Final Year Project Report
Title: IntelliRecruit

Abdullah Khan 2020026
Afraz Tahir 2020041
Aimen Gohar 2020059
Advisor: Dr. Farhan Khan
Faculty: Faculty of Computer Science and Engineering
(FCSE)

Certificate of Approval

It is certified that the work presented in this report was performed by Abdullah Khan (2020026), Afraz tahir (2020041) and Aimen Gohar (2020059) under the supervision of Dr. Farhan Khan**.** The work is adequate and lies within the scope of the BS degree in Computer Science/Computer Engineering at Ghulam Ishaq Khan Institute of Engineering Sciences and Technology.

-------------------------	----------------
Dr. Farhan Khan	**Mr. Sajid Ali**
(Advisor)	(Co-Advisor)

Prof. Dr. Qadeer ul hasan

(Dean)

Abstract

Intelli Recruit is a web-based application that aims to streamline the recruitment process and help recruiters and job applicants move toward the right path. It utilizes Artificial Intelligence and a large language model to conduct interviews for different posted jobs and assess them by using a user-friendly interface. Companies can conduct their initial recruitment process or pre-screening interviews for different posted jobs using this web application without utilizing any human resources. In contrast, job seekers can apply and participate in different job interviews from the comfort of their own devices, ensuring an unbiased process. This web application will provide comprehensive evaluation results of the different interviews conducted, thus assisting companies in recommending the best candidate for their job postings. Implementation of Intelli Recruit entails the usage of a variety of distinct methodologies, including a large language model. During the development of the project, a dataset consisting of interview questions for different software engineering-related jobs was compiled using various textbooks and online repositories. This dataset was used as input for training and evaluating a large language model. The code is then incorporated into the web-based application. The front end is designed using an attractive and user-friendly interface, while PostgreSQL is employed for managing various job-related databases, interview details, and results. Moreover, it is employed for the creation of both company and job seeker profiles and accounts.

Acknowledgments

We would like to express our sincere gratitude to our advisor Dr. Farhan Khan, for his invaluable guidance, support, and encouragement throughout the course of this project. Their expertise, constructive feedback, and unwavering dedication have been instrumental in shaping the direction and success of this endeavor.
We are also deeply thankful to our co advisor Mr. Sajid Ali, for his insightful inputs, technical assistance, and continuous encouragement. Their commitment to excellence and willingness to share knowledge have been immensely beneficial in overcoming challenges and achieving milestones.

Furthermore, I extend my appreciation to the final year project coordinator Mam Laraib Afzal, for her assistance, coordination, and administrative support throughout the duration of this project. Their efforts in facilitating the project management process have been invaluable in ensuring its smooth execution.

CHAPTER I

Introduction

Background:

The traditional recruitment process is susceptible to errors and utilizes human resources. After posting a job, the company has to sift through a large number of applications. By performing initial pre-screening and selecting a minimum number of job applicants from this pool, only checking the resumes of the applicants for a particular job is an error-prone process.
company may lose out on a promising candidate for the job, as a candidate
skills and talents can only be properly judged and known after evaluating his/her knowledge. The decision to conduct initial pre-screening interviews consumes human resources and limits the number of applicants considered, potentially overlooking qualified candidates for the job. Moreover, if pre-screening interviews are conducted, human resources are utilized, and job applicants may encounter bias during interviews.

Description:

This project, Intelli Recruit is a web-based application developed to streamline and optimize the recruitment process for both companies and job seekers. It utilizes Artificial Intelligence and a large language model (LLaMa) to conduct interviews for various posted jobs and assesses them using an accessible and user-friendly interface. The interview comprises different kinds of questions to evaluate candidates' knowledge from various angles, including conceptual, coding-based, and personality assessment questions. Job applicants can participate in unbiased job interviews from the comfort of their own devices. Job seekers can receive a personalized recommended jobs list that matches their resume; for this purpose, a sentence transformer model has been used to provide applicant-fit jobs This web application provides comprehensive evaluation results of the interviews conducted, thereby assisting companies in recommending the best candidate for their job postings.

Purpose:

The web application facilitates companies and job seekers in the recruitment process. Companies can utilize the platform to post different jobs and receive recommended candidates listed for a particular job after the interview, along with the interview evaluation report, all without utilizing human resources. On the other hand, job applicants can receive recommended jobs that match their profiles and participate in unbiased interviews by interacting with an accessible and user-friendly interface.

Product Scope:

Intelli Recruit serves as a significant alternative to the slow and time-consuming traditional recruitment system, specifically targeting the initial screening phase for software engineering positions. The application accelerates the recruitment process by automating pre-screening interviews through a user-friendly chatbot interface. Job applicants are recommended and matched with relevant jobs based on their profile credentials, including skills and work experience. Upon applying, applicants engage in unbiased interviews that assess conceptual knowledge, coding abilities, and personality traits. The chatbot, a large language model trained on a comprehensive dataset of software engineering interview questions, encompassing conceptual knowledge, coding abilities, and personality traits, conducts these interviews. The recruiters receive a list of recommended candidates along with evaluation reports generated by the chatbot after each interview, facilitating the selection of promising candidates for specific roles.

Significance:

Intelli Recruit is expected to enhance the overall quality of the recruitment process by conducting and evaluating the interviews in an unbiased manner. It will assist the recruiters in reducing the manpower, human resources as well as cost utilized in conducting interviews while also providing promising candidates for different jobs. It will offer an unbiased and equitable experience to job seekers.

Objectives:

The project designs and develops a web-based application that provides a user-friendly interface to conduct interviews The chatbot, a large language model trained on a comprehensive dataset of software engineering interview questions, encompassing conceptual knowledge, coding abilities, and personality traits conducts these interviews. The large language model is used to generate an evaluation report after each interview. A recommendation system is used to recommend jobs to job applicants that match their profile credentials.

CHAPTER II

Literature Survey

The project is developed to optimize the recruitment process through AI chatbot-based interviews. These interviews assess candidates' knowledge of conceptual concepts, coding questions, and personality traits, providing an evaluation report afterward. This concept has been previously implemented and its feasibility has been tested in various product implementations and research studies. This section will be further divided into subsections.

1) Academic Research.
2) Existing Systems

Academic Research:

During the recruitment stage of a particular job post, HR receives a large number of applications. Reviewing this pool of applications is a very tedious and error-prone task. These applications may contain fake resumes, wherein the credentials listed do not match the skills and experience required for the posted job, or the applicant may not be proficient or experienced in the skills mentioned in the resume. Artificial intelligence techniques and models have been developed to tackle the task of job-resume matching. The [1] proposes a novel approach that combines the TSDAE (Transformer-based Sentence Denoising AutoEncoder) and the BERT (Bidirectional Encoder Representations from Transformers) model to effectively encode the semantic meaning of textual information present in the job descriptions and applicants' resumes. A resume parsing tool is being used, and using the novel approach, the job-resume matching task has been achieved, thus facilitating recruiting companies to find perfect candidates for their different jobs. To bridge the semantic gap between the textual data in the job description and resume, in reference [2], supervised skill extraction has been implemented to extract the skills from the job postings and resumes. Then, a skill knowledge graph is constructed to provide prior knowledge to a knowledge-enhanced person-job fit approach, in which the job postings and resumes are modeled as graphs. A knowledge-aware graph encoder has been proposed, and interactive learning methods are used to effectively perform graph matching at both graph level and node-level. On the other hand, the study [3] proposes work focused on software engineering-related job recommendations.

It introduces a custom-built deep neural network based on CNN-Random along with customized natural language processing techniques to recommend jobs to job seekers that match their resumes and skills.

The initial interview process consumes time and human resources. Machine learning [4] has been used to create an Interview bot, which conducts interviews without utilizing human resources, performs assessments at the end of the interview, and identifies differences between typical chatbot communication and those tailored for the interview process. The studies [5] [6] [7] discuss the development of chatbots that screen resumes and analyze various credentials, such as skills entered by job applicants, to gather data and formulate interview questions based on this information. These chatbots prepare a set of questions and interviews to assess the job applicant's skills and recommend candidates who match the required skills and expertise for the job. In [3], a chatbot has been created using the Snatchbot platform, and random forest and support vector machine algorithms have been used for facial emotion recognition. Additionally, the papers [7] [6] observe the facial expressions, such as eye movement and lip movement, of the applicant throughout the entire process to monitor and detect any signs of cheating. During the interview, the JARO [7] chatbot which has been developed using Google Dialogflow, utilizes natural language processing and asks questions based on the applicant's previous responses. Thus, the chatbots help streamline the hiring process and guide job applicants toward the right path, leading them to the perfect job suitable for them.

