Software Requirement Specification Document for ResuFit: Precision CV Matching System

Omar Mohamed, Habeba Mohamed, Mohamed Ashraf, Donia Sameh
Supervised by: Eng:Reham Mohamed Kamal, Eng:Yasmin Kandil

April 30, 2024

Table 1: Document version history

Version	Date	Reason for Change
1.0	14-Jan-2024	SRS First version's specifications are defined.

GitHub: https://github.com/Omar2710/Gp_resufit.git

Contents

Abstract

ResuFit is a web-based application designed to ease the hiring and recruitment process by integrating natural language processing (NLP) using the SpaCy Python library to parse CVs and by integrating the HR module from Odoo ERP software. Applicants submit their CVs for parsing and analysis. The system uses the Python library for parsing the CV and extracting key information such as skills and years of experience, which are then compared to the job requirements provided by the recruiter. The similarity between the applicant's CV and the job vacancy is determined using cosine similarity and Jaccard similarity, resulting in a percentage rating that reflects the applicant's suitability for the position. To ensure a comprehensive evaluation, applicants are presented with screening questions defined by an expert recruiter. The system uses ChatGPT to validate and score applicants' responses by comparing them against model answers generated by the expert. By averaging CV ranking and screening question validation scores, the system generates a percentage-based result, providing a fair evaluation of each applicant.

1 Introduction

1.1 Purpose of this document

The goal of the Resufit SRS document is to illustrate detailed documentation of the Resufit project. Resufit is an applicant tracking system on which the applicant can submit their CV to be parsed, view and edit their parsed CV, and get a rating for their CV. Firstly, the purpose and scope of this document will be provided, followed by an overall explanation of the Resufit system. Moreover, system functionalities are as follows: submitting the CV, viewing and editing the CV, answering screening questions, and getting a rating for the submitted CV. Finally, the datasets used are our own CVs and job vacancies.

1.2 Scope of this document

The document's scope is to tackle similar systems to Resufit academically and business-wise; it also illustrates the system overview, context, and scope, as well as the objectives of the Resufit website and the user characteristics. Furthermore, this document goes through the functional and non-functional requirements of Resufit, the software and hardware limitations, the data design, and the object-oriented class diagram. Finally, this document also covers the operational scenarios of the system and the exact timeline of how this website will be developed.

1.3 Business Context

Statistics show that 90% of Fortune 500 companies use applicant tracking software [1]. ResuFit allows recruiters to match their job title with the applied job. Also, it removes obstacles for potential employees and increases visibility for internal job boards. The parsing process extracts key information based on what the recruiter has defined in the job description, such as personal skills, years of experience, and other criteria. The system utilizes cosine similarity and Jaccard to assess the alignment between the extracted information from the CV and the job requirements specified by the recruiter. Applicants are given screening questions that have been carefully designed by a professional recruiter to guarantee a thorough assessment. Applicant responses are verified and graded by comparing them to an expert-generated model, and the applicant's answers are validated and scored by ChatGPT. The percentage rating from CV parsing and the screening question validation scores gives each applicant a thorough and equitable assessment, which is provided by the system. A percentage-based result that represents the applicant's overall suitability for the role by averaging these scores.

2 Similar Systems

2.1 Academic

- **Irfan Ali** [2]: The objective of this study is to create an automated resume classification system (RCS) by employing approaches from natural language processing (NLP) and machine learning (ML) algorithms. The primary objective is to classify resumes according to job categories and enhance the efficiency of the classification process. To accomplish its goals, the study utilized a range of approaches, such as data gathering and visualization, preprocessing, feature engineering, model design, and model evaluation. Data pretreatment involved the utilization of several natural language processing (NLP) techniques, including word tokenization, stop word removal, stemming, and lemmatization. Various machine learning methods, including Support Vector Machine (SVM), K Nearest Neighbors (KNN), Logistic Regression, and Naive Bayes, were utilized to develop and evaluate the models. The dataset included in the research consisted of 962 resumes that had been processed and tagged, including 25 distinct career categories. The dataset was formatted as a Comma-Separated Values (CSV) file, containing three columns: ID, Category, and the text of the resumes. The research demonstrated notable improvements in the accuracy and performance of resume categorization. The classifier that had the highest performance, Support Vector Machine (SVM)-Linear SVC, achieved a true class prediction accuracy of over 98%. The study further conducted a comparison of several strategies for feature extraction and representation. The performance of these techniques was assessed using measures such as overall accuracy, F-Score, precision, and recall. Nevertheless, the study was subject to several limitations. A significant constraint encountered was the limited accessibility of an appropriate and standardized dataset for the purpose of training the machine learning models. The resumes exhibited varying forms and layouts, hence posing difficulties in accurately processing and categorizing them. The size of the dataset was rather small, potentially impacting the ability of the machine learning models to generalize. Furthermore, the primary emphasis of the study was placed on assessing total accuracy as a performance indicator, while neglecting the examination of alternative assessment metrics.

- **Pradeep Kumar Roy** [3]: This study aims to create a machine learning-driven model for automated resume recommendation, with the purpose of addressing the difficulties encountered in the recruiting procedure. The objective of the model is to categorize resumes into distinct classifications and provide recommendations for the most appropriate individuals depending on the job description. In order to accomplish their aims, the writers employed a range of methodologies. Machine learning-based categorization algorithms were utilized in order to identify the most pertinent resumes, employing similarity functions. The researchers also employed the Linear Support Vector Machine (SVM) classifier, which exhibited superior performance compared to other classifiers. Furthermore, the researchers employed the "textract" package to transform resumes of various forms, such as .doc or. PDF, into a unified format suitable for input into the model. The researchers also employed the "genism" library to generate summaries of the resumes. The dataset utilized in this study included resumes that were obtained from online portals as well as Kaggle. The dataset was presented in Excel format, comprising three columns: ID, Category (representing the industry sector), and Resume (containing the comprehensive curriculum vitae of the candidate). The findings of the investigation revealed that the suggested model attained a precision rate of 78.53% while employing the linear Support Vector Machine (SVM) classifier. This finding suggests that the model demonstrated effectiveness in accurately identifying and providing recommendations for appropriate resumes, utilizing the job description as a basis. Nevertheless, the report does acknowledge several limits or downsides. Initially, it is important to note that the present model exclusively supports resumes in CSV format. However, it is worth mentioning that in practical scenarios, resumes are commonly found in formats such as .doc or .pdf. One possible solution to address this constraint is to employ the "textract" package, which enables the conversion of many file formats into a unified format. Additionally, the utilization of the "genism" library for producing summaries may lead to the inadvertent omission of crucial information as a result of the compression process. The authors propose the refinement of the summary process in order to reduce the loss of information, especially in relation to significant aspects such as applicant abilities and experience.

- **Chirag Daryani** [4]: The main purpose of this study is to create a computerized resume screening system that streamlines the e-recruitment process through the extraction of pertinent details from resumes and their subsequent alignment with job descriptions. The primary objective of the system is to address the challenges encountered by recruiters while manually selecting candidates and to offer a screening procedure that is both more effective and equitable. In order to accomplish this goal, the present study outlines the utilization of Natural Language Processing (NLP) methodologies for the purpose of extracting information from resumes that lack standardized formatting. Various techniques, including tokenization, stemming, part-of-speech tagging, and named entity identification, are employed to extract job-related information such as skills, experience, education, and so forth. The information that has been retrieved is subsequently utilized in order to generate a condensed rendition of each curriculum vitae. During the second step, the system utilizes a vector space model to depict the resumes and job descriptions as vectors. The cosine similarity metric is employed as a means of quantifying the similarity of vectors, specifically in the context of comparing resumes to a given job description. This similarity measure is then utilized to rank the resumes depending on their degree of resemblance to the aforementioned job description. This rating system facilitates the identification of the most suitable candidates for the position. The dataset employed for the tests conducted in the publication is not explicitly stated. However, one might deduce that the system underwent testing by utilizing a pool of resumes and job descriptions to assess its efficacy in aligning individuals with job prerequisites. The findings derived from the experimental utilization of the system are outlined in Section 5 of the manuscript. Nevertheless, the precise particulars of the outcomes are not specified within the provided context. The potential limits or disadvantages of their research are not addressed within the provided context.

- **Hussain**[5]: The study illustrates the use of a novel similarity metric for Pythagorean fuzzy sets (PFSs) that is based on the Jaccard index (JI) in clustering. The authors want to correct the lack of attention in the literature that has already been written about a similarity measure between PFSs based on the Jaccard index. Additionally, they want to demonstrate the superiority and dependability of their suggested similarity measure using numerical examples and its use in clustering and multi-criteria decision-making.

  To accomplish their goals, the authors created a new similarity metric based on the Jaccard index and applied it to Pythagorean fuzzy sets using statistical and mathematical techniques. To prove the dependability and superiority of their suggested method over the current ones, they also carried out numerical analyses. They also used the suggested similarity measure for clustering based on Pythagorean fuzzy data sets and for multicriteria decision-making utilizing the Pythagorean Fuzzy Technique of Order Preference by Similarity to Ideal Solution (PF-TOPSIS).

  The context given does not specifically specify the dataset they used for their research. Nonetheless, it may be assumed that the authors illustrated the applicability and dependability of their suggested similarity measure using numerical examples and Pythagorean fuzzy data sets.

  The authors' accomplishments include the creation of a brand-new similarity metric for PFSs based on the Jaccard index, its effective use in multi-criteria decision-making and clustering, and the numerical examples provided to show its superiority and dependability.

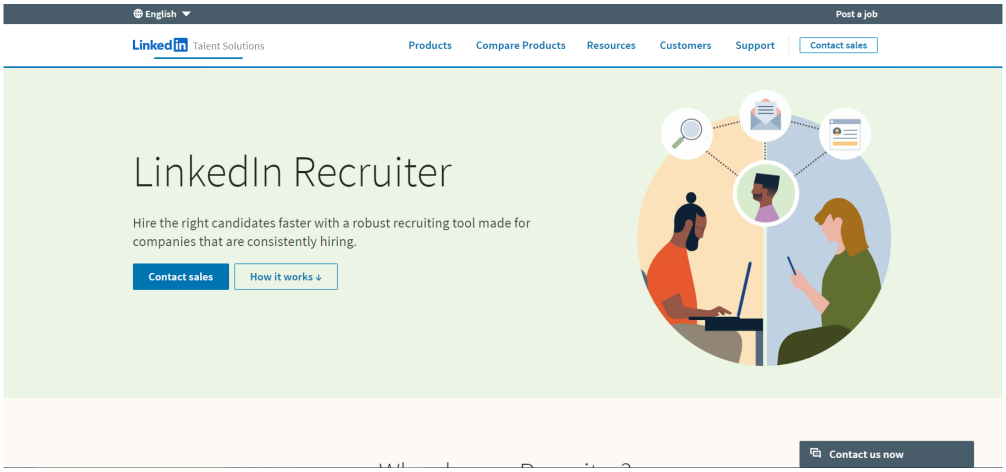
  Regarding restrictions or downsides, the author's work is not specifically mentioned in the context that is given. It is crucial to remember that additional validation may be required to ensure the suggested similarity measure is applicable and generalizable across various dataset types and real-world applications. Further research may also be possible in the domains of the suggested method's scalability and computational efficiency.

