International
Institute of Information
Technology Bangalore

2025

AI HR Assistant

Chatbot to screen the candidate profile based on job description.

Ritesh Kumar Rai

Table Of Content

Contents

1. Project Overview

1.1 Introduction

This document outlines the technical design and implementation of an AI-powered HR Assistant that automates candidate screening. The system allows candidates to submit their profiles, resumes, and job URLs, and then engages them in a chatbot-based screening process. Leveraging Azure OpenAI GPT-4.0 mini model, the system generates relevant screening questions based on the job description extracted from the provided URL. The candidate's responses are analyzed, and an assessment summary, including a score and overall evaluation, is generated.

1.2 Objectives

- Automate the initial candidate screening process.
- Provide a standardized and objective assessment of candidates.
- Improve efficiency and reduce the time spent on manual screening.
- Enhance the candidate experience through an interactive chatbot interface.

1.3 Technologies Used

- Backend: Python FastAPI
- Frontend: ReactJS
- AI Model: Azure OpenAI GPT-4.0 mini model
- Job Description Extraction: Selenium
- Database: PostgreSQL (PgSQL)
- Real-time Communication: WebSockets
- Cloud: Azure (for OpenAI)

2. System Design

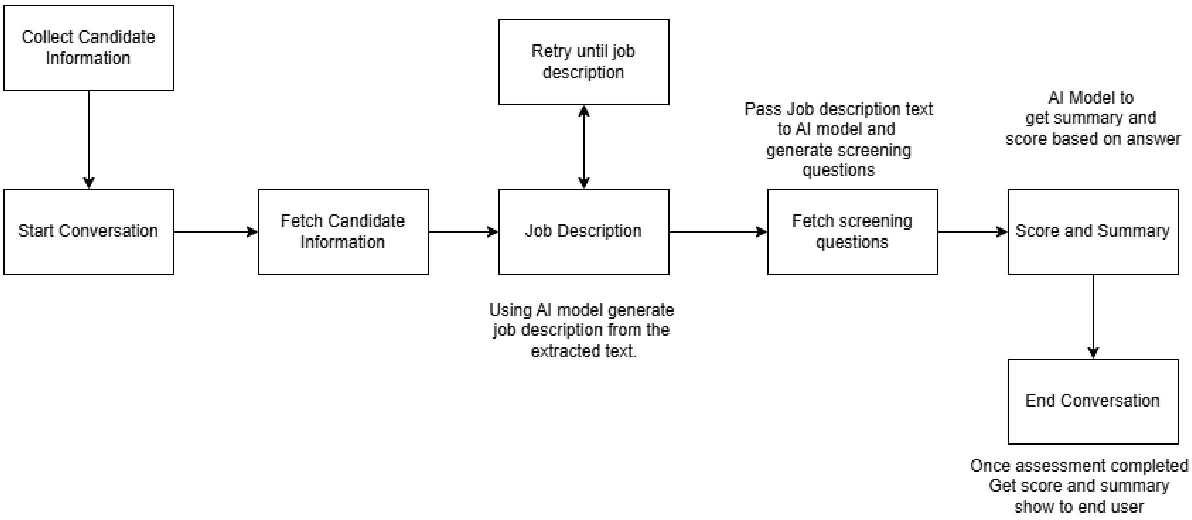
2.1 Components

The system follows a modular architecture, comprising the following components:

- Frontend (ReactJS): A user-friendly interface for candidates to submit their profiles, resumes, and job URLs, and interact with the chatbot.
- Backend (FastAPI): An API layer that handles data processing, job description extraction, question generation, and assessment.
- Azure OpenAI (GPT-4.0 mini model): Generates screening questions and assessment summaries based on job descriptions and candidate responses.
- PostgreSQL: Stores candidate data, questions, answers, and assessments.
- Selenium: Extracts job descriptions from provided URLs.
- WebSockets: Enables real-time communication between the frontend and backend for the chatbot interaction.