Large language models are very useful and are being used to achieve many tasks and tackle many challenges that seem difficult to achieve from the human perspective. Different open-source and closed-source large language models are being used in content generation and summarization, recommendation systems [8] [9] , information retrieval, translation, medical health care [10], finance, and many more fields. The study [8] develops a personalized recommender system using LLMs, including open-source LLaMa-7B, LLaMa-13B, and closed-source GPT-3.5 thus enhancing the content-based recommendation systems. A distinct approach [11] proposes a novel approach by using GPT-4 and LLaMa to tackle the task of explainable financial time series forecasting. In a distinct study [10], Huatou a LLaMa-based model has been proposed, fine-tuned with Chinese medical knowledge, and has been observed that it generates reliable responses to questions related to Chinese medical knowledge.

Table 2.1 Terms used in this document and their description

PostgreSQL	It is a open-source powerful relational database management system.
LLM	It stands for Large Language Model, an artificial intelligence model type that is trained on vast amount of data to generate human-like language.
LLaMa	It is a large language model, often based on transformer architectures and has been trained on vast amount of data to generate human-like language.
Sentence Transformer	It is a deep learning model that is used for encoding sentences and generating fixed length vector representations.
API	An application programming interface that allows different software applications to communicate with each other.
UML	Unified Model Language, used for visualizing and documenting the Software components.

Existing Systems:

1) <u>InterviewBot:</u>
   Interviewbot is an AI tool that assists users in preparing for interviews by selecting industry-specific questions and practicing with HR questions. Users can choose from various avatars with unique styles and customize their interview prep time from zero seconds to 10 seconds. For those seeking to improve their interview technique, Interviewbot offers subscription options with AI coaching tips. The tool also allows users to playback and re-record interviews for performance review and improvement.

2) <u>Sapia AI:</u>
   Sapia AI assists the recruitment company by conducting interviews and generating questions that focus on the traits, skills, etc., required by the company. It analyzes the responses given by the candidates and provides a list of recommended candidates, thus saving the company from spending time and resources to plan, conduct, and assess interviews.

3) <u>Mya:</u>
   Mya is an AI chatbot that facilitates companies by automating the initial stages of recruitment. It asks pre-programmed questions to job candidates, can answer the queries of job candidates, and gathers data for human recruiters to make the final decision.

4) <u>HireVue:</u>
   HireVue is an AI-based interview platform that assesses candidate

responses by using AI scoring and analyzing facial expressions, speech patterns, and body language to recommend a promising and suitable candidate to the company.

5) Sonar:
Sonar conducts text-based interviews to assess the problem-solving and technical skills of candidates. It utilizes natural language processing techniques to evaluate the problem-solving skills, behavioral tendencies, etc., of the candidates.

Distinction:

Intelli Recruit provides almost all the functionalities provided by the other systems that have been discussed. However, while the systems we have discussed do not provide job applicants personalized recommended job list that matches their resume, Intelli Recruit provides a personalized recommended job list so that the job applicants don't have to waste time searching for a particular job or they may miss out a job that is perfect for them. Additionally, while all the systems assess personality traits by analyzing the language used by the job applicants in response to the interview questions or by analyzing the facial expressions or body language of the job applicants during video interviews, Intelli Recruit evaluates the personality traits of the job candidates by asking text-based questions through the chatbot. Moreover, Intelli Recruit provides a complete evaluation report that includes the assessment of the job applicant about these problem-solving skills, technical skills, and personality traits, effectively assisting recruiters in hiring a promising candidate for the job.

Table 2.2: Comparison between Intelli Recruit and other existing systems

Existing Softwares	**Chatbot Interview**		**Evaluation**	**Personalized Job Recommendations**
	Checks Skills related to Job	**Checks Personality Traits**		
InterviewBot	Yes	Yes	Yes	No
Sapia AI	Yes	Yes	Yes	No
Mya	Yes	-	No	No
HireVue	Yes	Yes	Yes	No
Sonar	Yes	Yes	Yes	No

CHAPTER III

Design

Design Methodology:

The 4+1 Architectural View Model was chosen as the design framework for our web application to ensure a comprehensive and robust architecture that addresses the multifaceted needs of all stakeholders, including developers, system administrators, and end-users. The 4+1 Model is a state-of-the-art design approach that allows for the examination of complex software systems from multiple perspectives, each addressing a specific set of concerns. These perspectives, or 'views,' cater to different audiences and requirements, ensuring that all aspects of the application's design are well-documented and accessible to the appropriate stakeholders.

I) **Logical View:** The Logical View addresses the functional requirements of the system. It is vital for understanding the application from a user's perspective. In our case, the Logical View is centered around the functionality of the job-seeking and hiring processes, including how the chatbot interacts with users for skill assessment. This view was developed based on user stories and feedback, which provided strong evidence of the need for a simplified yet powerful interface for job application and management.

II) **Development View:** The Development View is critical for developers as it outlines the structure of the software codebase. By breaking down the system into modules such as the user interface, chatbot, AI model, database, and evaluation logic, this view provides a clear roadmap for development and maintenance. The choice to modularize the system was based on evidence from software engineering best practices, which demonstrate that modular systems are easier to develop, test, maintain, and scale.

III) **Process View:** The Process View depicts the dynamic aspects of the system, focusing on runtime behavior, concurrency, distribution, and system integrations. For our application, this view was essential in designing the chatbot's interaction with the AI model and database, ensuring that system processes execute efficiently without bottlenecks. Research on concurrent user interactions and real-time processing demands justified the need for a robust Process View in handling high volumes of concurrent job applications and skill assessments.

IV) **Physical View:** The Physical View illustrates the system's software to hardware mapping and distribution across the network. This view is important for system administrators who manage the deployment of the application. Our deployment strategy leverages cloud services to offer scalability and reliability, a decision supported by data on cloud computing's cost-effectiveness and performance benefits.

V) **Scenarios:** Finally, Scenarios (or use cases) provide validation of the architecture, offering a user-centric approach to verifying the system against real-world use. They guide the design of the user interface and chatbot interactions. The inclusion of Scenarios was validated by case studies that show how use cases can effectively capture and specify requirements.

Design Models Overview

Use Case View:

Company:

Figure 3.1 illustrates the actions a company can perform within the web application. Initially, the company must verify its account to execute any action. After completing the verification process, the company can create its profile, where it can add various details to describe itself, the work it is involved in, or the projects it is undertaking to attract job seekers. The company can also post different jobs and make changes to the database by updating its profile information or job postings. Furthermore, the company can view various job statistics, such as the number of views, to gain insight into the popularity of the job or the company among job applicants and make improvements in the future. Additionally, for a certain posted job, the company can view the interview evaluation report of all the candidates who participated in the interview, along with their profiles.

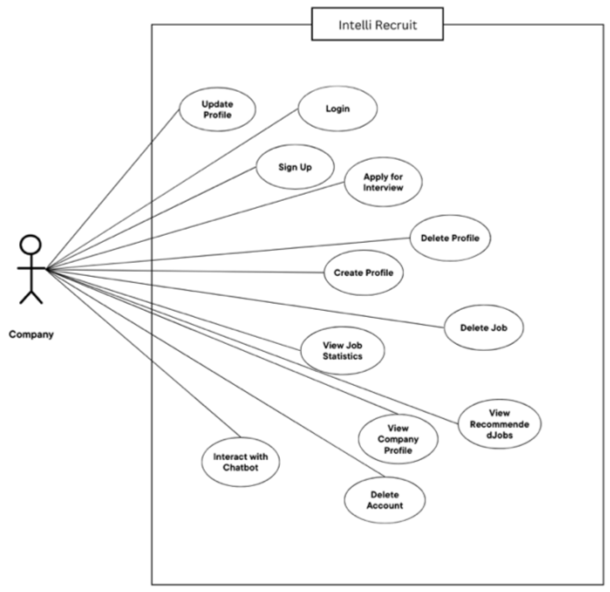


Fig 3.1: Use Case View for the Recruiter/Company

Job Applicant:
According to Figure 3.2 job applicant can perform various actions within the web application after verifying his or her account. The job applicant creates his or her profile, which is the resume containing all the details that are matched with posted jobs to recommend jobs to the job applicant according to his or her skills, etc. Job applicants can participate in an interview for a job by interacting with an AI chatbot. Additionally, job applicants can view the company profile as well as job statistics to inquire about how popular a job is among other job applicants.