- **A Ly**[6]: The aim of this study is to elucidate the potential prospects that may be harnessed by the insurance business via the utilization of natural language processing (NLP). The primary objective of the writers is to comprehensively elucidate various methodologies employed in practical settings, while also furnishing instances that exemplify the application of these strategies to insurance goods or services. This study additionally centers its attention on the utilization of natural language processing (NLP) models through the utilization of open-source libraries and Python scripts that have been produced by the SCOR Data Analytics team. In order to accomplish their goals, the authors employ a range of technologies, including text mining, natural language processing (NLP) models, and machine learning techniques. The authors elucidate the many stages included in the manipulation of textual data and furnish instances illustrating the practical use of these methodologies. The research moreover examines the utilization of cutting-edge natural language processing (NLP) models and their capacity to record word interactions and adjust attention dependent on the sentence's context. The specific dataset employed in the research is not clearly specified. Nevertheless, the utilization of Twitter's Tweet for Sentiment Extraction Dataset is referenced by the authors in several demonstrations and examples. The study primarily emphasizes the elucidation of methodology and techniques employed in Natural Language Processing (NLP) within the insurance business, rather than presenting particular outcomes or conclusions. This study offers valuable insights into the possible advantages and prospective uses of Natural Language Processing (NLP) in diverse insurance procedures. The offered context does not include any discussion about the limits or downsides of the work in question.

- **Kadambari Wailthar**[7]: This project aims to create and execute a resume screening system for job recruitment. Cosine similarity is used to rate resumes based on their resemblance to job requirements. System main aim to improve hiring productivity using automated resumes Screening strategies help identify the best candidates for job openings. Authors used numerous methods to achieve their goals. Data preprocessing methods were used to clean and translate job descriptions and resumes into feature vectors. Assessments used cosine similarity. Degree of match between job requirements and candidate credentials. The system ranks resumes by cosine similarity and presents them to HR managers for further evaluation. System implementation involved web application development using Python and Flask. The report does not provide the dataset used for experiments. However, the system's accuracy is largely impacted by the quality and diversity of training data. A dataset of job descriptions and resumes is considered for system training and evaluation. The proposed system yields promising results. The testing results showed that the system outperformed standard resume screening methods in accuracy and speed. The algorithm achieved 86% accuracy in resume assessment using cosine similarity. The automatic approach can quickly and accurately rank resumes, making it more efficient than manual screening. Despite the system's successes, the article lists its drawbacks. The quality and diversity of resumes and job descriptions used to train the algorithm may limit the screening procedure's accuracy. Prejudice and discrimination are other restrictions. During screening if the system is poorly designed and implemented. Furthermore, it is worth emphasizing that the machine can understand language's subtleties. may not meet job criteria.

2.2 Business Applications

- **LinkedIn**[8]: LinkedIn Recruiter offers features such as job postings, applicant tracking, and team collaboration tools to enable successful and efficient recruiting.

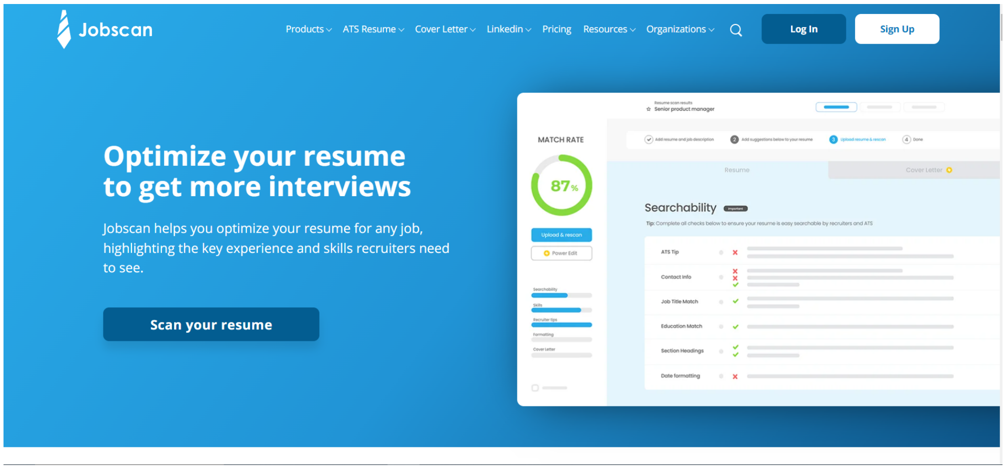


LinkedIn Recruiter

Hire the right candidates faster with a robust recruiting tool made for companies that are consistently hiring.

Figure 1: LinkedIn

- **JobScan**[9]: Jobscan helps optimizing the resumes keywords to fit the job position by emphasizing the most relevant experience and abilities that recruiters are looking for.



Optimize your resume to get more interviews

Jobscan helps you optimize your resume for any job, highlighting the key experience and skills recruiters need to see.

Resume scan results
☆ Senior product manager

MATCH RATE 87%

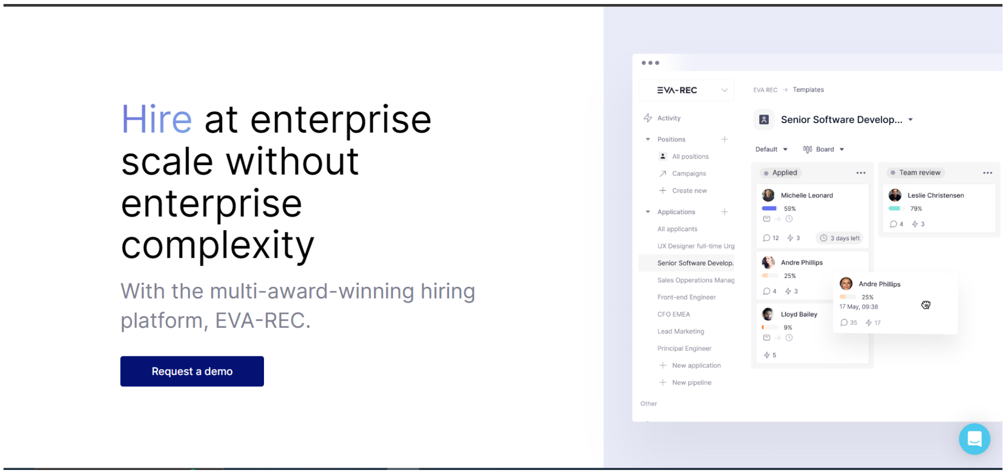
Searchability

Tip: Complete all checks below to ensure your resume is easy searchable by recruiters and ATS

ATS Tip	✗
Contact Info	✗ ✗ ✓
Job Title Match	✓
Education Match	✓
Section Headings	✓ ✓
Date formatting	✗

Figure 2: JobScan

- **Elevatus**[10]: Elevatus is leading the way in automating workflows and reintroducing document management. Elevatus streamline your whole hiring process, from job requisition creation to onboarding new applicant.



Hire at enterprise scale without enterprise complexity

With the multi-award-winning hiring platform, EVA-REC.

Figure 3: Elevatus

3 System Description

3.1 Problem Statement

Time-consuming manual activities and inefficiency are common problems for traditional hiring processes. Without a clear and standardized screening procedure, determining which applicants are the most qualified becomes a difficult assignment. In the absence of a systematic assessment process, it can be challenging to ensure objective evaluations, which makes it more difficult to improve CV ranking and expedite the hiring process. A complete solution is needed to handle these problems and speed up decision-making. Utilizing the most recent technological advancements, this solution should streamline and enhance the hiring procedure, guaranteeing a more impartial and effective assessment of applicants.

3.2 System Overview

As shown in figure 4, the applicant logs in to the system and submits his CV. Secondly, the NLP model, which is the SpaCy Python library, parses the CV and allows applicants to edit their parsing. Moreover, it then gets compared with the parsed job vacancy that is written by the recruiter, where the system then rates the parsed CV against the job vacancy criteria using the cosine similarity rule and Jaccard similarity, giving it a percentage of similarity. Furthermore, the system allows applicants to answer screening questions that are written by the recruiter. Additionally, Chat GPT validates it with model answers written by the recruiter, giving it a score. Finally, the final ranking result is calculated using the average of both CV ranking and screening questions validation and is validated by an HR expert.

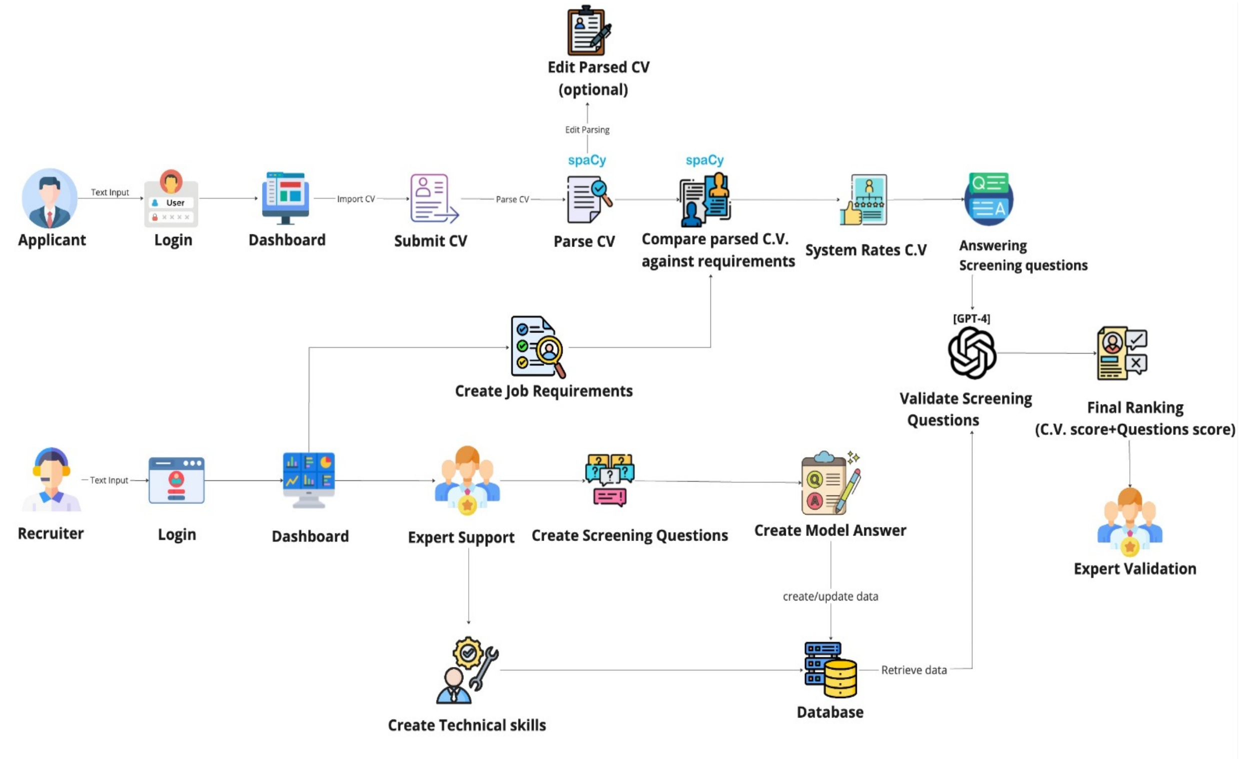


Figure 4: System overview

3.3 System Scope

- Allow applicants to know if they are qualified for a certain job or not.
- Provide a set of screening questions to the applicants.
- Give the provided CV a precise ranking.
- Display the final result to recruiters for decision-making purposes.