2.2 Diagram

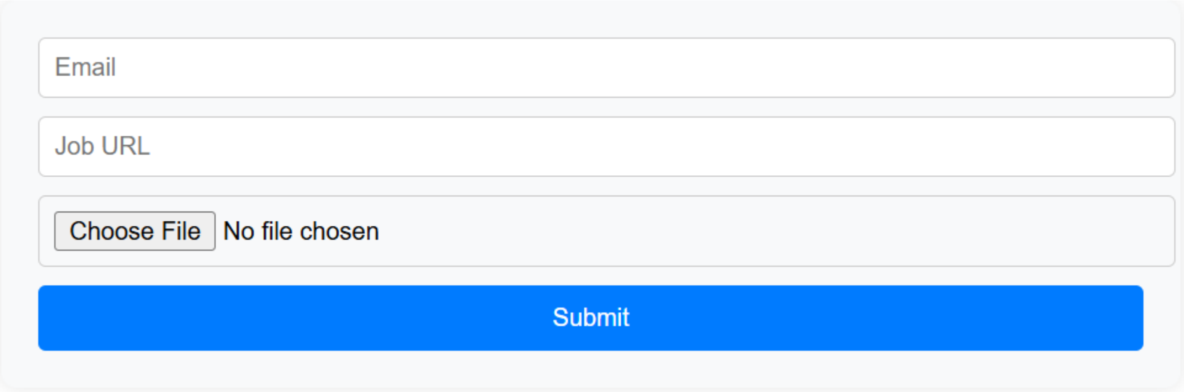
Here's a detailed design diagram representing the system flow, similar to the image you provided:


Collect Candidate Information
Start Conversation
Fetch Candidate Information
Retry until job description
Job Description
Using AI model generate job description from the extracted text.
Pass Job description text to AI model and generate screening questions
Fetch screening questions
AI Model to get summary and score based on answer
Score and Summary
End Conversation
Once assessment completed Get score and summary show to end user

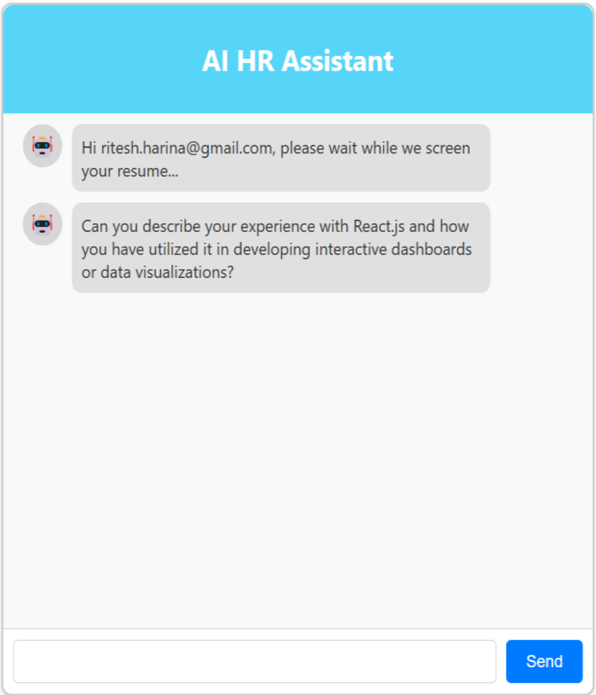
How to run code?

1. Replace pgsql connection string on backend-fastapi folder, main.py file.
2. Add env variable for endpoint and open ai key
3. Run backend application `uvicorn main:app --reload`
4. Navigate to frontend-react folder, run frontend application by `npm start`

3. Demo



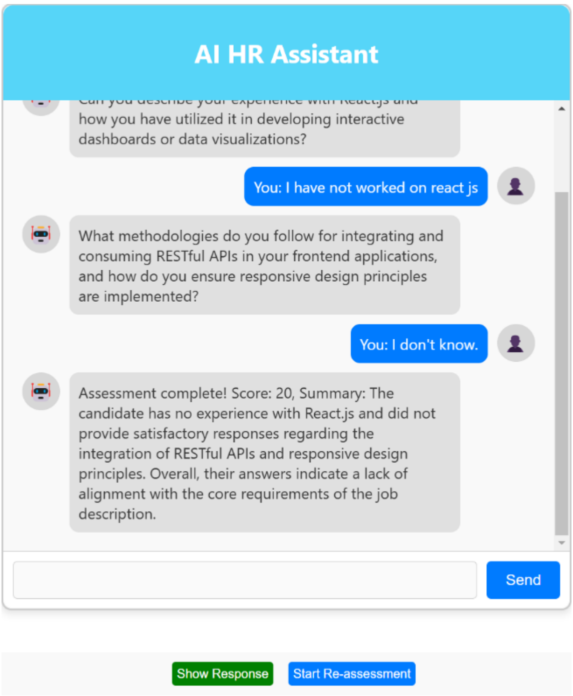
3.1 Capture candidate information



Hi ritesh.harina@gmail.com, please wait while we screen your resume...

Can you describe your experience with React.js and how you have utilized it in developing interactive dashboards or data visualizations?

3.3. Assessment Score



Can you describe your experience with React.js and how you have utilized it in developing interactive dashboards or data visualizations?