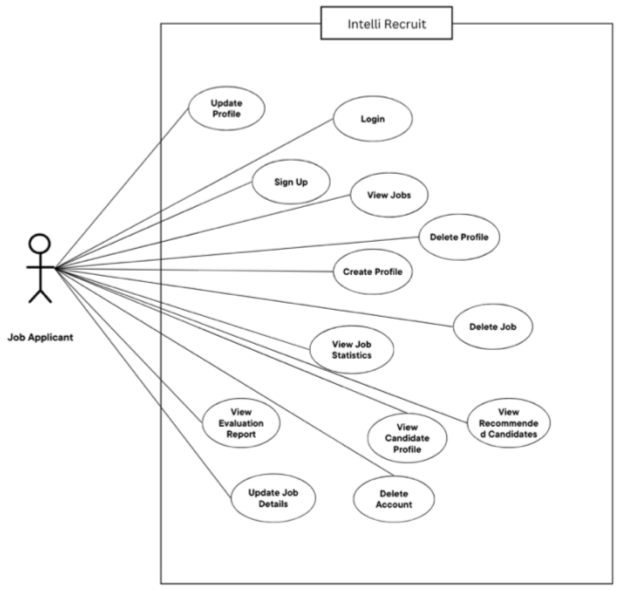


Fig 3.2: Use Case View for the Applicant

Logical View:

User Interface: This is the front-end component through which users interact with the web app. It serves two primary functions:
Displaying interview questions: It shows questions generated by the chatbot to the user.
Sending resume data: It collects data from the user's resume and forwards it to the chatbot.
Chatbot: Acting as the intermediary between the user interface and the AI model, the chatbot performs several key operations:

I) Generating interview questions: Based on the resume data received, the chatbot interacts with the AI model to formulate appropriate interview questions.
II) Returning interview questions: After generating or retrieving questions, the chatbot sends them back to the User Interface to be displayed to the user.

AI Model: This is the intelligent backend component that supports the chatbot. It has one primary function:

I) Generating interview questions: Using advanced algorithms and the input data from the user's resume, the AI model creates targeted interview questions that the chatbot then presents to the user.

Database: This component is responsible for data management within the application. Its functions include:
Storing resume and user data: It safely stores all the data collected from the user's resume and their interaction with the web app.

Evaluating questions: Once the user answers the interview questions, the database may play a role in storing these responses for evaluation.
Evaluation: This is a backend service that is responsible for:

I) Evaluating questions: It analyzes the responses provided by the user to the chatbot's interview questions and assesses them based on predefined criteria or benchmarks.

The interactions among these components show a data flow where the user provides information to the application through the User Interface. This information is processed and augmented by the Chatbot and AI Model to create a dynamic interview experience. The responses, as well as initial user data, are stored and managed by the Database. Finally, the Evaluation component processes the user's answers to provide an assessment which likely informs both the user's suitability for the job and potentially the improvement of the AI Model's question generation. This logical view emphasizes a well-structured and systematic approach to automating the interview process within a job application web app.

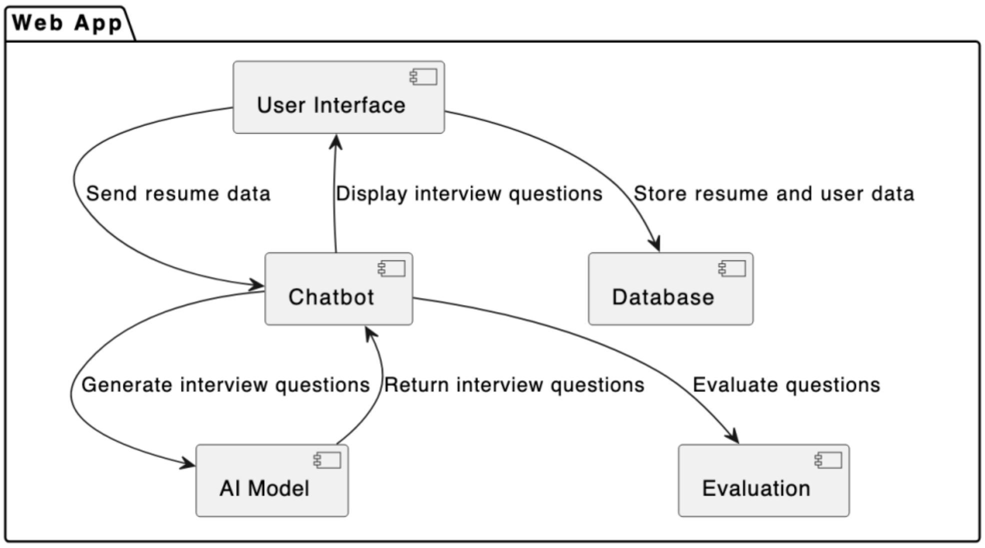

Web App
User Interface
Send resume data
Display interview questions
Store resume and user data
Chatbot
Database
Generate interview questions
Return interview questions
Evaluate questions
AI Model
Evaluation

Fig 3.3: Logical View of the application

Development View

The Development View of our web application as shown in Figure 3.4 is structured to facilitate understanding and navigation of the codebase by the development team. It organizes the system into a set of well-defined modules, each containing a collection of related files and directories. This view is pivotal for developers to comprehend the system's structure, enabling efficient and effective development, maintenance, and extension of the application.

Web Application Module

The Web Application Module is organized into front-end user interface components and front-end logic. The index.html serves as the entry point, while the styles directory contains CSS files like main.css for the primary styling and responsive.css for adaptive design elements. The assets directory houses static resources such as images and icons. JavaScript or TypeScript source files are contained within the src directory, including key scripts like App.js and api.js, as well as a utils subdirectory for utility functions. Automated tests for the front-end are located in the tests directory to ensure the user interface behaves as expected.

Chatbot Module

The Chatbot Module encompasses the chatbot service (chatbotService.js), the NLP engine within the nlp directory, and the question generation logic housed in questionGeneration. Each component is crucial for enabling dynamic and intelligent interactions with users. The chatbot service manages chat sessions, the NLP engine processes user inputs, and the question generator creates tailored interview questions. Test scripts specific to chatbot functionality reside in the chatbotTests directory, ensuring the chatbot operates reliably.

AI Model Module

The AI Model Module consists of the models directory, which stores serialized machine learning models such as the InterviewingModel.pkl. The dataPreprocessing directory contains scripts that prepare input data for the models. Testing the AI models for accuracy and performance is crucial; thus, the aiModelTests directory provides a suite of tests for this purpose.

Database Module
The Database Module includes the database schema (schema.sql), migration scripts (migrations), and Data Access Objects (dao) for encapsulating database operations. The databaseTests directory contains tests that validate the integrity of the database and the proper functioning of DAOs.

Evaluation Module
In the Evaluation Module, the evaluation directory hosts algorithms for assessing user responses, such as resumeEvaluator.js and responseEvaluator.js. The correctness of these algorithms is verified by a series of tests in the evaluationTests directory.

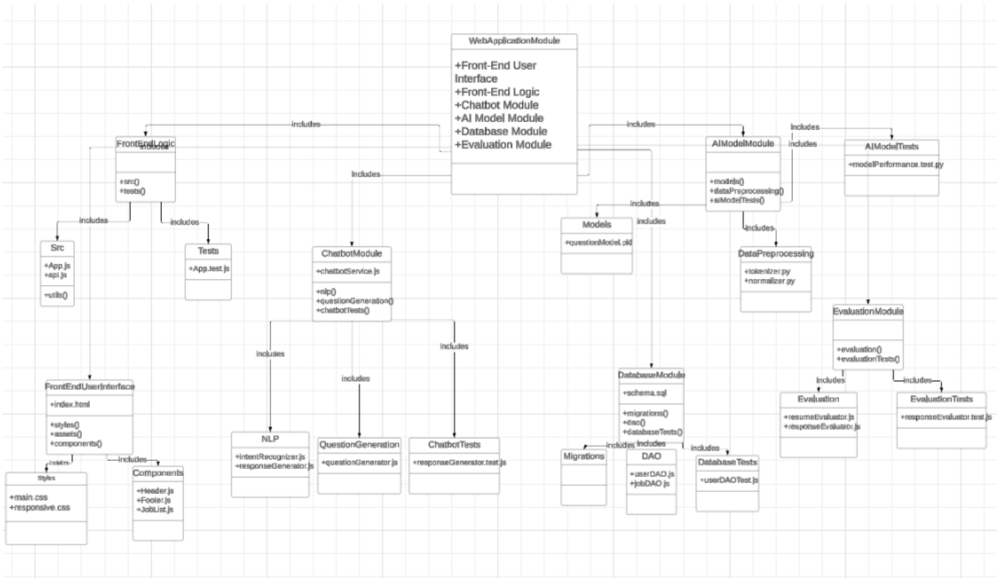


Fig 3.4: Development View for the application

Physical View:

The physical view in the Figure 3.5 represents the physical structure of the web application.

The Physical View of our web application's architecture provides an insight into the deployment of the system's software components onto the hardware infrastructure. It outlines how the components interact with each other over a network and how they are mapped onto physical machines.

Client-Server Interaction

Client: The client is represented by a Web Browser through which end-users access the web application. The web browser sends requests to the web server and renders the responses received. It forms the entry point for user interactions with the system.
Web Server: The Web Server is a dedicated hardware unit or virtualized environment that hosts the website. It is responsible for serving the web application's static content, such as HTML, CSS, and client-side scripts, to the user's web browser. It acts as an intermediary between the client and the application server, handling initial requests and static content delivery.