3.4 System Context

As shown in figure 5, the applicant will be able to upload CVs to be compared to job requirements added by the recruiter, and the final score will be shown to the applicant at the end. Moreover, the recruiter will be able to create screening questions, answers, and job requirements. Spacy Python, which is an open-source software library for advanced natural language processing written in the programming language Python, will parse and give rank to the CV uploaded by the applicant. Moreover, Chat GPT compares the model answer given by the recruiter with the answer of the applicant and then returns the score.

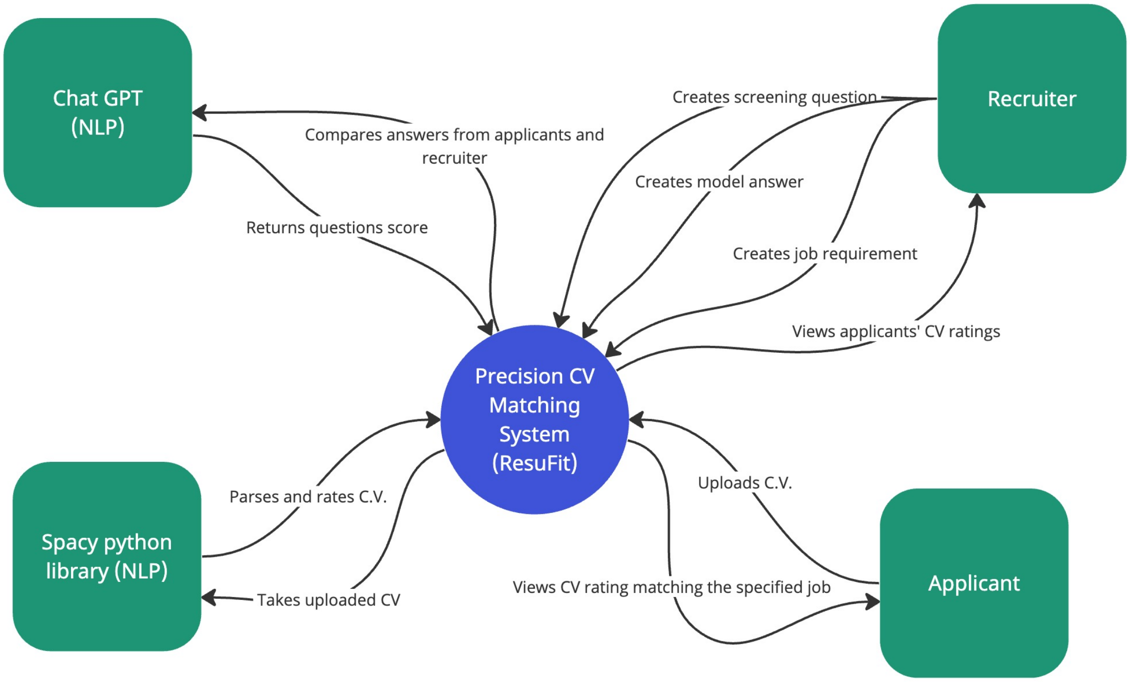

Chat GPT (NLP)
Recruiter
Creates screening question
Compares answers from applicants and recruiter
Creates model answer
Returns questions score
Creates job requirement
Views applicants' CV ratings
Precision CV Matching System (ResuFit)
Spacy python library (NLP)
Parses and rates C.V.
Takes uploaded CV
Uploads C.V.
Views CV rating matching the specified job
Applicant

Figure 5: Context diagram

3.5 Objectives

- Reduce administrative overhead and time-to-hire by streamlining the hiring process by offering a platform for quick applicant CV submission, parsing, and review.
- Provide a complete assessment of every applicant by merging CV ranking and screening question validation.
- Verify applicant answers to screening questions with ChatGPT to ensure a consistent and fair assessment of all applicants.
- Provide a user-friendly interface that is simple to understand and use for both recruiters and applicants.
- Reduce manual efforts in the applicant assessment process so that recruiters may concentrate on strategic decision-making rather than routine tasks.
- Recruiters will be able to find applicants whose resumes closely match the job requirements by using the cosine and jaccard similarity methods, which provides an objective CV ranking.
- Integrate this system with the HR module using the Odoo ERP system.

3.6 User Characteristics

- Since English will be the user interface's default language, the user must have a basic understanding of the language.
- Standard users who are at least eighteen years old, regardless of gender, can use the website.
- ResuFit does not require any specific advanced computer knowledge to use it, except for the developers and administrators of the system.
- Recruiters and hiring managers can use the ResuFit website to track applicants during the recruiting process and manage applications.
- Job seekers can use the ResuFit website to submit their CVs for a certain job.

4 Functional Requirements

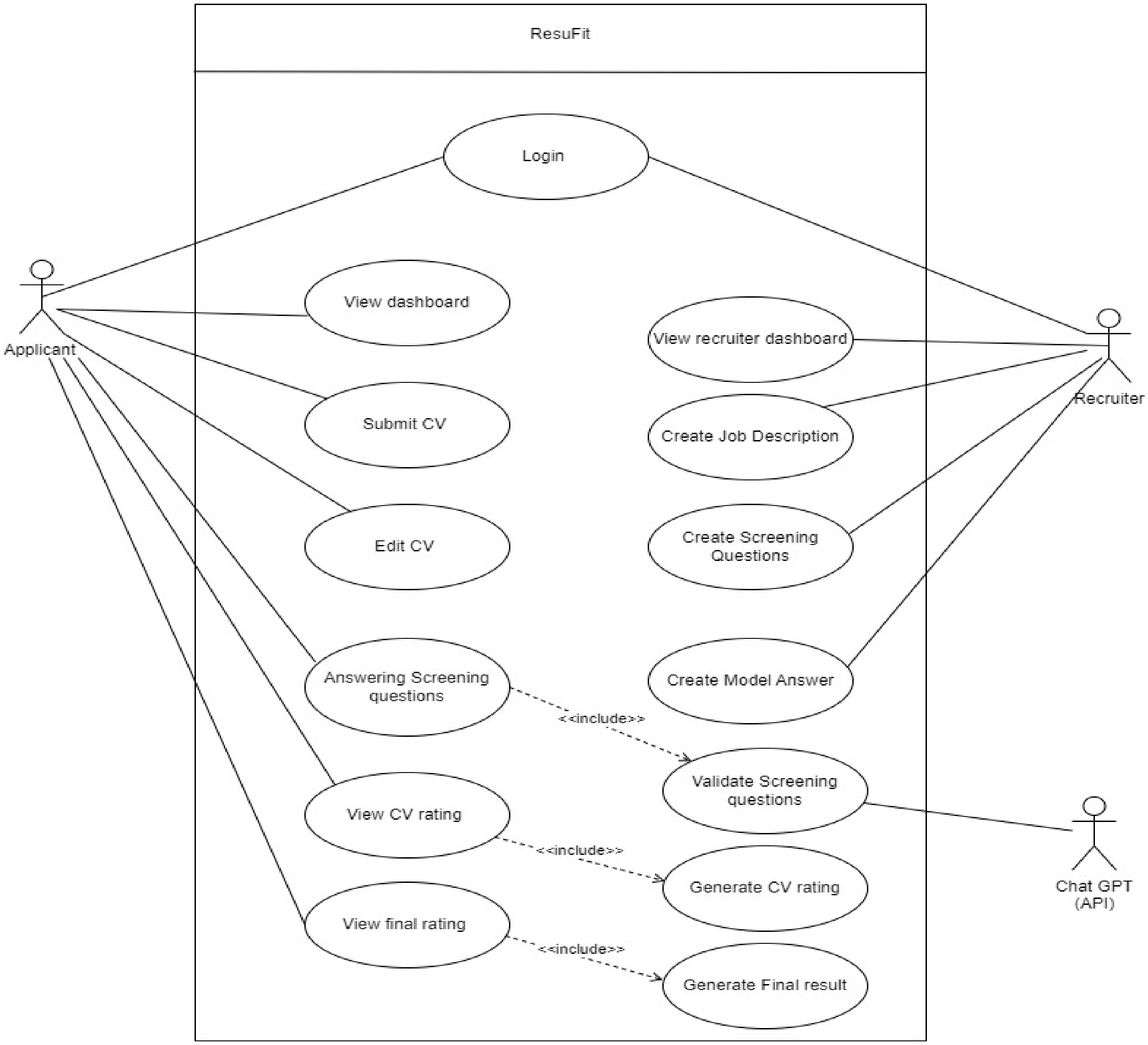


Figure 6: Use-Case diagram

4.1 System Functions

- The system shall use ChatGPT to verify the applicant's answers against model answers for screening questions and record the outcomes for every screening question.
- The system shall create an algorithm that takes the average of the CV ranking and the validation of the screening questions to determine the outcome and present the result as a total score that represents the applicant's overall compatibility.
- The system shall utilize the Spacy Python library to accurately extract relevant CV content using natural language processing.
- The system shall create an approach that uses cosine similarity and Jaccard similarity to compare the parsed CV and the required job vacancy to determine the degree of fit and provide a percentage rating of the applicant's suitability.
- Recruiters shall be able to provide job vacancies as well as the technical stack, skills, and experience for each job position.
- Recruiters shall be able to assign each job position a set of screening questions.
- Recruiters shall be able to assign each screening question its model answer.
- The applicants shall be able to create an account and log in to the system with their personal information.
- The applicants shall submit their CVs.
- The applicants shall view the parsing results and edit them.
- The applicants shall edit the profile by re-uploading the CV.
- The applicants shall answer the provided screening questions.
- The applicants shall view the CV rating.
- The applicants shall view the final ranking which are CV score and questions score.

4.2 Detailed Functional Specification

Table 2: Sign-Up

Name:	**Sign-Up**
Code:	Fn 1
Priority:	Extreme
Critical:	The data entered by the user must be checked to see if it has any errors.
Description:	It checks the data of the user before being sent to the sign-up to be added to the database.
Input:	Email, Password, First and last name, and resume.
Output:	A string that prints the type of error that occurred.
Pre-condition:	None
Post-condition:	If the data has no errors, send the applicant to Login.
Dependency:	None
Risk:	None

Table 3: Login

Name:	**Login**
Code:	Fn 2
Priority:	Extreme
Critical:	An active and reliable internet connection is essential for users to access all services on the website.
Description:	It searches in a database for the entered username and password to see if they're valid or not.
Input:	Email and Password
Output:	Boolean(found or not found)
Pre-condition:	The applicant must already have created an account.
Post-condition:	If found, go to the homepage. If not, say that the email or password is incorrect.
Dependency:	Fn 1
Risk:	A previous session didn't end.

