Intelligent Matching Platform for University Students' Employability

Linna Li[1,*], Jiayu Li[1], Yifan He[1], Qingfeng Zhou[2]
[1]*School of Artificial Intelligence and Big Data, Henan University of Technology, Zhengzhou, Henan, China*
[2]*iFLYTEK Co., Ltd., Hefei, Anhui, China*
**Corresponding Author*

Abstract: Therefore, this paper will introduce an intelligent matching platform for the employment prospects of college students and address the above issues. The Platform has been divided into a front-end and a back-end. Vue 3 and Element Plus are used for the user interface, Spring Boot and MyBatis for the back-end service, and JWT is used to ensure stateless authentication. Kafka is used for real-time data processing as a message queue, Flink performs stream-based intelligent matching calculations, and ClickHouse stores the matching results and statistics. The five dimensions of the multi-dimensional weighted scoring model are city, education level, major, salary and skills; accurate job recommendations are thus generated. An AI job-seeking assistant based on the DeepSeek-V3 large language model will offer real-time career advice. The three types of users for the system are students, enterprises and administrators, and all have data dashboards for employment monitoring. Experimental tests of less than 100 concurrent users showed that the average response time was less than 500 milliseconds, and thus the platform had effectively improved the accuracy of work matching, simplified employment management, and raised the general standard of university employment services.

Keywords: Intelligent Job Matching; Employment Service Platform; Real-Time Data Processing; Hybrid Recommendation Algorithm; Multi-Dimensional Weigscoring Model; Vue 3

1. Related Technologies Overview

(1) System Technical Architecture. The system has the B/S mode and is divided into front-end and back-end. Vue 3 is used on the front end to build a single-page application, and Spring Boot is employed for the back end to provide RESTful API services. Kafka and Flink are used to build a real-time data processing and storage system for the big data part.

(2) Front-End Technology. Vue3 is a gradually adopted JavaScript library for front-end development. It supports the Composition API, is based on a component model, and can build interactive web applications efficiently. The front-end of this system is based on Vue 3's Composition API, Vue Router for page routing, Pinia for global state management, and Axios for encapsulating HTTP requests [1].

Element Plus is a desktop component library based on Vue 3 that provides a large number of UI components, such as forms, tables, dialog boxes, etc., and can be used to build beautiful and consistent management interfaces quickly.

(3) Back-End Technology. Spring Boot is a simple, small-featured implementation of the Spring framework. It has an integrated Tomcat server by default and supports automatic configuration and quick deployment. Spring Boot is used for the development of the backend RESTful API in this system, and MyBatis is used for database operations. The Design will separate the business logic from the data access layer cleanly [2].

JWT is a JSON Web Token that can be used to securely pass information across different platforms; it is open-source. This system uses JWT for stateless login authentication [3]. Log in successfully, and then the server returns a Token. For all following requests, the client will add this Token to the request and the server will authenticate the user based on it.

(4) Big Data Technology. Kafka is a high-throughput distributed message queue for asynchronous data transmission [4]. Kafka is used to transport student and job data in the collection-to-processing stage of this system [5].

Flink is a high-throughput, low-latency stream-processing system. Flink will be used in this system to perform real-time matching of student resumes and job information based on Kafka. ClickHouse is a high-performance OLAP database for columns [6]. ClickHouse will be employed to store the matched results and statistics of this system. ZooKeeper provides distributed coordination services for the Kafka and Flink clusters [7].

(5) Database and Deployment Technology. MySQL is an excellent open-source relational database system that supports ACID transactions and has good performance and stability. MySQL will be used in this system to store the primary data, such as user information, job details and application history.

Docker is a containerisation deployment technology that can package applications and their dependencies into lightweight containers to achieve environment consistency [8,9], and this system uses Docker for containerised deployment of service components such as Kafka, Flink, ClickHouse and ZooKeeper.

2. System Requirements and Design

2.1 System Overview

The three types of users for this system are college graduates, corporate recruiters and university administrators; the whole is an employment service platform that includes job posting, intelligent recommendation, application management and employment tracking. The system has been divided into a front-end and a back-end, the front-end serves as the user interface, the back-end processes business logic, and a big data platform is used to store and analyze large amounts of data.