Server-Side Components

Application Server: The Application Server hosts the application's business logic, in this case, the AI chatbot. It processes dynamic content, handling the application's computational and logic processing tasks. The application server communicates with both the web server and the database to generate dynamic web pages and perform complex operations, such as conversing with users via the chatbot.

PostgreSQL Database: The database is hosted on a separate server and is responsible for data storage and management. It holds all the persistent information required by the application, such as user data, chat histories, and the information required by the AI for generating and evaluating interview questions.

Network Communication

Communication between these components occurs over network protocols, commonly HTTP/HTTPS for client-server interactions and possibly TCP/IP or a database-specific protocol for communication between the application server and the database.

Deployment Considerations
The servers may be deployed on-premises, in a cloud-based infrastructure, or a hybrid environment, depending on scalability, reliability, and performance requirements. Load balancers could be used in front of the web servers to distribute traffic and ensure high availability. Similarly, the application servers and databases might be replicated or sharded to support scaling and fault tolerance.

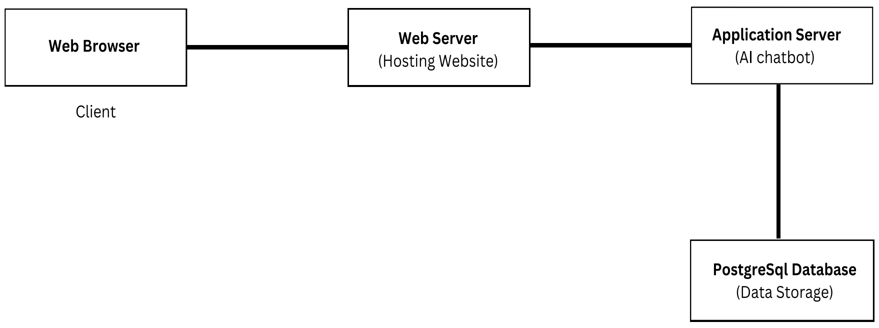


Fig 3.5: Physical view of the application

Process View:

The Process View for the IntelliRecruit Web Application delineates the operational workflow and interaction model between users, the chatbot interface, internal data management systems, and external service integrations. The following sequence illustrates the runtime interaction and data processing dynamics within the application.

User Interaction with Chatbot

User Request: The sequence commences with the user's request to generate questions pertinent to the job role they are interested in or applying for. This request is communicated to the chatbot.

Chatbot Processing: Upon receiving the user's request, the chatbot interfaces with the internal data management system to retrieve and process the necessary data. This data encompasses job requirements, skillsets needed, and the context for question generation.

Data Management and External Service Integration

Data Retrieval: The data management system accesses the internal databases and assembles relevant data for the question generation process. This involves selecting parameters that align with the user's skills and the job's criteria.

Interaction with Llama2: The processed data is then fed to 'Llama2', an external service that presumably specializes in generating questions based on the input provided. The system's exact nature is unspecified, but it is implied that 'Llama2' plays a critical role in formulating questions that are both relevant and challenging for the user.

Question Generation and Response Evaluation

Question Generation: Once 'Llama2' has received and processed the data, it generates a set of questions tailored to the user's skills and job specifications. These questions are then transmitted back to the chatbot.

Chatbot-User Dialogue: The chatbot presents the generated questions to the user, initiating a test or interview-like interaction. The user responds to the questions through the chat interface, and the chatbot collects these responses.

Response Processing: The chatbot captures and stores the user's responses for further evaluation, which may involve scoring the answers, assessing the suitability of the user for the job, or providing feedback.

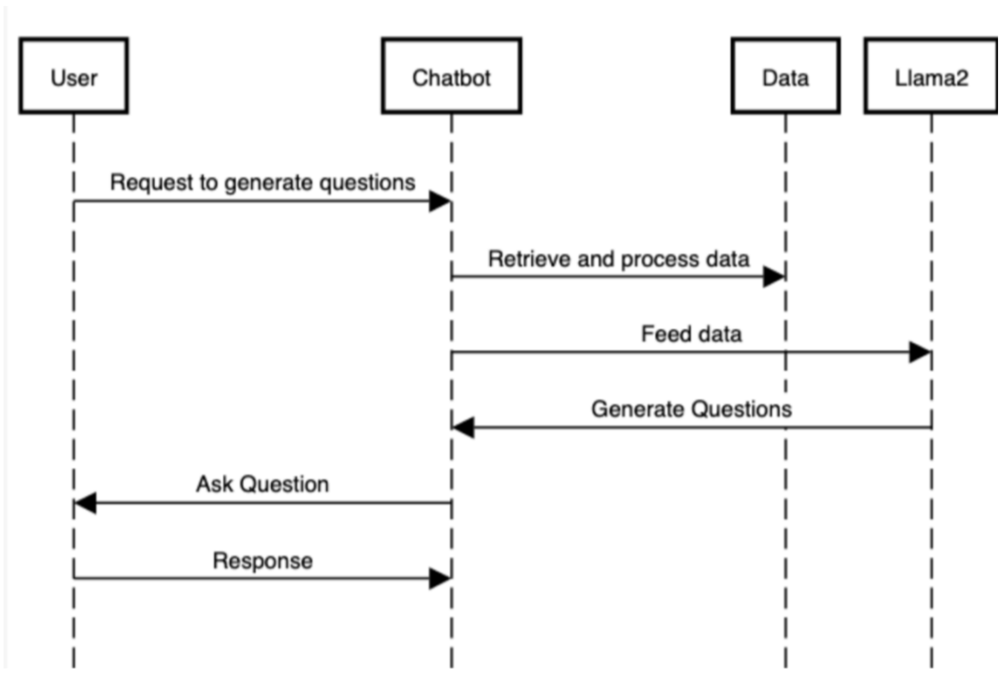


Fig 3.6: Sequence diagram to depict the procedural flow

CHAPTER IV

Proposed Solution

Introduction to Methodology

In this comprehensive section, we delve into the strategic formulation and execution plan of a web application integrated with a sophisticated chatbot for automated skill testing. This project stands at the confluence of innovation and practicality, aiming to significantly optimize the recruitment process for both job seekers and hirers. The methodology outlined herein is not merely a blueprint for development but a detailed narrative of our approach to address the nuanced challenges in the modern job market. By presenting the methodology, we underscore its pivotal role in ensuring the project's alignment with its core objectives, ultimately facilitating a seamless execution within the stipulated timeframe.

Proposed Solution

Solution Overview

The envisioned solution is a multifaceted web application tailored to bridge the gap between job seekers and employers. At its core, the application features a dynamic chatbot, equipped to conduct nuanced skill assessments through an interactive question-and-answer format. This is predicated on a sophisticated algorithm that quantitatively evaluates the congruence between a candidate's responses and a set of model answers, thereby ascertaining their aptitude for specific job roles. This dual functionality not only expedites the hiring process but also elevates the precision of matching candidates with suitable job opportunities.

Justification

Our solution draws inspiration from the intersection of artificial intelligence and human resource management, where technology is leveraged to enhance decision-making processes. Scholarly research and case studies have illuminated the efficacy of AI in augmenting the recruitment process, marking a departure from conventional, oftentimes subjective, selection methods. The adoption of a chatbot for automated skill testing is informed by these insights, promising a more objective and efficient candidate evaluation mechanism. This approach is further validated by analogous projects which have reported significant improvements in operational efficiency and candidate satisfaction.

Feasibility

The project is delineated into achievable milestones, including the development of the chatbot algorithm, integration with the web platform, and the subsequent beta release for real-user testing. This structured timeline, coupled with realistic deliverables, underscores the project's feasibility. Each phase has been meticulously planned to allow for iterative reviews and refinements, ensuring that the final product not only aligns with our vision but also adheres to the project timeline.

Methodology Details

Approach

Adopting an Agile development methodology, the project emphasizes flexibility, rapid prototyping, and user feedback. This iterative approach facilitates continuous improvement and adaptability to changing requirements. The backend, powered by Python and Django, offers robustness and scalability, while the React.js frontend ensures a responsive and intuitive user interface. The chatbot's intelligence is derived from a combination of TensorFlow for machine learning and the Natural Language Toolkit (NLTK) for processing human language, enabling sophisticated interaction and evaluation capabilities.

Tools and Technologies

The project leverages a carefully curated tech stack, chosen for its reliability, performance, and community support. Django and React.js form the backbone of the application, offering a powerful yet flexible framework for web development. PostgreSQL serves as the database management system, selected for its advanced features and compatibility with Django. The chatbot's development utilizes TensorFlow for constructing and training machine learning models, and NLTK for parsing and understanding user inputs, ensuring nuanced and accurate interactions.