Table 4: Submit CV

Name:	**Submit CV**
Code:	Fn 3
Priority:	Extreme
Critical:	None
Description:	It allows the applicant to upload their CVs to be ranked.
Input:	CV
Output:	Ranked CV
Pre-condition:	The applicant must upload a precise CV that matches the job description.
Post-condition:	CV gets Ranked
Dependency:	Fn 2
Risk:	None

Table 5: Edit CV

Name:	**Edit CV**
Code:	Fn 4
Priority:	Medium
Critical:	None.
Description:	It allows the applicant to edit the CV that has been submitted.
Input:	CV.
Output:	The edited CV.
Pre-condition:	The applicant must be logged in.
Post-condition:	The CV was updated successfully.
Dependency:	Fn 3
Risk:	None

Table 6: Create job vacancy

Name:	**Create job vacancy**
Code:	Fn 5
Priority:	Extreme
Critical:	Open job vacancy
Description:	The recruiter will create a job vacancy according to the newly opened job vacancy.
Input:	Job description, Years of experience, Language, Education.
Output:	Job requirements.
Pre-condition:	None
Post-condition:	Comparing the job vacancy with the parsed CV.
Dependency:	None
Risk:	None

Table 7: Create screening questions

Name:	**Create screening questions**
Code:	Fn 6
Priority:	Extreme
Critical:	Open Screening questions
Description:	The recruiter will create a screening question.
Input:	Questions, key answers
Output:	Final rating (CV score + Questions Score)
Pre-condition:	CV ranking
Post-condition:	Final rating
Dependency:	Fn5
Risk:	None

Table 8: Parsing CV

Name:	**Parsing CV**
Code:	Fn 7
Priority:	Extreme
Critical:	Submit CV
Description:	Extracts keywords from the CV using SpaCy.
Input:	CV
Output:	Parsed CV
Pre-condition:	Submit CV
Post-condition:	None
Dependency:	Fn 3
Risk:	Lack of Context Understanding

Table 9: CV Ranking

Name:	**CV Ranking**
Code:	Fn 8
Priority:	Extreme
Critical:	Parsed CV
Description:	CV gets ranked using methods of similarities
Input:	Parsed CV
Output:	Ranked CV
Pre-condition:	CV gets parsed.
Post-condition:	None
Dependency:	Fn 7
Risk:	None

5 Design Constraints

5.1 Standards Compliance

The ResuFit website can work with online browsers such as Edge, Firefox, Safari, and Chrome.

5.2 Hardware Limitations

The user needs a PC that functions to upload their CV.

5.3 Network Constraint

For the website to function properly, a reliable internet connection is required.

6 Non-functional Requirements

These non-functional guidelines describe the characteristics and qualities that support the general effectiveness, dependability, and security of ResuFit. They serve as the main goals for evaluating the system's performance and ensuring a dependable and understandable solution.

6.1 Performance:

- Response Time: To ensure a smooth and efficient user experience, the system must respond to user input in less than two seconds.
- Scalability: The system must support at least 100 concurrent users without exhibiting a discernible decrease in performance.

6.2 Reliability:

- Availability: The system should be up and running 99.9% of the time with scheduled downtime that is communicated to users and administrators in advance.
- Fault Tolerance: Unexpected errors should be handled by the system, and users should get informative error messages.

6.3 Security:

- Data Encryption: Encrypting all data communications, including CV submissions and validation results, requires the use of industry-standard protocols.

6.4 Privacy:

- Data Privacy: To guarantee the confidentiality of user, CVS, and evaluation data, the system needs to adhere to data protection regulations.

6.5 Scalability:

- Database Scalability: The database must be designed to grow along with the volume of CVS and assessment data that are gathered.

6.6 Usability:

- User Interface Clarity: The user interface must be easy to use and give applicants and recruiters clear instructions and help.

6.7 Maintainability:

- Code Maintainability: Code should be modular, follow best practices for coding, and have extensive documentation to facilitate maintenance.

6.8 Compatibility:

- Cross-browser compatibility: The majority of web browsers, including Edge, Firefox, Safari, and Chrome, ought to work with the system.

6.9 Integration:

- Third-Party Integration: The system ought to have the ability to interface with external services, such as ChatGPT, to verify the screening questions.
- API Documentation: The documentation for each API that the system exposes or utilizes will be supplied.

7 Data Design

In this system, we used two datasets, which are the CVs and the job vacancy from a company. Here in figure 7, the database for ResuFit's website is demonstrated.

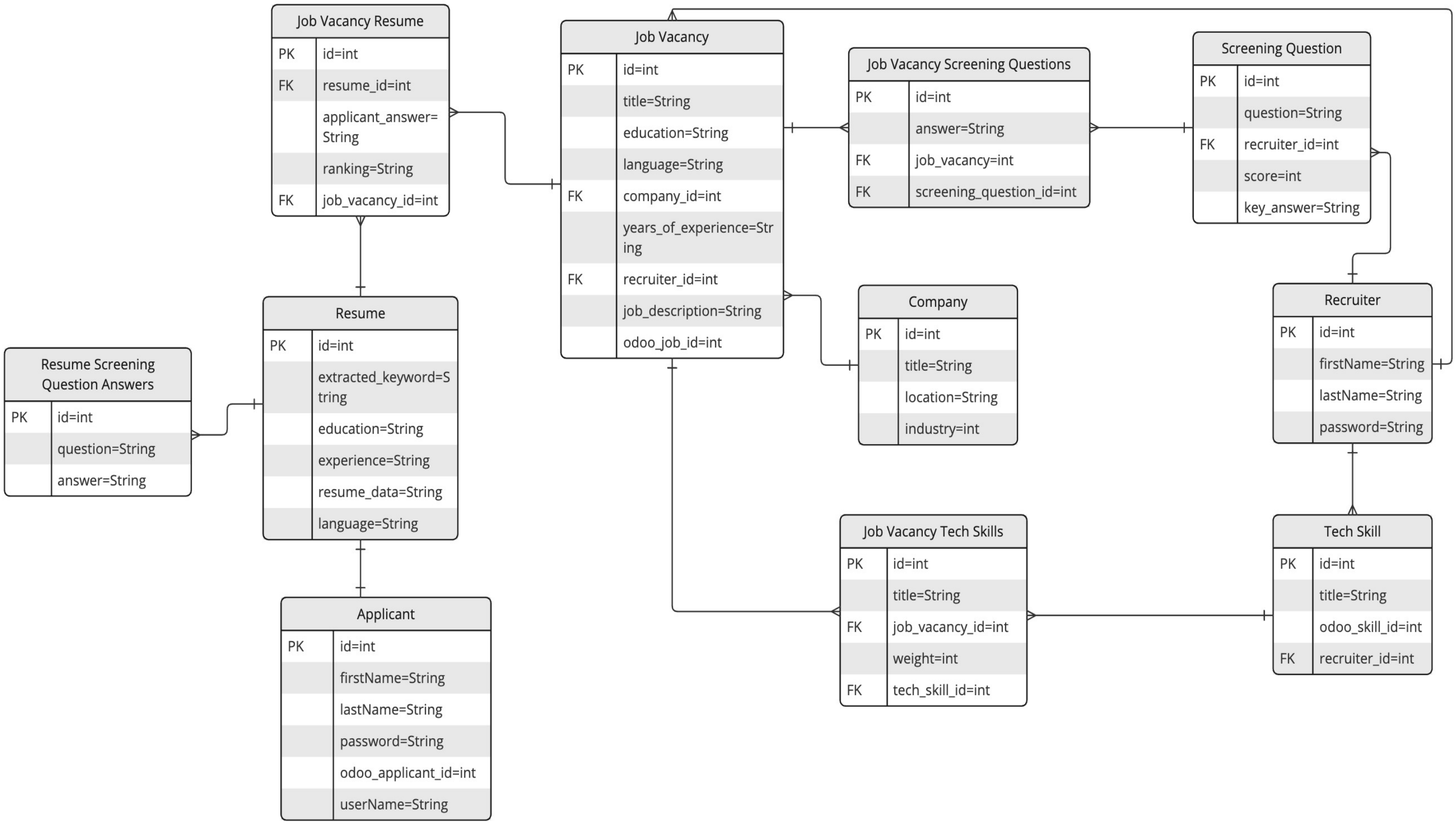
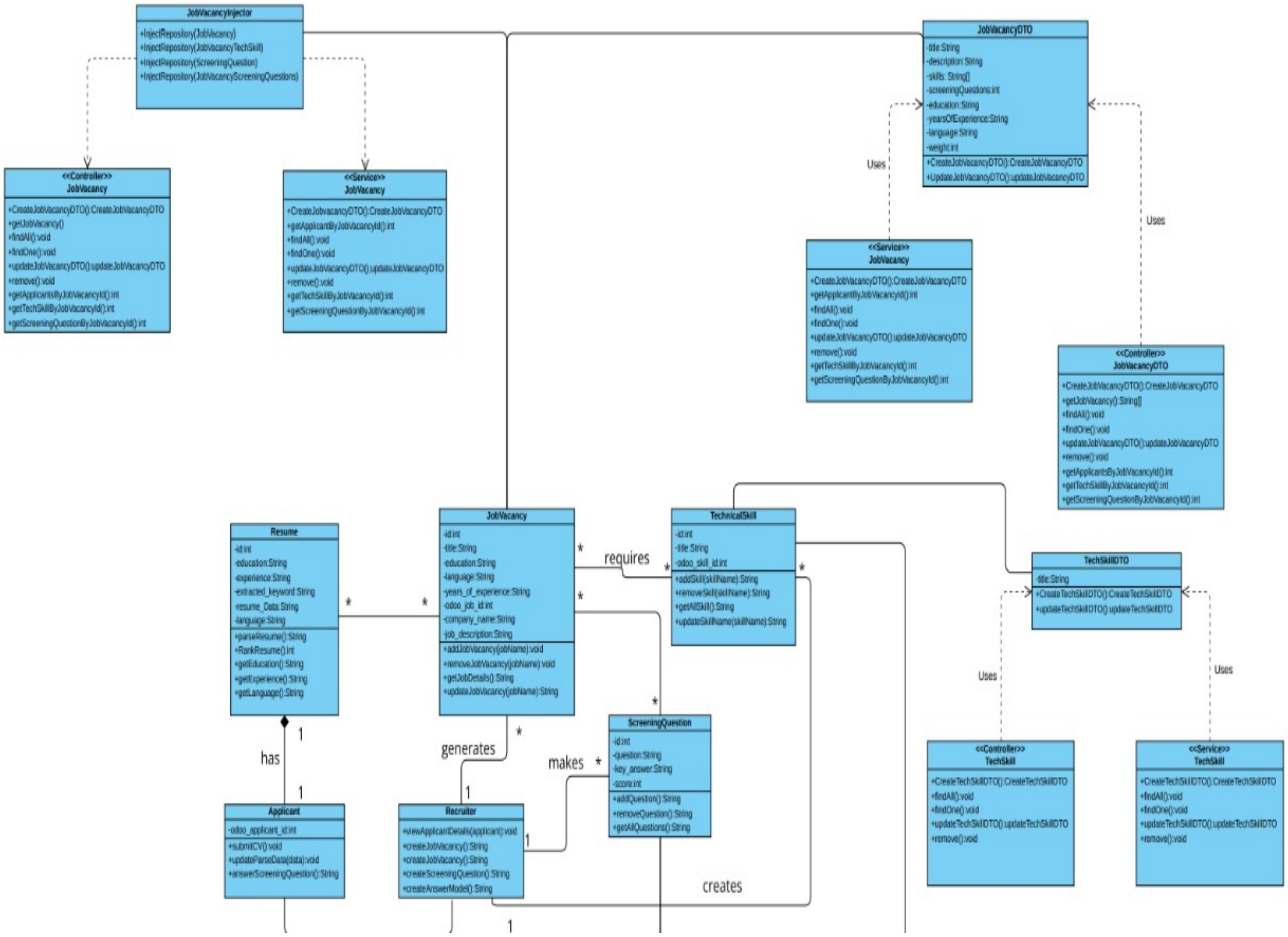