2.2 User Role Analysis

The three kinds of users for the system are students, enterprises and administrators. The Requirements for different users are different.

(1) The main users of the system are students. They can do many things in the system. They can complete the personal information and resume. Browse and search for job postings. Then it will show some suitable jobs for them. They can apply for the positions they wish to have. They will be able to see the application status. Sign the employment contract and start working; then, register your employment information in the system. They can also see the employment policies published by the school. If they have any problems with the job search process, they can ask the AI job assistant in the system at any time.

(2) The motivations for companies using this platform are fundamentally different from those of students. For a business, the first step is to register for an account and complete the mandatory corporate verification. Once verified, they can post job openings and manage their entire recruitment process.

Then, the company will have received some applications from students and will review them. Finally, they will have chosen one or the other for each candidate: hire or not.

(3) Administrators are the gatekeepers of the platform, tasked with comprehensive review and oversight. They vet every enterprise's registration details and job postings, ensuring nothing goes live until it has received formal approval. On top of that, they are in charge of managing the entire student records system.

They are also to distribute employment-related policies and official announcements. They regularly view the data dashboard for changes in the key employment indicators during their daily work and adjust all operations according to this information.

2.3 Functional Requirements Analysis

2.3.1 Student-Side Functional Modules

The student end is the first place that people use the system. It provides graduating students with all the necessities for job hunting in one place. As shown in Figure 1, the main functions for students are:

(1) Login and Registration Module. Students register for an account with their student ID and need to fill in other information such as their name, student ID, major and mobile phone number during registration. Successfully registered; now they can log in with their student ID and password. After the backend verifies it, a JWT token is created and then returned to the front end.

(2) Resume Management Module. Students can fill out their own resumes online and provide all sorts of information, such as basic details, skills and strengths, work experience, language proficiency, awards, etc. The intelligent recommendation algorithm uses resume data.

(3) Job Browsing and Application Module. All approved job postings in the system can be viewed by students, and keyword search

functions for job titles and company names are available to apply for a suitable position.

(4) Intelligent Recommendation Module. According to the contents of a student's resume, a multi-dimensional weighted scoring model in the system automatically calculates how well it matches with the requirements of various job openings. Positions with a high matching degree will be recommended to students first, and the matching percentage and reason for recommendation will also be shown.

(5) Employment Registration Module. After securing a job, the student can add the employment details to the system and select a status, such as employed, awaiting a job offer, continuing studies, or starting a business, along with the name of the employing organisation.

(6) Policy Viewing Module. Students can see the list of jobs, notices and other information uploaded by administrators.

(7) AI Job-Seeking Q&A Module. The system integrates an AI job-seeking assistant by connecting to the SiliconCloud DeepSeek-V3 large model API. Students can input job-seeking related questions (such as "What is a tripartite agreement," "How to prepare for an interview," etc.), and the system returns professional answers through the large model API.

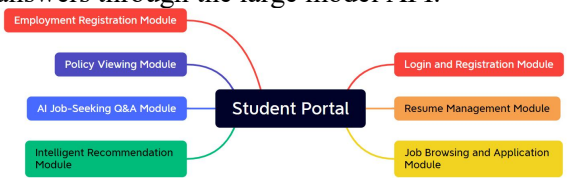

Figure 1. Student Use Case Diagram

2.3.2 Enterprise-side Functional Modules

The end of the enterprise is a recruiting company, and it has the functions of enterprise registration, job posting and talent screening. Its first few are:

(1) Enterprise Registration and Login Module. Enterprises are required to fill in their business registration information to register. After registration, the default status is "pending review," and they must wait for administrator approval before they can log in and use the system.

(2) Enterprise Information Management Module. After logging in, enterprises can view and edit their basic information, including company profile, contact phone number, email address, and address.

(3) Job Posting and Management Module. Enterprises can publish job postings, which must be reviewed and approved by the administrator before being displayed to the student end. The module supports editing, removing, or deleting published positions.

(4) Application Processing Module. Enterprises can view all application records submitted by students for their company's positions, and perform "accept" or "reject" operations on the applications.

2.3.3 Administrator-Side Functional Modules

The administrator end provides management functions, and the primary ones are as follows:

(1) Administrator Login Module. The administrator logs into the system with a particular account, and a JWT token is generated after the backend verifies successfully.

