWLEDGMENT

The authors express their sincere gratitude to Sini Prabhakar for guidance and support throughout the development of this project. The authors also thank the Department of Artificial Intelligence and Data Science, Sri Shakthi Institute of Engineering and Technology, for providing an academic environment and resources to develop this work.

REFERENCES

[1] P. D. F. L. B. and others, “Recommender Systems: An Introduction,” Springer, 2011.

[2] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.

[3] D. Jurafsky and J. H. Martin, Speech and Language Processing, draft edition.

[4] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python, O’Reilly Media, 2009.

[5] F. Ricci, L. Rokach, and B. Shapira, Recommender Systems Handbook, Springer, 2015.

[6] T. Mikolov et al., “Efficient Estimation of Word Representations in Vector Space,” arXiv, 2013.

[7] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” EMNLP-IJCNLP, 2019.

[8] A. B. S. and others, “A Survey on Bias and Fairness in Machine Learning,” relevant academic literature on responsible AI.

[9] Python Software Foundation, “Python Documentation.”

[10] React Team, “React Documentation.”

[11] FastAPI, “FastAPI Documentation.”

[12] PostgreSQL Global Development Group, “PostgreSQL Documentation.”