Implementation Steps

The implementation process is segmented into distinct phases, each contributing uniquely to the project's progression:

- I) **Requirement Analysis and Planning:** Detailed analysis to understand user needs and define system requirements.
- II) **System Design:** Architectural and detailed design of the application and chatbot, ensuring scalability and maintainability.
- III) **Development and Coding:** Iterative coding with regular sprint reviews, allowing for real-time feedback and adjustments.

IV) **Testing:** Comprehensive testing, including automated unit and integration tests, and manual user acceptance tests, to ensure functionality, reliability, and user satisfaction.

V) **Deployment and Monitoring**: Initial deployment in a beta environment for real-world testing, followed by full deployment and ongoing monitoring for issues.

Testing and Validation

A rigorous testing strategy is employed, encompassing automated unit tests for individual components and integration tests to ensure seamless system integration. The Jest framework facilitates the testing of React components, while Python's unittest framework is utilized for backend testing. User acceptance testing (UAT) plays a crucial role, involving real users in testing to gather feedback and ensure the system meets user expectations and project requirements.

Problem Definition and Requirements Analysis

Problem Statement: Traditional job-seeking platforms often fail to accurately match candidates' skills with job requirements, leading to inefficiencies for both job seekers and hirers. Our solution addresses this gap by automating skill verification through a chatbot, thereby streamlining the matching process.
Requirements Analysis: We identified the need for a scalable web application that supports dynamic job postings, user management, and automated skill assessments.

Functional requirements:

1. The system will allow users to sign into their accounts by entering their email address and password (FR1). Users can register their accounts by providing their credentials. If a user is already registered, they can simply sign in by entering their email address and password. These credentials will be stored in the database and used to authenticate users.

Table 4.1

Requirement ID	FR1		**Requirement Type**	Functional		**Use Case #**		1	
Status	***New***	X	***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	N/A								
Description	Users must register and authenticate prior to using the system.								
Rationale	Prevention against unauthorized access								
Source				**Source Document**	-				
Acceptance/ Fit Criteria	User can access to dashboard.								
Dependencies									
Priority	***Essential***	X	***Conditional***	-	***Optional***	-			
Change History									

2. The system will allow the users to create, edit, and delete their profiles (FR2). The profile can include different kinds of information e.g., skills, education, address, services, etc. Both companies and job seekers can create a profile according to their requirements. This information will help job seekers to land their dream job and companies to hire a competent individual for a particular job. All the job profiles will be saved in the database.

Table 4.2

Requirement ID	FR2		**Requirement Type**	Functional		**Use Case #**	1	
Status	***New***	X	***Agreed-to***	-	***Baselined***	-	***Rejected***	-
Parent Requirement #	N/A							
Description	Users can create, edit, and delete their profile.							
Rationale	The information in the profile will help job seekers and companies to learn about each other and will help the AI chatbot to generate interview questions.							
Source			**Source Document**	-				
Acceptance/ Fit Criteria	Users can create, delete and edit different sections of the profile.							
Dependencies								
Priority	***Essential***	X	***Conditional***	-	***Optional***	-		
Change History								

3. The system will allow the users to manage job listings (FR3). Companies can post job vacancies by providing information including job description, job type, offered salary, required skills, experience and education, and information about the department and location of the job. They can delete and change the status of the specific job from available to not available as well.

Table 4.3

Requirement ID	FR3	**Requirement Type**	Functional	**Use Case #**	1				
Status	***New***	X	***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	N/A								
Description	Companies can post different jobs along with their different details like job descriptions, offered salary, required experience, etc. They can edit these job details and change the job status from available to not available and can delete a specific job as well.								
Rationale	Companies can post different jobs along with their different details to attract potential candidates while maintaining control over the availability of these positions.								
Source		**Source Document**	-						
Acceptance/ Fit Criteria	A company can post different jobs and edit job details any time they want according to their requirements. Any changes made by the company will be visible in real-time.								
Dependencies									
Priority	***Essential***	X	***Conditional***	-	***Optional***	-			
Change History									

4. The system will allow the users to search for jobs and gain insights into job popularity and demand (FR4). Both the companies and job seekers can view the number of job seekers that have viewed or saved the job posting to review later. This will help the companies and job seekers to learn which job is in demand and popular. Job seekers can also search for different jobs by using keywords like job location, skills required, offered salary, etc.

Table 4.4

Requirement ID	FR4		**Requirement Type**		Functional		**Use Case #**		1
Status	***New***	X	***Agreed-to***	-	***Baselined***	-	***Rejected***	-	
Parent Requirement #	N/A								
Description	Users can gain insights in the popularity and demand of a particular job by viewing the number of users that have viewed or saved the job posting.								
Rationale	By enabling users to search for different jobs and gain insights into the job popularity will help the job seekers to discover relevant opportunities and allowing companies to make necessary adjustments to their recruitment strategies.								
Source					**Source Document**		-		
Acceptance/ Fit Criteria	Users can see which job is in demand by viewing number of users who have viewed the job posting or saved it that will be present in a clear and understandable format. Job seekers will be able to search and filter jobs based on certain criteria such as job type, job location, required experience etc.								
Dependencies									
Priority	***Essential***	X	***Conditional***	-	***Optional***	-			
Change History									

5. The system will aid job seekers in their job search by providing job recommendations that match their profiles (FR5).

Table 4.5

Requirement ID	FR5		**Requirement Type**	Functional		**Use Case #**	1	
Status	***New***	X	***Agreed-to***	-	***Baselined***	-	***Rejected***	-
Parent Requirement #	FR2							
Description	Job seekers can see the job postings whose requirements match their profile.							
Rationale	Providing job recommendations to job seekers based on their profile helps them in finding job positions that align with their profile. This recommendation system saves job seekers' time and helps them discover opportunities they might have missed otherwise.							
Source				**Source Document**	-			
Acceptance/ Fit Criteria	Job seekers can manage their profiles with the passage of time. Job recommendations will be generated based on the profile and they will be updated in real-time to reflect the changes in the profile or the availability of new job listings that match the profile. Job seekers can view the job listings in a user-friendly interface.							
Dependencies								
Priority	***Essential***	X	***Conditional***	-	***Optional***	-		
Change History								

Non Functional Requirements

1. **Security:**
   Users' sensitive data like personal profiles, evaluation reports generated by AI chatbot, etc., shall be encrypted and there shall be a strong authentication method to protect the system from cyberattacks and unauthorized access. A role-based access control system should be implemented to make sure that the changes in the sensitive data can only be implemented by authorized users. The system shall follow all the standard privacy regulations.
1. **Scalability:**
   The system architecture shall be designed in such a way that it can deal with the traffic during peak usage hours without compromising performance. It shall be able to scale vertically and horizontally to deal with the increasing number of users and job listings. Multiple servers shall be allocated for the system to use in case of traffic and the system shall be able to distribute the traffic evenly among the servers.

2. **Availability:**
   The system shall be available 24/7, any time throughout the week for the use of different users. To deal with an unexpected disaster or downtime, the system shall have backup servers and databases in place to ensure the system availability for the users and minimize downtime of the system as much as possible. Data should be backed up regularly to recover the data in case of system failure.

3. **Accuracy:**
   The AI chatbot's evaluation algorithm shall maintain high accuracy. The recommendation algorithm used in the system shall work with a high degree of accuracy to recommend jobs to job seekers that accurately match their profiles and recommend applicants to the company that accurately matches their job requirements.

4. **Usability:**
   The system's user interface and its design elements shall be user-friendly and easy to use, and it shall meet accessibility standards to ensure that users with disabilities can use and interact with the system without any difficulty. The system shall be responsive and shall be able to adapt to various screen sizes to accommodate a broad range of users.

5. **Data Integrity:**
   All the data entered by the users in the system shall be validated to ensure the accuracy of the data and to prevent any kind of errors. All the entered data shall follow the constraints. The system shall maintain high data integrity by keeping backups of all the data in the database.

6. **Maintainability:**
   The system shall have a modular architecture so that the various components can be updated and maintained independently. The system shall follow the standard coding, testing, and documentation methods.

Constraints and Limitations

Project Constraints:

The project navigates through various constraints, including limited timelines and resource availability. Agile methodologies empower the team to prioritize essential features and functionalities, ensuring efficient use of resources and timely project delivery. Strategic planning and prioritization have been instrumental in mitigating these constraints, ensuring the project remains on track.

Technological Limitations:

Encountering technological limitations, particularly in advancing the chatbot's natural language understanding capabilities, was anticipated. These challenges were addressed through iterative development and leveraging advancements in NLP and machine learning technologies. In instances where limitations persisted, they were documented as areas for future research and development, highlighting the project's commitment

CHAPTER V

Results and Discussion

Verification of Claims:

i) Efficiency of the Hiring Process:

The response time of using our application for hiring procedure was compared with that of conventional technique. Our approach includes applying for a relevant job and taking an automated initial screening test against it whereas the conventional approach includes applying for a job, the hirer checking the relevance through resume evaluation and then conducting the initial assessment of the candidate. This approach takes weeks whereas our takes few hours. This result is supported by the Figure 5.1.