Figure 7: ResuFit's database

8 Preliminary Object-Oriented Domain Analysis



JobVacancyInjector
+InjectRepository(JobVacancy)
+InjectRepository(JobVacancyTechSkill)
+InjectRepository(ScreeningQuestion)
+InjectRepository(JobVacancyScreeningQuestions)
<<Controller>>
JobVacancy
+CreateJobVacancyDTO():CreateJobVacancyDTO
+getJobVacancy()
+findAll():void
+findOne():void
+updateJobVacancyDTO():updateJobVacancyDTO
+remove():void
+getApplicantsByJobVacancyId():int
+getTechSkillByJobVacancyId():int
+getScreeningQuestionByJobVacancyId():int
<<Service>>
JobVacancy
+CreateJobvacancyDTO():CreateJobVacancyDTO
+getApplicantByJobVacancyId():int
+findAll():void
+findOne():void
+updateJobVacancyDTO():updateJobVacancyDTO
+remove():void
+getTechSkillByJobVacancyId():int
+getScreeningQuestionByJobVacancyId():int
JobVacancyDTO
-title:String
-description:String
-skills: String[]
-screeningQuestions:int
-education:String
-yearsOfExperience:String
-language:String
-weight:int
+CreateJobVacancyDTO():CreateJobVacancyDTO
+UpdateJobVacancyDTO():updateJobVacancyDTO
Uses
Uses
<<Service>>
JobVacancy
+CreateJobVacancyDTO():CreateJobVacancyDTO
+getApplicantByJobVacancyId():int
+findAll():void
+findOne():void
+updateJobVacancyDTO():updateJobVacancyDTO
+remove():void
+getTechSkillByJobVacancyId():int
+getScreeningQuestionByJobVacancyId():int
<<Controller>>
JobVacancyDTO
+CreateJobVacancyDTO():CreateJobVacancyDTO
+getJobVacancy():String[]
+findAll():void
+findOne():void
+updateJobVacancyDTO():updateJobVacancyDTO
+remove():void
+getApplicantsByJobVacancyId():int
+getTechSkillByJobVacancyId():int
+getScreeningQuestionByJobVacancyId():int
Resume
-id:int
-education:String
-experience:String
-extracted_keyword:String
-resume_Data:String
-language:String
+parseResume():String
+RankResume():int
+getEducation():String
+getExperience():String
+getLanguage():String
*
*
JobVacancy
-id:int
-title:String
-education:String
-language:String
-years_of_experience:String
-odoo_job_id:int
-company_name:String
-job_description:String
+addJobVacancy(jobName):void
+removeJobVacancy(jobName):void
+getJobDetails():String
+updateJobVacancy(jobName):String
*
requires
*
*
TechnicalSkill
-id:int
-title:String
-odoo_skill_id:int
+addSkill(skillName):String
+removeSkill(skillName):String
+getAllSkill():String
+updateSkillName(skillName):String
*
TechSkillDTO
-title:String
+CreateTechSkillDTO():CreateTechSkillDTO
+updateTechSkillDTO():updateTechSkillDTO
Uses
Uses
<<Controller>>
TechSkill
+CreateTechSkillDTO():CreateTechSkillDTO
+findAll():void
+findOne():void
+updateTechSkillDTO():updateTechSkillDTO
+remove():void
<<Service>>
TechSkill
+CreateTechSkillDTO():CreateTechSkillDTO
+findAll():void
+findOne():void
+updateTechSkillDTO():updateTechSkillDTO
+remove():void
1
has
1
generates
*
1
makes
*
*
ScreeningQuestion
-id:int
-question:String
-key_answer:String
-score:int
+addQuestion():String
+removeQuestion():String
+getAllQuestions():String
Applicant
-odoo_applicant_id:int
+submitCV():void
+updateParseData(data):void
+answerScreeningQuestion():String
Recruiter
+viewApplicantDetails(applicant):void
+createJobVacancy():String
+createJobVacancy():String
+createScreeningQuestion():String
+createAnswerModel():String
1
creates
1

Figure 8: Class Diagram part 1

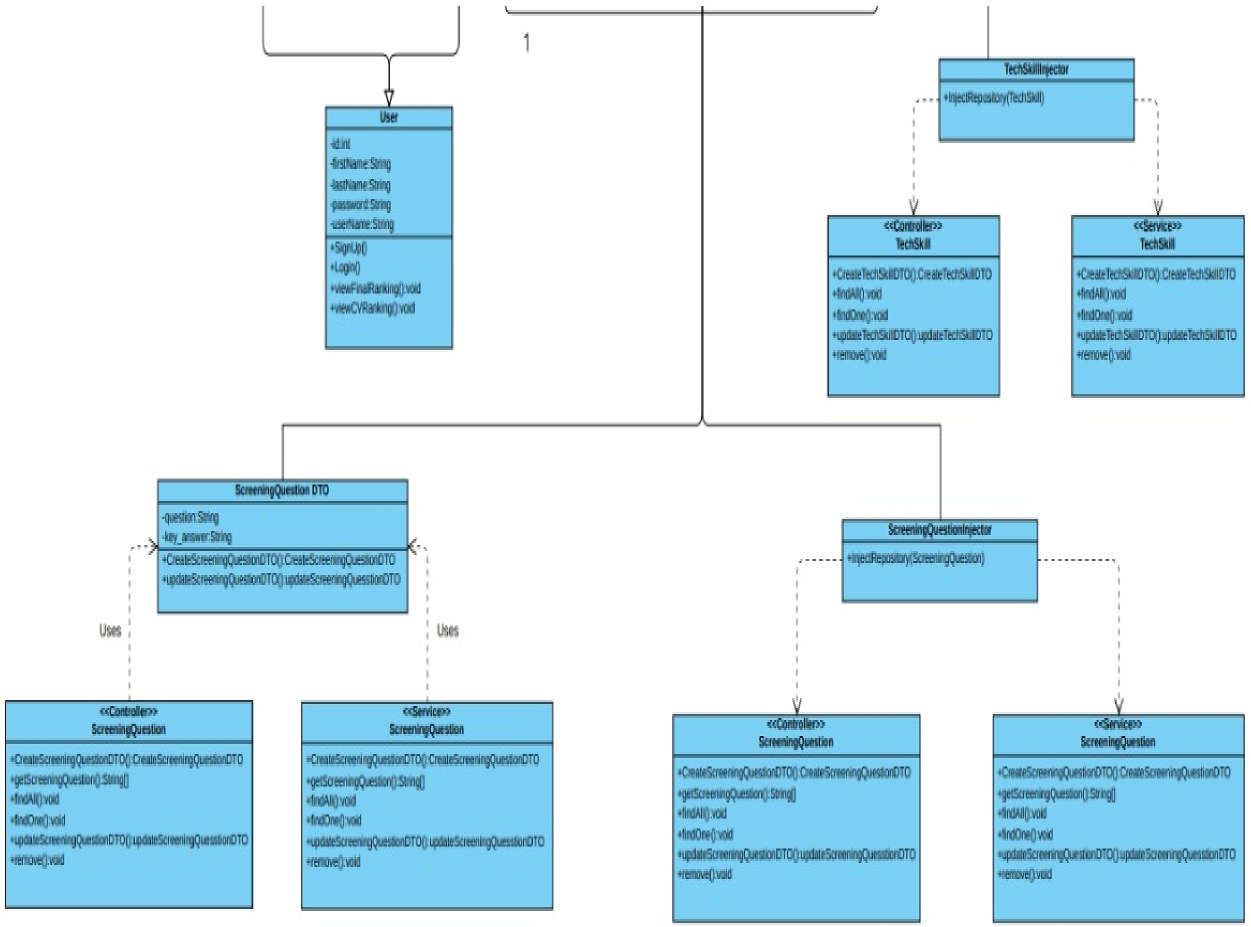

1
User
-id:int
-firstName:String
-lastName:String
-password:String
-userName:String
+SignUp()
+Login()
+viewFinalRanking():void
+viewCVRanking():void
TechSkillInjector
+InjectRepository(TechSkill)
<<Controller>>
TechSkill
+CreateTechSkillDTO():CreateTechSkillDTO
+findAll():void
+findOne():void
+updateTechSkillDTO():updateTechSkillDTO
+remove():void
<<Service>>
TechSkill
+CreateTechSkillDTO():CreateTechSkillDTO
+findAll():void
+findOne():void
+updateTechSkillDTO():updateTechSkillDTO
+remove():void
ScreeningQuestion DTO
-question:String
-key_answer:String
+CreateScreeningQuestionDTO():CreateScreeningQuestionDTO
+updateScreeningQuestionDTO():updateScreeningQuesstionDTO
Uses
Uses
ScreeningQuestionInjector
+InjectRepository(ScreeningQuestion)
<<Controller>>
ScreeningQuestion
+CreateScreeningQuestionDTO():CreateScreeningQuestionDTO
+getScreeningQuestion():String[]
+findAll():void
+findOne():void
+updateScreeningQuestionDTO():updateScreeningQuesstionDTO
+remove():void
<<Service>>
ScreeningQuestion
+CreateScreeningQuestionDTO():CreateScreeningQuestionDTO
+getScreeningQuestion():String[]
+findAll():void
+findOne():void
+updateScreeningQuestionDTO():updateScreeningQuesstionDTO
+remove():void
<<Controller>>
ScreeningQuestion
+CreateScreeningQuestionDTO():CreateScreeningQuestionDTO
+getScreeningQuestion():String[]
+findAll():void
+findOne():void
+updateScreeningQuestionDTO():updateScreeningQuesstionDTO
+remove():void
<<Service>>
ScreeningQuestion
+CreateScreeningQuestionDTO():CreateScreeningQuestionDTO
+getScreeningQuestion():String[]
+findAll():void
+findOne():void
+updateScreeningQuestionDTO():updateScreeningQuesstionDTO
+remove():void