(2) Student Management Module. Administrators can view the information of all students in the system, and keyword search by name or student ID is available.

(3) Enterprise Management and Job Review Module. Administrators can view all registered enterprise information and job postings posted by enterprises, and perform "approve" or "reject" operations on enterprises or positions in the "pending review" state.

(4) Policy Management Module. Administrators can post employment policies and announcements, and the posted policies support editing and deletion.

(5) Data Dashboard Module. The Data Dashboard is used to present the employment data in various visual forms, such as a ring chart of the employment rate, a line chart showing the trend of applications, a popular job ranking chart, a salary distribution bar chart, a city distribution map, an enterprise distribution pie chart, etc.

2.4 System Architecture Design

The system is a front-end and back-end separated type in general, and it has integrated big data real-time processing technology. The five levels are, in order, the presentation layer, the business logic layer, the message queue layer, the processing layer and the storage layer.

The Vue 3 framework is used to build a front-end single-page application (SPA) in the presentation layer, which will be responsible for displaying and interacting with the user interface. Acquire data by invoking the back-end RESTful APIs via Axios.

Spring Boot is used at the business logic level to process requests from the front-end and perform all kinds of business operations, such as user authentication, job recommendation algorithms and application process management.

Kafka is used in the message queue layer to perform asynchronous data transmission and buffering, and the collected student data and job data from the collection end are distributed to the processing nodes.

The processing layer is a Flink stream processing engine that performs real-time matching calculations on the data stream from Kafka and generates matching results based on a multi-dimensional weighted scoring model.

MySQL will be employed in the data storage layer for general business data, and ClickHouse will be used to store matching results and statistics. Figure 2 is the connection structure of the five layers.

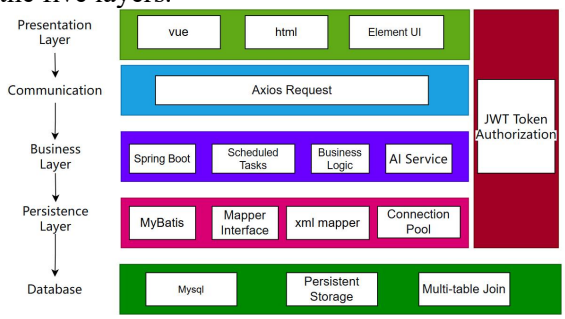

Figure 2. Overall System Architecture Diagram

2.5 System Functional Module Division

Based on the analysis of the functional requirements, the main modules of the system will be as follows:

(1) User Authentication Module: Responsible for the registration, login, and identity authentication of three categories of users: students, enterprises, and administrators.

(2) Resume Management Module: Responsible for the creation, editing, and storage of student resumes.

(3) Job Position Management Module: Responsible for the release, review, display, and search of job positions.

(4) Intelligent Recommendation Module: Responsible for calculating the matching degree between resumes and job positions and making recommendations based on the calculation.

(5) Application Management Module: Responsible for students' job applications, enterprises' processing of applications, and the tracking of application status.

(6) Employment Management Module: Responsible for the registration of students' employment information and the statistics of employment rates.

(7) Policy Management Module: Responsible for the release and viewing of employment policies.

(8) Data Analysis Module: Responsible for the statistical analysis and visual display of employment data.

3. System Implementation

3.1 Big Data Processing Module

3.1.1 Data Simulation and Transmission

The MockDataGenerator module of the system generates and provides simulated student resumes and job recruitment data with various attributes, such as city, education, major, salary, skills, etc. FastJSON is used to serialize the generated data into a JSON format, and then this data is published to a particular Kafka Topic by the Kafka Producer for asynchronous transmission.

3.1.2 Data Consumption and Storage

Independent Kafka consumer modules (StudentKafkaToCK and JobKafkaToCK) are deployed to subscribe to the Topics for student data and job data, respectively. Pull messages from Kafka and then write the data to a ClickHouse database via the ClickHouse JDBC driver to store the data persistently.

3.1.3 Real-time Matching Calculation

The two-way real-time matching calculation for students and job positions in the system is carried out by a Flink stream-processing engine. The Flink job reads student data and job data from ClickHouse, performs weighted scoring based on the five dimensions of city, educational background, major, salary, and skills, calculates the matching degree, and then writes the results to ClickHouse for storage. The above mechanism can support both cases: students looking for jobs and companies recruiting students.