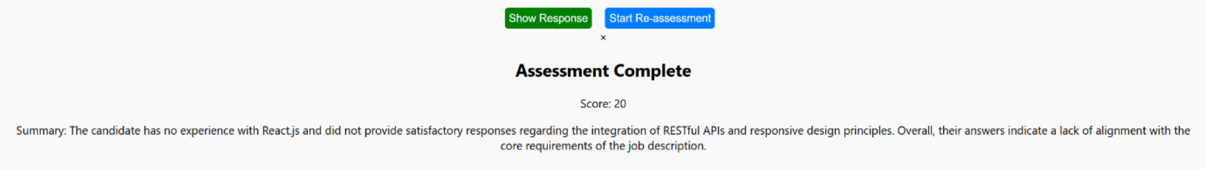
You: I have not worked on react js

What methodologies do you follow for integrating and consuming RESTful APIs in your frontend applications, and how do you ensure responsive design principles are implemented?

You: I don't know.

Assessment complete! Score: 20, Summary: The candidate has no experience with React.js and did not provide satisfactory responses regarding the integration of RESTful APIs and responsive design principles. Overall, their answers indicate a lack of alignment with the core requirements of the job description.

3.2 Assessment screen



Assessment Complete

Score: 20

Summary: The candidate has no experience with React.js and did not provide satisfactory responses regarding the integration of RESTful APIs and responsive design principles. Overall, their answers indicate a lack of alignment with the core requirements of the job description.

3.4 score details

4. Challenges

This project presented both technical and AI-specific challenges that required careful consideration and problem-solving.

3.1 Technical Challenges

- **Job Description Extraction:** Extracting job descriptions from different websites proved to be a significant hurdle. Websites vary widely in their HTML structure, with job postings often embedded within complex layouts and dynamic content. This necessitated a meticulous approach to configuring Selenium, employing techniques like XPath selectors, CSS selectors, and regular expressions to accurately locate and extract the relevant job description text while avoiding extraneous information. Additionally, some websites employed anti-scraping measures that required further adjustments to Selenium's configuration, such as setting user agents and implementing delays to mimic human browsing behaviour.

- **JSON Response Handling:** While the GPT-4.0 mini model generally provided responses in JSON format as requested, it occasionally included additional text outside the JSON structure, which interfered with standard parsing libraries. To address this, custom parsing logic was developed to identify and extract the valid JSON payload from the response, ensuring that the application could reliably process the AI's output. This involved string manipulation techniques and pattern recognition to isolate the JSON object within the response text.

3.2 AI Challenges

- **Ensuring Proper Response Structure from the Bot:** Achieving consistency in the structure and format of the AI's responses proved challenging. While prompt engineering played a crucial role in guiding the model towards the desired output, the inherent variability of large language models meant that responses sometimes deviated from the expected format. This required a combination of prompt refinement, response validation, and in some cases, post-processing of the AI's output to enforce the required structure. For example, if the model failed to generate a JSON object as expected, fallback mechanisms were implemented to extract key information from the response and restructure it into the necessary format.

- **Prompt Engineering for Question Generation:** Crafting effective prompts for generating relevant and insightful screening questions was an iterative process. It required careful consideration of the job description's nuances, the desired level of question difficulty, and the specific skills or qualities being assessed. Prompts were continuously refined based on the quality and relevance of the generated questions, balancing specificity with flexibility to accommodate a range of job descriptions.

5. Lessons Learned

- Prompt Engineering is Crucial: Well-crafted prompts are essential for obtaining accurate and structured responses from AI models.
- Robust Error Handling: Implementing robust error handling for web scraping and API calls is vital for system stability.
- Iterative Development: An iterative approach, with continuous testing and refinement, is crucial for developing complex AI systems.
- Importance of Modular Design: A modular design facilitates easier maintenance and scalability.
- Data Validation and Sanitization: Implementing data validation and sanitization is essential to prevent errors and security vulnerabilities.

6. Conclusion

The AI HR Assistant project demonstrates the potential of artificial intelligence to transform traditional HR processes. By automating the initial candidate screening, the system successfully achieved a more standardized, efficient, and potentially less biased approach to candidate evaluation. This project underscored the critical role of prompt engineering in eliciting desired outputs from large language models, as well as the importance of robust error handling and data validation in ensuring system reliability. The iterative development process allowed for continuous refinement and optimization, leading to a solution that effectively addresses the challenges of candidate screening. This project serves as a valuable case study for organizations looking to leverage AI in HR, highlighting both the potential benefits and the key considerations for successful implementation. Ultimately, the insights gained from this project will inform and guide future AI initiatives, paving the way for more innovative and efficient solutions in the HR domain and beyond.