Figure 9: Class Diagram part 2

9 Operational Scenarios

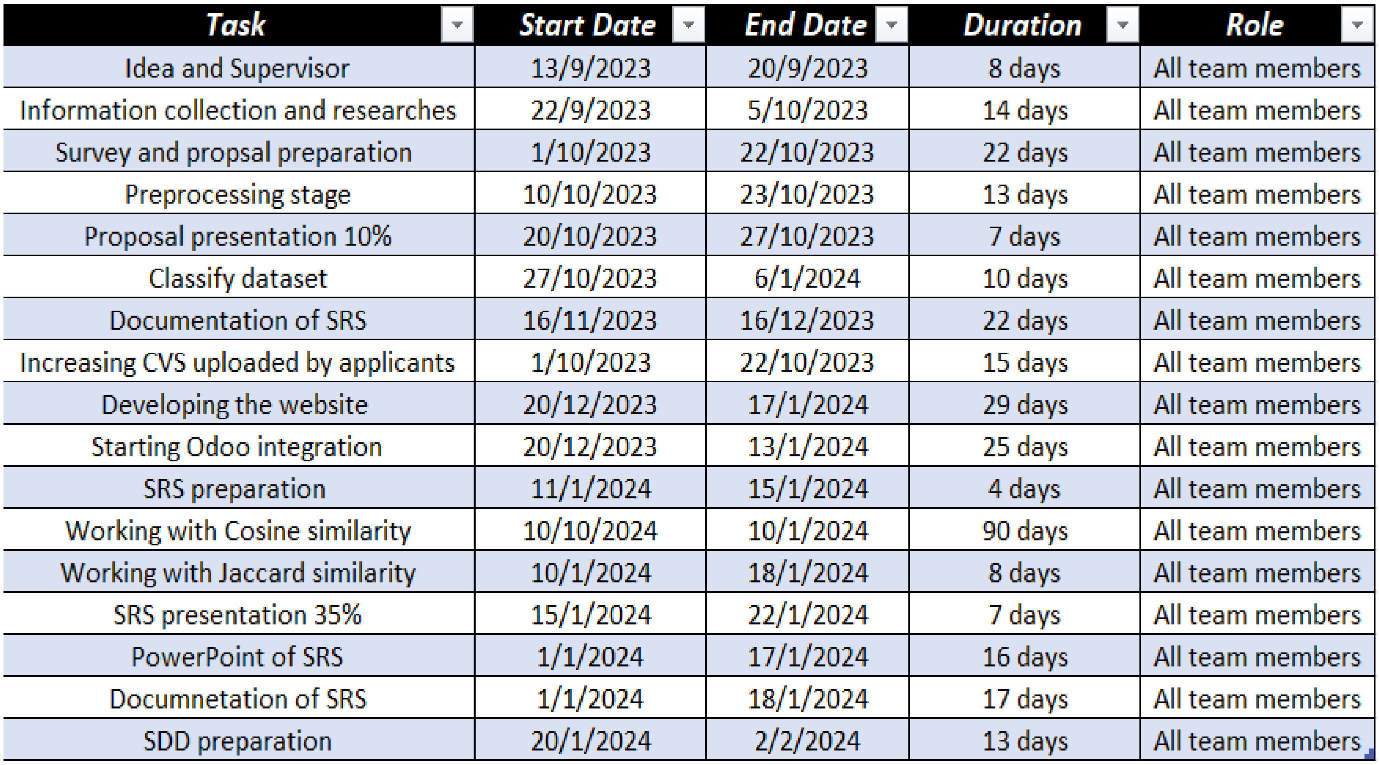
- **Scenario 1:** The applicant will log in to have an account and be able to use the system. First, the applicant has to upload a CV to have it parsed. Second, the system compares it with the specified job description and gives it a ranking. Third, the applicant answers the screening questions. Finally, the applicant gets a full ranking on his CV and questions score.

- **Scenario 2:** The applicant will also edit the parsed CV data after uploading it. And if the applicant wants to update his profile, he will resubmit his CV.

- **Scenario 3:** The recruiter will log in to have access to the system, create required job vacancies, create screening questions as well as model answers for these questions, and view the applicant's final reports.

10 Project Plan

Figure 10 below is the timeline of this project, from the end of the proposal to SDD.



Task	Start Date	End Date	Duration	Role
Idea and Supervisor	13/9/2023	20/9/2023	8 days	All team members
Information collection and researches	22/9/2023	5/10/2023	14 days	All team members
Survey and propsal preparation	1/10/2023	22/10/2023	22 days	All team members
Preprocessing stage	10/10/2023	23/10/2023	13 days	All team members
Proposal presentation 10%	20/10/2023	27/10/2023	7 days	All team members
Classify dataset	27/10/2023	6/1/2024	10 days	All team members
Documentation of SRS	16/11/2023	16/12/2023	22 days	All team members
Increasing CVS uploaded by applicants	1/10/2023	22/10/2023	15 days	All team members
Developing the website	20/12/2023	17/1/2024	29 days	All team members
Starting Odoo integration	20/12/2023	13/1/2024	25 days	All team members
SRS preparation	11/1/2024	15/1/2024	4 days	All team members
Working with Cosine similarity	10/10/2024	10/1/2024	90 days	All team members
Working with Jaccard similarity	10/1/2024	18/1/2024	8 days	All team members
SRS presentation 35%	15/1/2024	22/1/2024	7 days	All team members
PowerPoint of SRS	1/1/2024	17/1/2024	16 days	All team members
Documnetation of SRS	1/1/2024	18/1/2024	17 days	All team members
SDD preparation	20/1/2024	2/2/2024	13 days	All team members