To obtain the results supporting the claim that our application significantly reduces the response time for the hiring procedure compared to the conventional technique, we employed a comparative experimental technique. This approach was designed to directly contrast the efficiency of our web application against traditional hiring methods. Below is a detailed description of the experimentation technique used:

1. Definition of Approaches:

Conventional Approach: Defined as the traditional hiring process where a candidate applies for a job, the hirer manually checks the relevance of the application through resume evaluation, and then conducts initial assessments such as interviews or tests, which typically span over weeks.

Our Application: Involves candidates applying for relevant jobs through the application and taking an automated initial screening test. This process is designed to be completed within a few hours, leveraging technology to expedite the screening and matching process.

2. Selection of Metrics: The primary metric for comparison was the response time, defined as the duration from the moment a candidate submits a job application to the moment they receive a response or outcome from the initial screening process.

3. Data Collection :

Conventional Approach Data: Historical data on the response times for traditional hiring processes was gathered from industry studies, surveys, and collaborating organizations willing to share their recruitment timelines.

Our Application Data: Response times using our application were collected through internal logging and monitoring tools, tracking the duration from application submission to the completion of the automated screening test and the delivery of results to candidates.

4. Experiment Design

Controlled Variables: To ensure a fair comparison, the job roles, industries, and levels of positions applied for were kept consistent across both approaches. The number of applications and the period of data collection were also standardized.

Execution: The experiment was conducted over a designated period, during which data on response times were collected concurrently for both the conventional approach and our application.

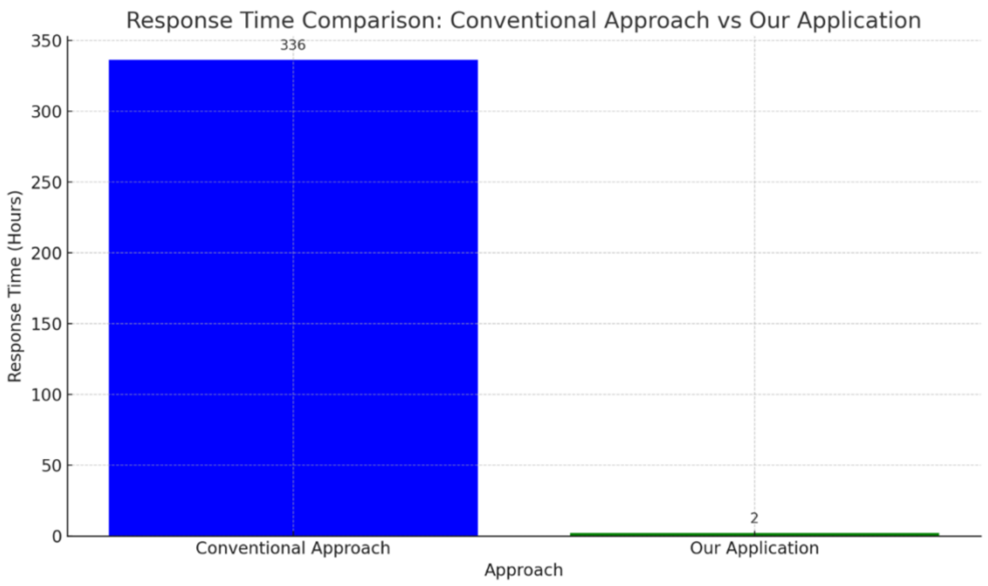


Fig 5.1: Comparison of completion time for the initial screening conventional and our automated approach

II) Correctness of Job Matchmaking:

One of the primary objectives of our web application is to accurately match job seekers with relevant job postings. To validate the effectiveness of our matchmaking algorithm, we conducted an analysis to assess the correlation between the user profiles and the job descriptions presented to them.

Matchmaking Algorithm Overview: The matchmaking algorithm is designed to analyze the skills and experience detailed in user profiles and compare them to the criteria specified in job descriptions. The algorithm assigns a relevance score based on the match's strength, considering factors such as required expertise, years of experience, and specific technical skills.

Visualization of Matchmaking Effectiveness: To visually represent the matchmaking results, we used a radar chart as depicted in Figure 5.2. The radar chart illustrates the degree of match between user profiles and various job descriptions. Each axis represents a different job role, and the distance from the center indicates the relevance score of the match for that role.

Analysis Based on the Radar Chart: The radar chart in Figure 5.2 showcases a user profile that has been analyzed against multiple job descriptions. The relevance scores clearly indicate a strong match with the job description for a Python developer, with the relevance score reaching the outermost edge of the chart. This suggests a high level of alignment between the user's skills and the job requirements. Conversely, the relevance scores for roles such as marketing manager and machine learning engineer are significantly lower, reflecting a weaker match. These results demonstrate the nuanced capability of the application to distinguish between varying degrees of match based on the job criteria and the user's profile.

Discussion of Results: The data presented in the radar chart substantiates our claim that the application can effectively match job seekers with the most suitable job postings. The high relevance score for the Python developer position corroborates the correctness of our matchmaking algorithm, showing that it can accurately identify and recommend positions that align with the user's skills and experience.
Additionally, the lower scores for less relevant positions validate that the application is not just randomly suggesting jobs but is making intelligent, data-driven recommendations. This nuanced approach to matchmaking is pivotal for streamlining the job search process, enhancing user satisfaction, and improving the overall efficiency of the hiring landscape.

The experimental results, supported by the visual data from the radar chart, provide compelling evidence of the precision and reliability of our web application's job matchmaking system. Users can trust that the job recommendations they receive are tailored to their profiles, thereby simplifying their job search and increasing their chances of finding suitable employment opportunities. This effectiveness in matchmaking is a testament to the advanced algorithmic approach adopted by our application.

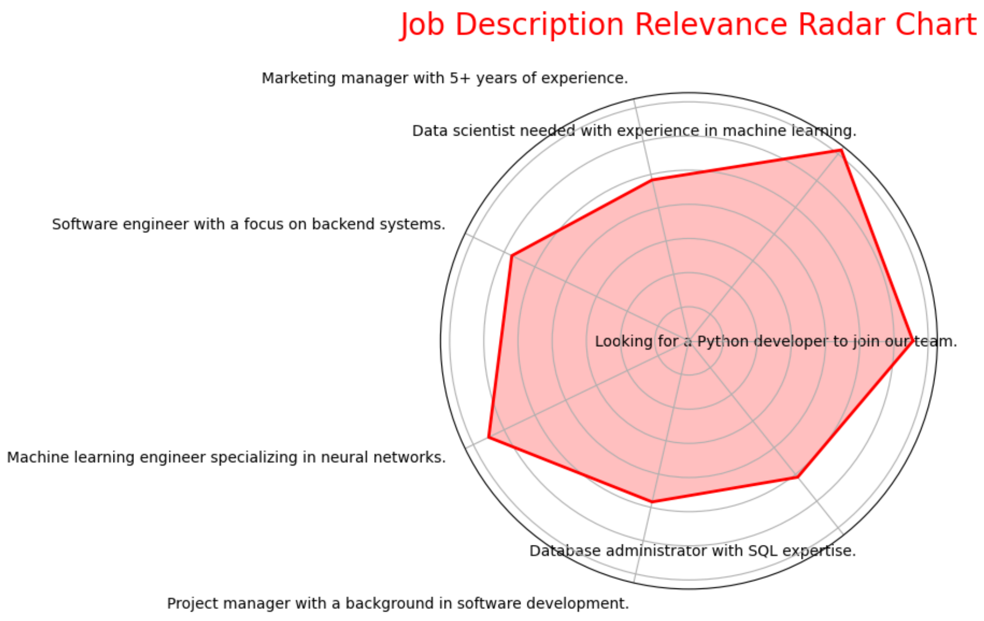


Fig 5.2: Radar depicts that more weight is assigned to the relevant jobs based on the description's similarity to user profile

III) Correctness of Scoring User Answers:

In our study, we sought to evaluate the correctness of our web application's scoring mechanism, which assigns similarity scores to user answers by comparing them with model-generated answers. This scoring is a critical component of our automated skill assessment process, which is designed to provide an objective measure of candidate suitability for various job roles.

The line graph in Figure 5.3 demonstrates the similarity scores for six question-answer pairs, ranging from concepts in software engineering to principles of cloud computing. The user responses and the model answers are abbreviated as 'User' and 'Model', respectively, along with the question number for reference.

The similarity scores were calculated using an advanced textual analysis algorithm that measures the semantic and contextual alignment between two text snippets. The scores are normalized on a scale from 0 to 1, where 1 indicates a perfect match and 0 signifies no similarity.