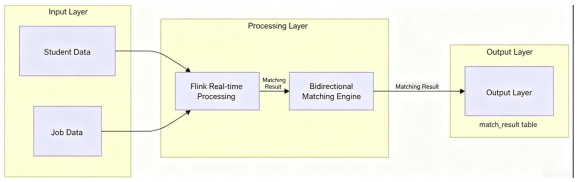

Figure 3. Design of Real-Time Matching Engine

3.2 Backend Service Module

3.2.1 Implementation of User Authentication

The frontend sends the username and password to the back-end upon login. After the backend successfully completes the verification, it will generate a JWT token and return it. The frontend stores the token locally, and then adds it to the

Authorization header of subsequent requests for identity verification. The backend Filter will not pass the request if the Token is invalid.

3.2.2 Implementation of the Intelligent Recommendation Algorithm

The two kinds of algorithms that are employed in the intelligent recommendation are cooperative filtering and content-based filtering. First, the five attributes of the students' resumes are obtained: city, educational background, major, salary and skills. Then, the matching degree of each dimension with the job requirements is calculated independently, and then a weighted score is obtained. City matching considers the relevance between the job location and the students' desired cities; educational background matching determines if the students' qualifications meet the minimum requirements for the position; major matching evaluates the degree of alignment of majors; salary matching analyzes the overlap between the expected salary and the salary range of the job; and skill matching is scored based on the degree of keyword overlap. Finally, all the scores of the dimensions are combined to get a matching percentage, and then sorted and used for recommendation.

3.2.3 API Interface Design

The backend exposes RESTful endpoints organised by domain. User-related endpoints are for registration, login and profile queries. Resume Endpoints handle creation, updates and retrieval. Job endpoints support listing, detail view and keyword search. Application endpoints are used to submit applications and query application results. Employment Endpoints record hiring results and return summarised statistics.

All endpoints use the same convention: standard HTTP methods, JSON request and response bodies, uniform error response structure. Not architecturally outstanding, but consistent with the other two client types to reduce integration friction.

3.3 Frontend Application Module

3.3.1 Route Configuration

Vue Router is responsible for front-end routing. Set up route guards for all pages that require authentication. If an unauthenticated user tries to access a protected route directly by entering the URL in the address bar, for instance, the guard will intercept the request and redirect to the login page. The three groups of Routes are student routes, enterprise routes and administrator routes.

3.3.2 State Management

Pinia Manages Global State. Login status, JWT tokens and resume data are all available to multiple components via Pinia stores. Thus, we do not need to pass props down many levels of nested components solely for sharing a piece of state.

3.3.3 Key Page Implementation

The main pages in the student area are: the home page showing recommended jobs, the job listing page with search and filtering, the job detail page presenting all information and an application button, and a personal centre page for resume modification and application history.

Enterprises primarily interact via the job management page (for posting creation, modification and deletion) and the consolidated view of all received applications on the application processing page.

Administrators have a large number of pages, including student management, enterprise evaluation, policy publication and a data dashboard that presents important indicators in charts and summaries on one page.

4. System Testing

4.1 Test Environment

The Test Environment is based on the local construction development environment. Back-end service operation: JDK 11 and Spring Boot 2.7 are used; the front-end runs on Node.js 16, and MySQL 8.0 is used as the database.

4.2 Functional Testing

Based on the actual use cases, we will divide the functional test cases and not test the modules separately. The three types of user operation paths are as follows: Student side, which includes registration and login; Resume completion, job search, view recommended positions, submit applications and track status; Company side, which covers job posting, application review, candidate screening and recruitment marking; and University side, which focuses on managing the back-end Dashboard view and data filtering. In each case, we manually created more than 20 test cases that covered the normal workflow and edge cases, such as submitting a resume without required fields, applying for the same position multiple times, or loading the dashboard without data. No

issues were found in any of the above links. We have verified that after adding new application records, the employment rate, industry distribution and average salary indices in the dashboard have been updated correctly and meet our requirements. The final Dashboard interface is as shown in Figure 4.