Figure 10: Time Plan

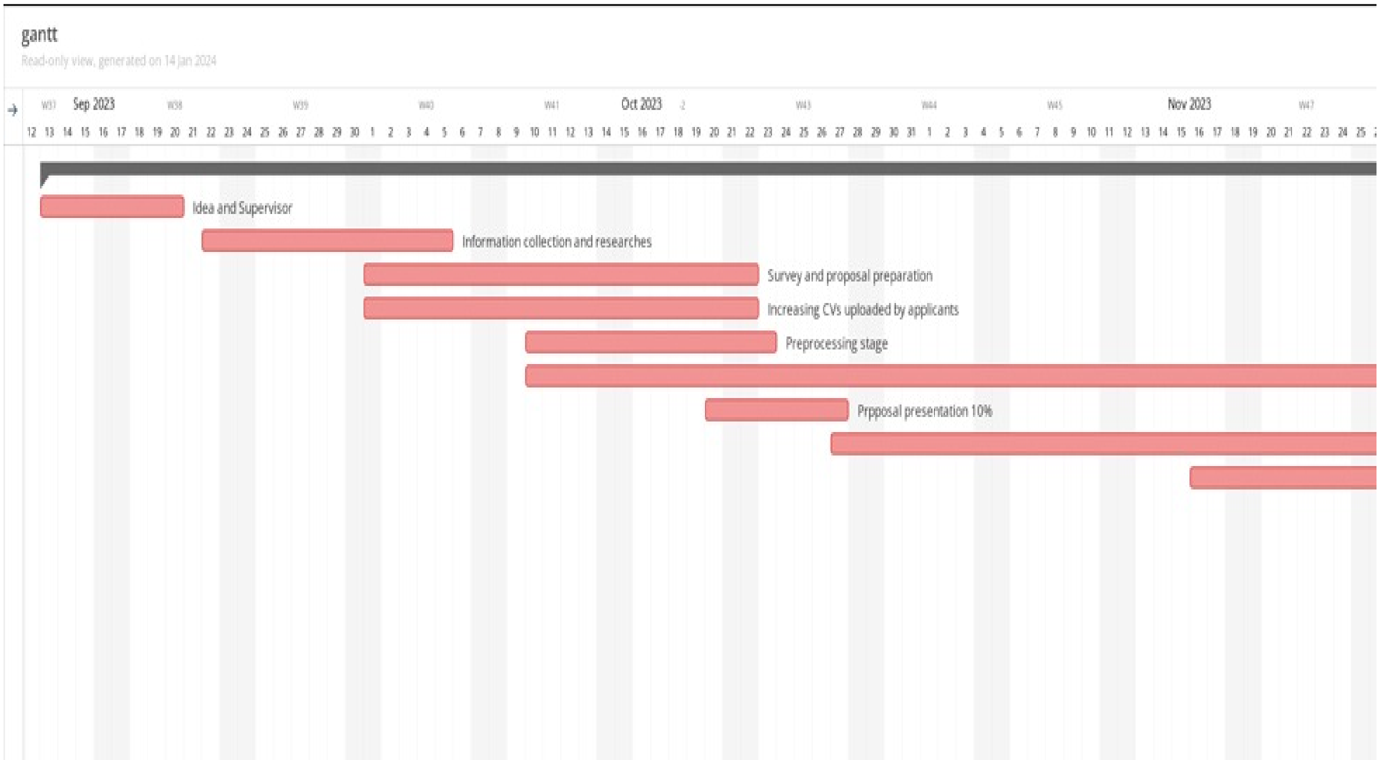


Figure 11: Gantt chart

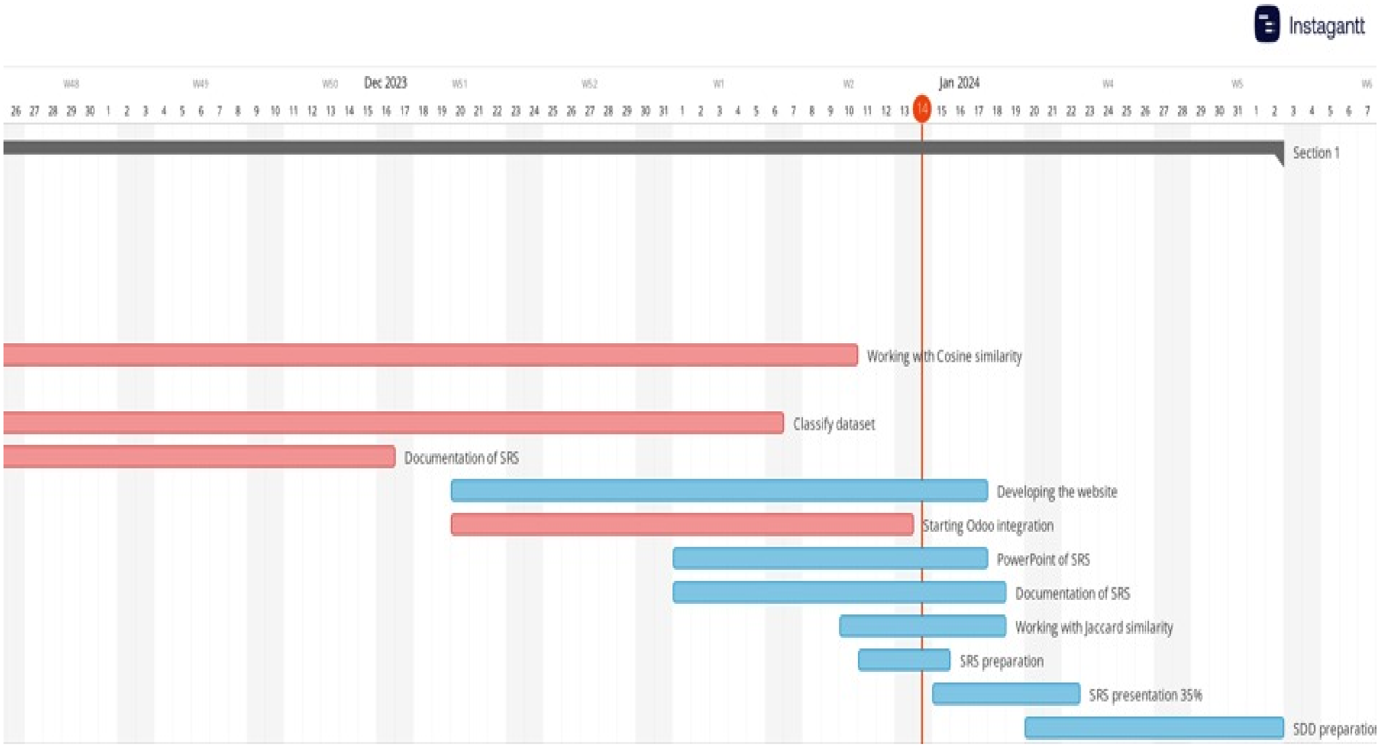


Figure 12: Gantt chart

11 Appendices

11.1 Definitions, Acronyms, Abbreviations

Abbreviation	**Definition**
SRS	Software Requirement Specification
NLP	Natural Language Processing
SpaCy	Open-source natural language processing (NLP) library, It is developed in Python.
Cosine Similarity	Is a measure used to assess the similarity between resumes or job descriptions.
Jaccard Similarity	Is often applied to evaluate the similarity between job descriptions and resumes.
Chat GPT	Language model developed by Open-AI based on the GPT (Generative Pre-trained Transformer) architecture.
SDD	Software Design Document.
Odoo ERP	(Enterprise Resource Planning)is a suite of open-source business applications designed to streamline and automate various business processes.

11.2 Supportive Documents

- Survey:

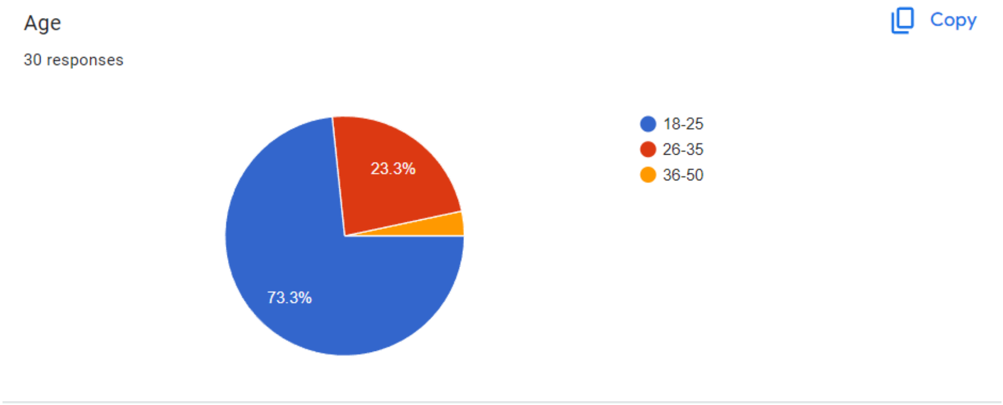


Figure 13: Question 1

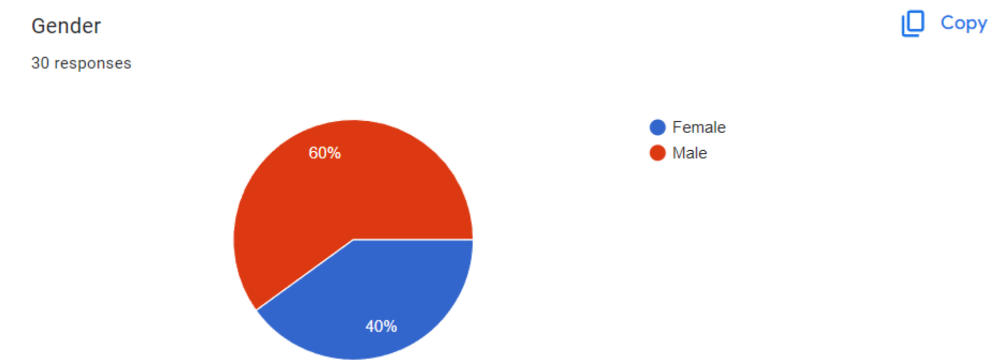
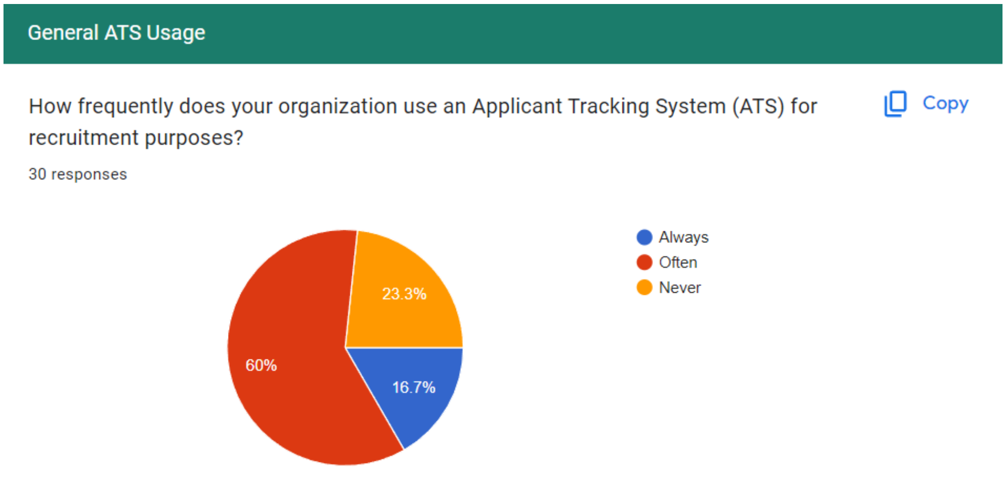


Figure 14: Question 2

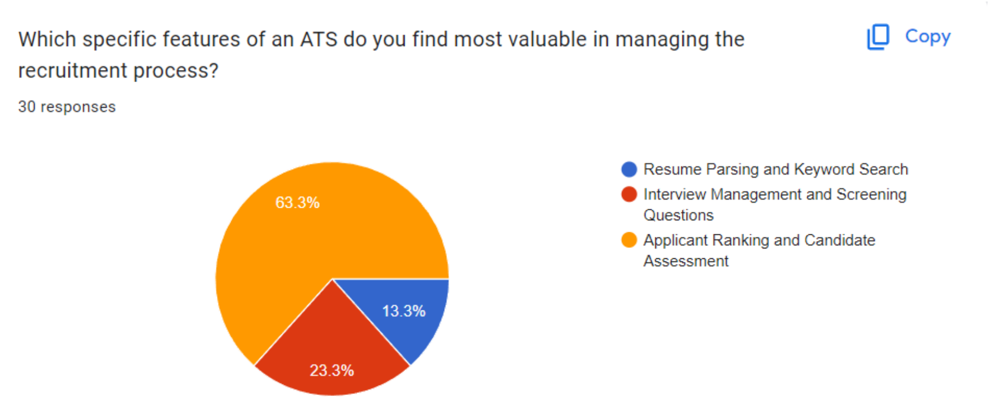


General ATS Usage

How frequently does your organization use an Applicant Tracking System (ATS) for recruitment purposes?

30 responses

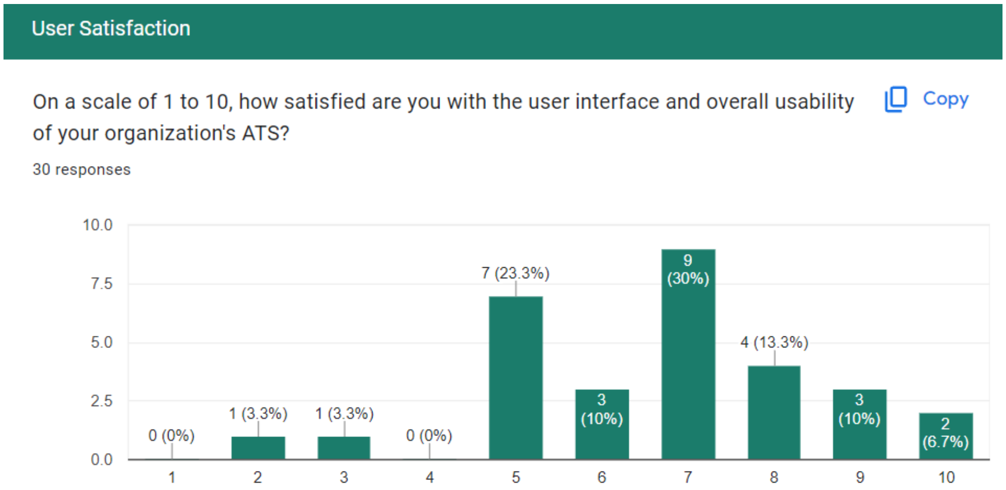
Figure 15: Question 3



Which specific features of an ATS do you find most valuable in managing the recruitment process?

30 responses

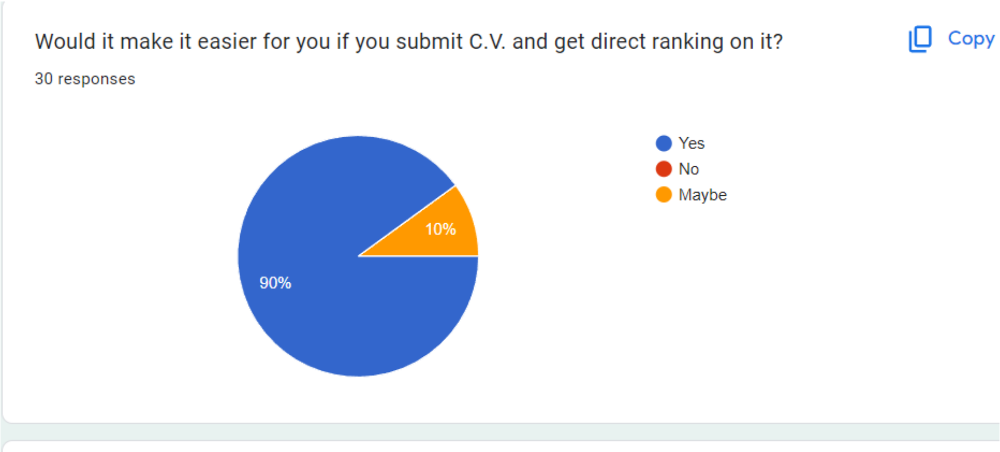
Figure 16: Question 4


User Satisfaction

On a scale of 1 to 10, how satisfied are you with the user interface and overall usability of your organization's ATS?