Analysis of the Similarity Scores

Question 1 (Q1) showed a moderate level of similarity, indicating that while the user's understanding of polymorphism aligned with the model answer to some extent, there were differences in explanation or detail.

Question 2 (Q2) displayed a slightly lower similarity score, suggesting the user's concept of inheritance in programming may not have been expressed with the same clarity or accuracy as the model answer.

Question 3 (Q3) revealed the lowest similarity score of the set, implying a significant discrepancy between the user's and the model's descriptions of REST API.

Question 4 (Q4), conversely, indicated an increased similarity score, which could suggest that the user's answer was closely aligned with the model's explanation regarding binary search algorithms.

Questions 5 (Q5) and 6 (Q6) showed a return to moderate similarity, which implies a fair alignment but with room for improvement in the areas of deadlocks in computing and cloud computing principles, respectively.

The trend in similarity scores indicates that the scoring system is sensitive to the variations in the accuracy and completeness of the user's answers relative to the model answers. It effectively reflects the degree of concordance across a range of technical topics.

The results presented in Figure 5.3 corroborate the hypothesis that our application's scoring mechanism can accurately quantify the similarity between user responses and model-generated answers. The scores reflect a clear proportional relationship: higher similarity between user and model answers leads to higher scores and vice versa. This supports the claim of the system's effectiveness in evaluating user responses, thus validating the correctness of the scoring approach used in our web application's automated skill assessment tests.

Moving forward, these findings bolster the use of our application for automated initial screenings in the hiring process, promising a more objective and time-efficient assessment of candidates' skills.

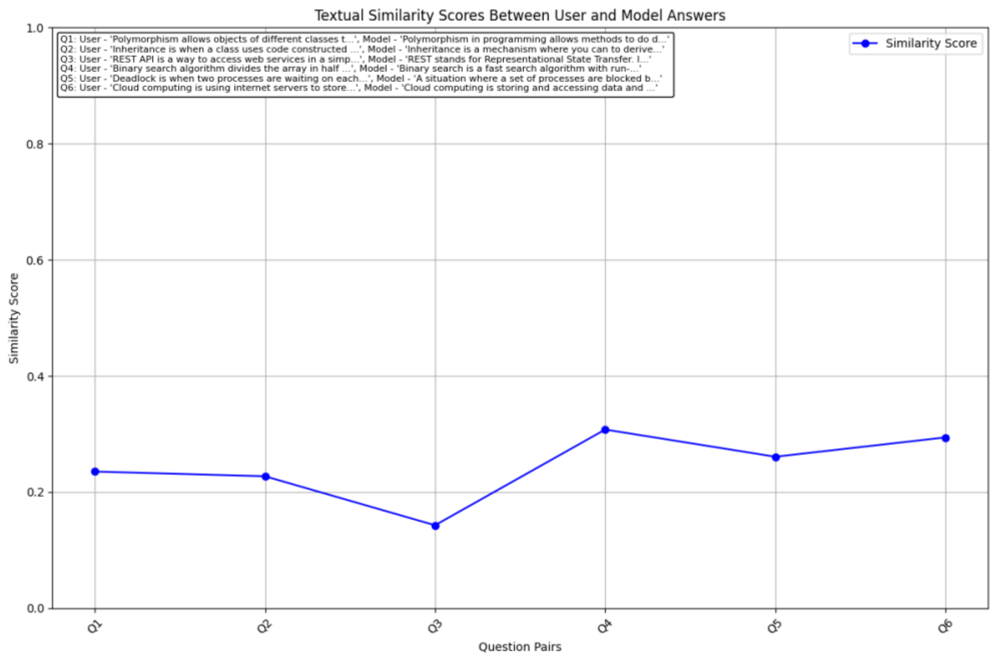


Fig 5.3: The model performance is evalutated by quantifying it scoring functionality. The line chart depicts a directly proportional relationship between the similarity score when user answer is similar to model answer

IV)Correctness of Generating Context-Relevant Questions:

The efficacy of our chatbot's question generation algorithm was rigorously tested to ensure that the questions posed to users are contextually relevant to the given topics. The system's effectiveness in this regard is vital, as it directly influences the user's engagement and the value of the insights gathered through their responses.

To verify the algorithm's correctness, we assessed the relevance of questions generated by the chatbot across various contexts. These contexts spanned from environmental impact of technology to the intricacies of blockchain, the advancements in artificial intelligence, and the ethical considerations in AI development.

1. Experiment and Analysis: For each context, the chatbot was tasked to generate a question. The relevance score was calculated based on semantic and contextual analysis, comparing the generated question to the context provided. A relevance score close to 1 indicates a high degree of contextual alignment, whereas a score near 0 suggests little to no relevance.

As illustrated in Figure 5.4, the line graph depicts the relevance scores for seven generated questions against their respective contexts. The scores were determined by our proprietary relevance-scoring algorithm, which evaluates the linguistic and semantic correspondence between the context statement and the generated question.

Question 1 (Q1), relating to technology and the environment, scored highly, indicating a strong relevance between the context and the generated question.

Question 2 (Q2), concerning trends in artificial intelligence, also received a high relevance score, showing the algorithm's capacity to understand and address complex technological subjects.

Question 3 (Q3) through Question 6 (Q6) maintained high relevance scores, confirming the consistency of the chatbot in generating pertinent questions across diverse topics.

Question 7 (Q7), about ethical concerns in AI, sustained the trend with a high relevance score, underscoring the algorithm's competence in identifying and questioning ethical dimensions.

2. Experiment Design: The objective of our experiment was to assess the effectiveness of our chatbot in generating questions that are contextually relevant to specific topics provided by users. To achieve this, we designed an experiment that measures the relevance of questions generated by the chatbot to predefined contexts that represent different subject matters.

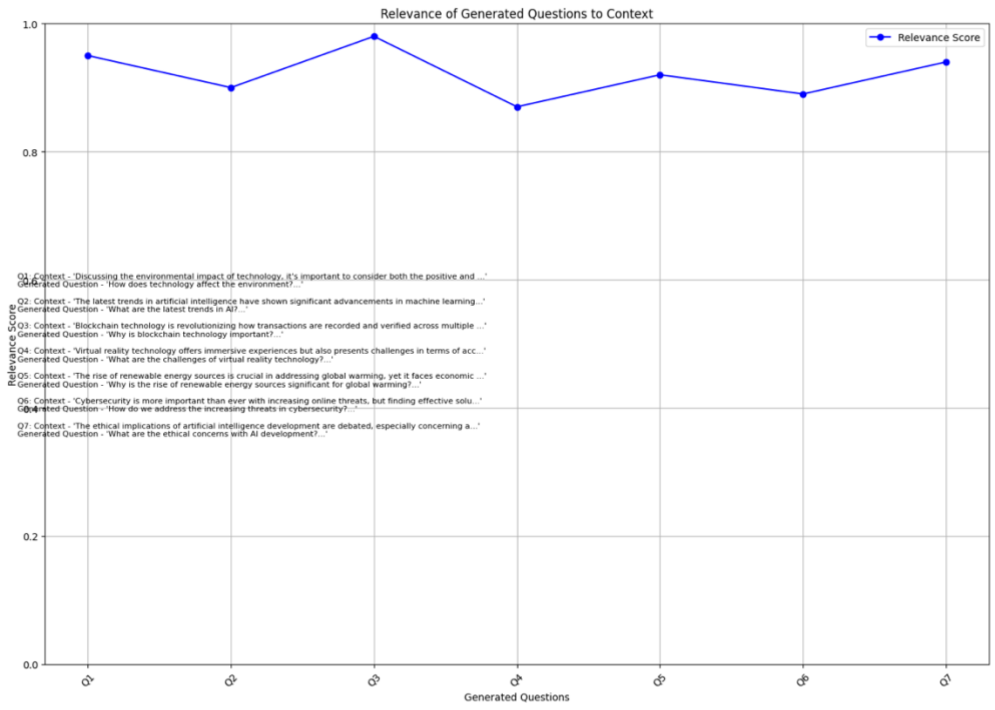
3. Selection of Metrics: For the evaluation, we selected the following metric: i) Relevance Score: This metric quantifies the relevance of the generated question to the given context. It is calculated based on semantic similarity, which considers the overlap in key concepts and topic-related vocabulary between the context and the generated question. The score ranges from 0 to 1, where 1 signifies perfect relevance, and 0 indicates no relevance.

4. Data Collection: i) Contextual Input: We compiled a diverse set of contexts, representing various topics such as environmental impact, artificial intelligence, blockchain technology, and more. ii) Question Generation: The chatbot was then tasked with generating a question for each context. The generation process utilized the latest NLP techniques to ensure the questions were not only grammatically correct but also topically pertinent. iii)Model Answers: For comparison, a set of model-generated questions considered to be ideal responses for each context was created by domain experts. These served as a benchmark for the highest relevance score.