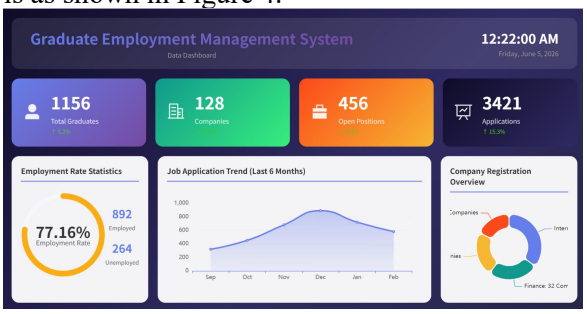

Figure 4. Data Dashboard First Page

4.3 Performance Testing

Apache JMeter (version 5.5) is used to perform a stress test of the two interfaces with the highest expected traffic: paging job list query interface and personalised recommendation API. The test case has continuously sent requests for over 15 minutes by 100 concurrent users. To avoid the impact of a temporary system anomaly, the experiment was repeated three times on different days. According to the above experiments, the average response time of the work list interface is around 380ms, and the average response time of the recommended interface is about 420ms; both are less than the 500ms limit we set [10]. At the same time, we have also observed that the CPU and memory usage of the back-end server during the stress test remained below 75%; therefore, it can still support a larger number of users.

5. Conclusion

Design and build an intelligent job-matching system for employment services for college graduates in this paper. A client-server architecture is used to separate the demonstration layer, business logic layer and data persistence layer effectively.

Vue 3 is used for the front end, and due to its reactive component model, it has been selected as the main library; Element Plus is employed as the component library to speed up development. Spring Boot 2.7 is used for the back-end, and MyBatis is employed to map objects to the database. In terms of the real-time recommendation and analysis pipeline, we have introduced three big data components: Kafka serves as the event collection layer to collect user clickstreams and application events; Flink is used for sliding-window aggregation to compute real-time feature vectors; and ClickHouse stores the summary results of low-latency dashboard queries. All services use Docker for containerisation and Docker Compose for organisation to ensure that the production environment can be replicated in the development, testing and pre-release stages. The three kinds of users of the platform are students, enterprises and university administrators; each has a personal dashboard and particular operating rights, and a five-dimensional weighted scoring model is used to recommend suitable positions for students. We have added a data visualisation dashboard for the university career centre, in addition to the recommendation function. Employment data by department and academic year for the entire institution are displayed in this dashboard.

In short, this study shows that by integrating a modern network framework and a streaming big data tool, an agile, high-capacity, and practical employment service platform can be built. We believe that this system will provide a feasible path for promoting the informatization of university employment management, and we plan to add mobile support and automatic interview scheduling functions in the future.

References

[1] Zhang J Y. Analysis of the Application of Vue Framework in Front-end Development. Information and Computer (Theoretical Edition), 2024, 36(13).

[2] Wei C Y. Design and Implementation of Smart Campus Management System Based on Spring Boot Framework. Changjiang Information and Communications, 2024, (2).

[3] Xu X, Chen Y J, Wang Y C, et al. A JWT Token-based Authentication and Authorization Scheme for Power System Microservices. Electric Engineering, 2021, (16).

[4] Liu T. Research on SK-means Strategy Based on Stream Processing Improvement. Journal of Beijing Information Science and Technology University (Natural Science Edition), 2021, 36(5).

[5] Chu J P, Ma X D. Practice of Real-time Data Warehouse Based on ClickHouse. Digital Communication World, 2023, (7).

[6] Wang S S. Research on Distributed

Database Coordination Technology—ZooKeeper. Science and Technology Outlook, 2016, (1).
[7] Zhang X Y. Research on Building Docker Private Repository Based on Harbor. Modern Industry and Economy Informatization, 2022, 12(9).
[8] Zhao Y S, Li Z D, Yang H Y. Design of Student Management System Based on Hybrid Recommendation Algorithm. Information Technology and Informatization, 2023, (6).
[9] Wang S S, Li J Q. Research on Automatic Software Quality Verification Method Based on Real Business Data. Telecommunications Science, 2024, 40(3).
[10] Dong J Q. Research and Implementation of Distributed MCPTT System. Beijing Jiaotong University, 2020.