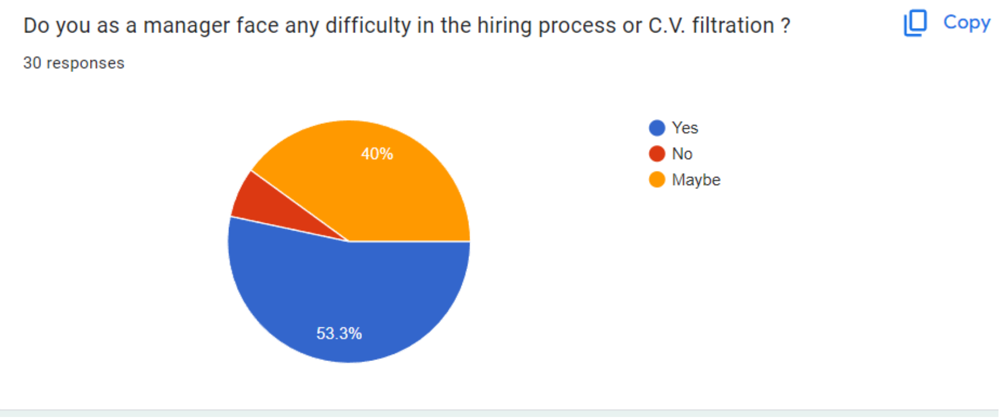
30 responses

Figure 17: Question 5


Would it make it easier for you if you submit C.V. and get direct ranking on it?

30 responses

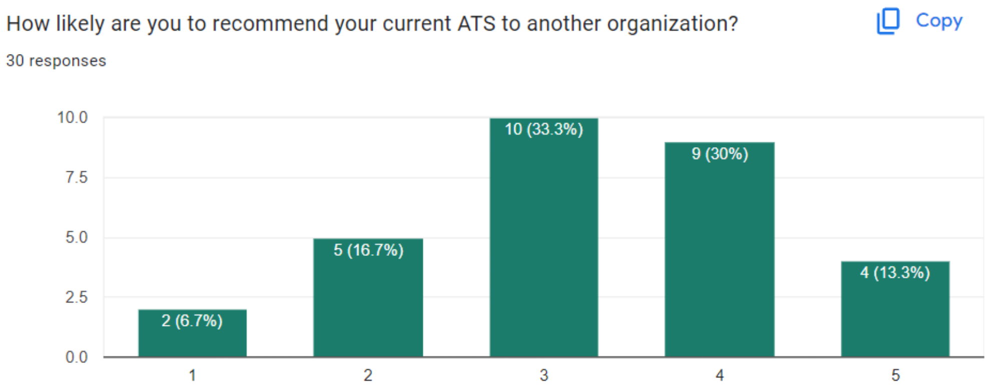
Figure 18: Question 6



Do you as a manager face any difficulty in the hiring process or C.V. filtration ?

30 responses

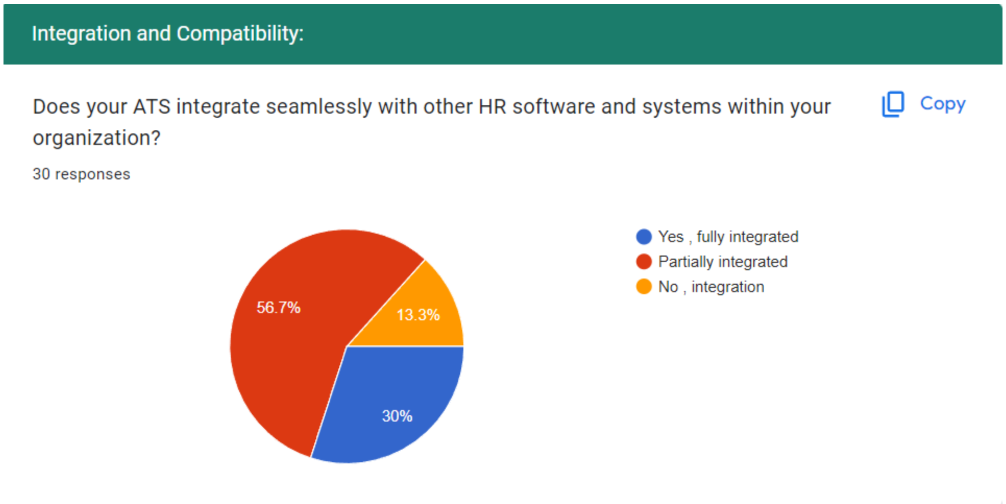
Figure 19: Question 7



How likely are you to recommend your current ATS to another organization?

30 responses

Figure 20: Question 8

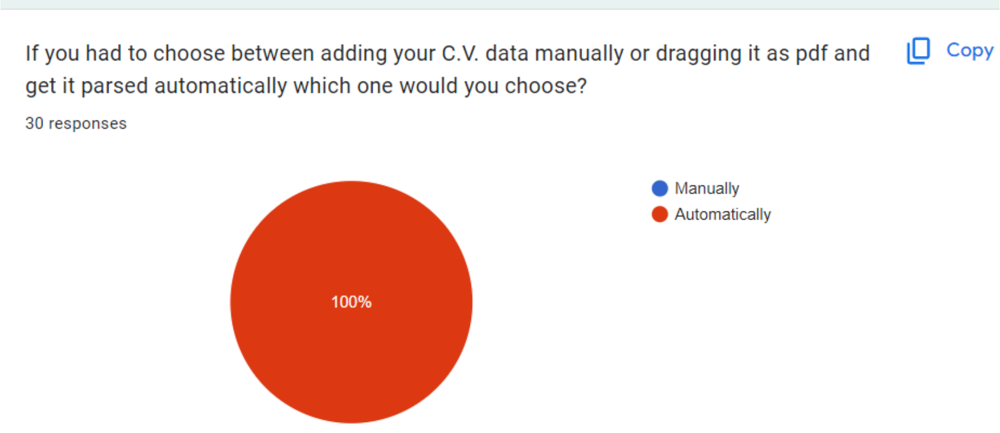


Integration and Compatibility:

Does your ATS integrate seamlessly with other HR software and systems within your organization?

30 responses

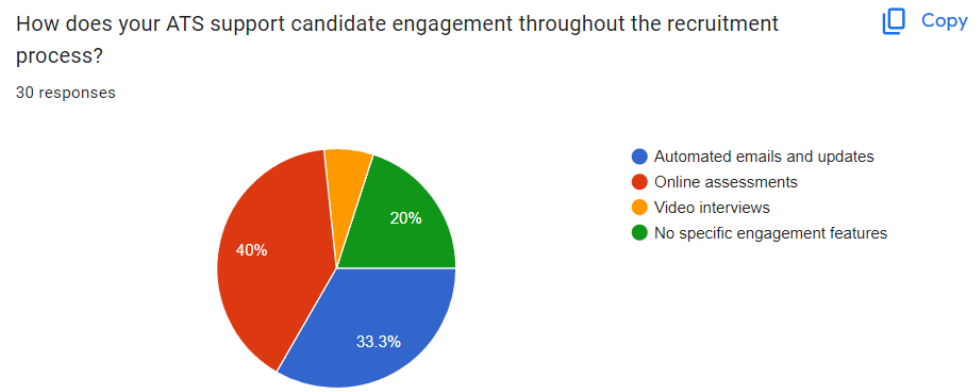
Figure 21: Question 9



If you had to choose between adding your C.V. data manually or dragging it as pdf and get it parsed automatically which one would you choose?

30 responses

Figure 22: Question 10

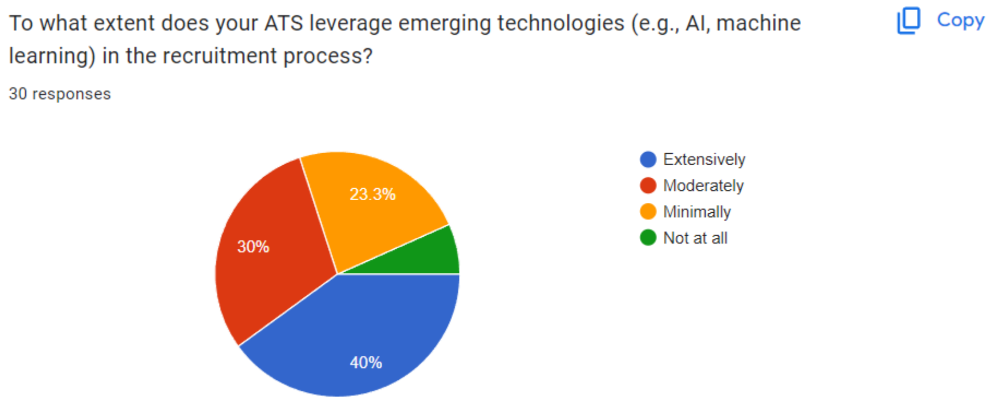


How does your ATS support candidate engagement throughout the recruitment process?

30 responses

Copy
● Automated emails and updates
● Online assessments
● Video interviews
● No specific engagement features
20%
40%
33.3%

Figure 23: Question 11



To what extent does your ATS leverage emerging technologies (e.g., AI, machine learning) in the recruitment process?

30 responses

Figure 24: Question 12

References

[1] Miroslava Peicheva. “Data analysis from the applicant tracking system”. In: *Choveshki Resur. Tehnol. HR Technol. Creat. Space Assoc* 2 (2022), pp. 6–15.

[2] Irfan Ali et al. “Resume classification system using natural language processing and machine learning techniques”. In: *Mehran University Research Journal Of Engineering & Technology* 41.1 (2022), pp. 65–79.

[3] Pradeep Kumar Roy, Sarabjeet Singh Chowdhary, and Rocky Bhatia. “A Machine Learning approach for automation of Resume Recommendation system”. In: *Procedia Computer Science* 167 (2020), pp. 2318–2327.

[4] Chirag Daryani et al. “An automated resume screening system using natural language processing and similarity”. In: *ETHICS AND INFORMATION TECHNOLOGY [Internet]. VOLKSON PRESS* (2020), pp. 99–103.

[5] Zahid Hussain et al. “New similarity measure of Pythagorean fuzzy sets based on the Jaccard index with its application to clustering”. In: *Ain Shams Engineering Journal* 15.1 (2024), p. 102294.

[6] A Ly, B Uthayasooriyar, and T Wang. *A survey on natural language processing (nlp) and applications in insurance. ArXiv*. 2010.

[7] Kadambari Wailthare et al. “A COSINE SIMILARITY-BASED RESUME SCREENING SYSTEM FOR JOB RECRUITMENT”. In: ().

[8] Reid Hoffman. “LinkedIn”. In: (2003). https://business.linkedin.com/talent-solutions/recruiterworks (Last visited on 14/11/2023).

[9] CEO James Hu. “JobScan”. In: (2013). https://www.jobscan.co/ (Last visited on 14/11/2023).

[10] “elevatus”. In: (). https://www.elevatus.io/advanced-hiring-platform/?utm$_{s}ource = capterra$ (Last visited on 14/11/2023).