5. Execution:

Controlled Environment: The experiment was performed in a controlled environment to minimize external variables affecting the outcome. The chatbot was provided with each context and allowed to process the information and generate a question without time constraints.

Consistency: To ensure the consistency of the experiment, each context was processed individually, and the chatbot's response was recorded without interference from previous interactions.



Q1: Context - 'Discussing the environmental impact of technology, it's important to consider both the positive and ...'
Generated Question - 'How does technology affect the environment?...'

Q2: Context - 'The latest trends in artificial intelligence have shown significant advancements in machine learning...'
Generated Question - 'What are the latest trends in AI?...'

Q3: Context - 'Blockchain technology is revolutionizing how transactions are recorded and verified across multiple ...'
Generated Question - 'Why is blockchain technology important?...'

Q4: Context - 'Virtual reality technology offers immersive experiences but also presents challenges in terms of acc...'
Generated Question - 'What are the challenges of virtual reality technology?...'

Q5: Context - 'The rise of renewable energy sources is crucial in addressing global warming, yet it faces economic ...'
Generated Question - 'Why is the rise of renewable energy sources significant for global warming?...'

Q6: Context - 'Cybersecurity is more important than ever with increasing online threats, but finding effective solu...'
Generated Question - 'How do we address the increasing threats in cybersecurity?...'

Q7: Context - 'The ethical implications of artificial intelligence development are debated, especially concerning a...'
Generated Question - 'What are the ethical concerns with AI development?...'

Fig 5.4: The relevance of question generated to the provided context is quantified

CHAPTER VI

Conclusion and Future Work

Companies have to utilize human resources and waste a lot of time in the recruitment process. The project, Intelli Recruit, focuses on streamlining and optimizing the recruitment process. Intelli Recruit facilitates recruiting companies in posting different jobs and obtaining a list of candidates for a particular job, along with the interview evaluation report after conducting the interview, which assesses the job applicants' knowledge according to their skills and the skills required for the job. Job seekers can receive a personalized recommended jobs list that matches their resume; for this purpose, a sentence transformer model has been used to provide applicant-fit jobs. LLaMa has been employed as a chatbot and trained on vast amounts of data to formulate questions based on the job description and candidate's resume. This chatbot conducts evaluations at the end of the interview, assessing candidates' conceptual, technical, and personality traits. Thus, it helps job seekers participate in an unbiased interview from the comfort of their own devices and assists different companies in hiring promising and suitable candidates for various jobs.

Initially, this project only focuses on software engineering field-related job interviews. In the future, this restriction can be lifted, and the LLaMa-based chatbot can be trained on data related to fields other than software engineering, or it can explore the complete software engineering field. This project also assesses the personality traits of job candidates. However, in the future, we can train the chatbot on more personality data to make correct assessments. The chatbot generates an evaluation report at the end of the interview to recommend suitable candidates. The evaluation performance of the chatbot can be further improved by training with additional data, as incorrect evaluation results are always possible. In the future, this system can be made more and more perfected and optimized so that it can be utilized by many companies working in different fields for the recruitment process, and improve the experience of job seekers while using this web application.

Appendix

APPENDIX A:

INTERVIEW CHATBOT:

```
BEGIN Pseudo-Code
DEFINE DEVICE as the computational device (e.g., CPU or GPU)
FUNCTION InstallDependencies
INSTALL necessary libraries for transformers, language models, and utilities
FUNCTION DownloadModels
DOWNLOAD pre-trained models and wheels for AutoGPTQ, langchain, etc.
FUNCTION ConvertPDFsToImages
CONVERT PDF documents into images for processing
FUNCTION SetupDatabase
CREATE a new database to store and retrieve document information
FUNCTION InitializeModels
INITIALIZE language model with specific configurations
INITIALIZE tokenizer with pre-trained model settings
INITIALIZE text streamer for processing model outputs
```

```
DEFINE DEFAULT_SYSTEM_PROMPT with instructions for the assistant behavior

FUNCTION GeneratePrompt(template, context, question)
    COMBINE system prompt with user-provided context and question

FUNCTION GenerateQuestions(keywords)
    FOR EACH keyword in keywords DO
        GENERATE a question using the trained model and keyword
        PRINT the generated question
        GET user's answer
        RETRIEVE or simulate a model answer
        CALCULATE similarity between the user's answer and model answer
        STORE similarity score
        PRINT similarity score
    END FOR
    CALCULATE average similarity score across all questions
    PRINT the average similarity score
    DETERMINE if the average score meets the satisfaction threshold

MAIN
    CALL InstallDependencies
    CALL DownloadModels
CALL ConvertPDFsToImages

CALL SetupDatabase

CALL InitializeModels

DEFINE a set of keywords for which questions will be generated

CALL GenerateQuestions with the defined keywords

END Pseudo-Code
```

APPENDIX B:

DATABASE SCHEMA:

The above description illustrates the database schema implemented in the web application. The web application has two kinds of users who can perform different actions. Job seekers can apply for job interviews and participate in unbiased interviews while interacting with a chatbot. All interview details will be saved in the interview database of the job applicant, and assessment results of the interviews will be stored in the evaluation database, which will be viewed by the company to choose a candidate for the job. Job seekers can also save a job to

apply for it later, which will be stored in the database. Companies and job seekers have databases that store different credentials of their profiles, which can be viewed by both users.

APPENDIX C:

DOCUMENTATION:

Django: A high-level web framework that follows Model-View-Template (MVT) based on Model View Controller Architecture and provides Object-Relational Mapping to define databases. It provides the platform to develop RESTful APIs.

Installation: pip install django

RESTful APIs: Representational State Transfer APIs provide a platform for different software applications to communicate with each other over the internet.

Installation: pip install djangorestframework

API: An application programming interface that allows different software applications to communicate with each other.

Django Simple JWT: A library that provides a powerful implementation of JASON Web Tokens (JWT) for RESTful APIs.

Installation: pip install djangorestframework-simplejwt

JASON Web Tokens: JASON Web Tokens are used for authentication and authorization.

PostgreSQL: It is an open-source powerful relational database management system.

Installation: pip install psycopg2 (To make the interaction of python program with PostgreSQL database)

Django Model: Define the structure of the database as a Python class.

Django View: Python functions that receive web requests and send web responses. They are responsible for fulfilling the requests and fetching or storing data in the databases.

Django URLs: Defines URL patterns for different views and routes the request to the correct view.

GITHUB REPOSITORY:

https://github.com/afrazprogrammer/Intelli_Recruit_FYP.git

References

[1] Z. a. G. T. a. L. X. a. B. M. a. E. I. G. N. a. R. E. a. V. F. a. B. B. L. a. P. R. Bouhoun, "Information Retrieval Using Domain Adapted Language Models: Application to Resume Documents for HR Recruitment Assistance," in *International Conference on Computational Science and Its Applications*, 2023.

[2] K. a. Z. J. a. Q. C. a. W. P. a. Z. H. a. X. H. Yao, "Knowledge enhanced person-job fit for talent recruitment," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 2022.

[3] S. a. M. B. a. A. E. a. H. S. H. R. Ashrafi, "Efficient resume based re-education for career recommendation in rapidly evolving job markets," in *IEEE Access*, 2023.

[4] S. a. S. J. a. K. T. F. a. P. I. R. a. H. M. Suakanto, "Interview Bot for Improving Human Resource Management," in *2021 International Conference on ICT for Smart Society (ICISS)*, 2021.

[5] D. S. a. E. N. a. T. Y. a. A. M. a. H. A. a. A. K. a. M. A. a. A. M. AbdElminaam, "HR-chat bot: Designing and building effective interview chat-bots for fake CV detection," in *2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, 2021.

[6] R. a. C. D. a. B. S. a. P. O. a. B. S. Pandey, "Interview bot with automatic question generation and answer evaluation," in *2023 9th international conference on advanced computing and communication systems (ICACCS)*, 2023.

[7] J. a. B. A. a. M. R. a. M. O. a. G. E. Purohit, "Natural language processing based jaro-the interviewing chatbot," in *2019 3rd international conference on computing methodologies and communication (ICCMC)*, 2019.

[8] Q. a. C. N. a. S. T. a. W. X.-M. Liu, "Once: Boosting content-based recommendation with both open-and closed-source large language models," in *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024.

[9] D. Di Palma, "Retrieval-augmented recommender system: Enhancing recommender systems with large language models," in *Proceedings of the 17th ACM Conference on Recommender Systems*, 2023.

[10] H. a. L. C. a. X. N. a. Q. Z. a. Z. S. a. Q. B. a. L. T. Wang, "Huatuo: Tuning llama model with chinese medical knowledge," 2023.

[11] X. a. C. Z. a. L. Y. Yu, "Harnessing LLMs for temporal data-a study on explainable financial time series forecasting," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 2023.