NLP Interview Web Portal

By

TAN KE YU



FACULTY OF COMPUTING AND

INFORMATION TECHNOLOGY

TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

KUALA LUMPUR

ACADEMIC YEAR

**2024/25**

NLP AI Interview Web Portal

By

TAN KE YU

Supervisor: Mr. Koh Yin Hern

A project report submitted to the

Faculty of Computing and Information Technology

in partial fulfillment of the requirement for the

Bachelor of Software Engineering (Honours)

**Department of Software Engineering and Technology**

Faculty of Computing and Information Technology

Tunku Abdul Rahman University of Management and Technology

Kuala Lumpur

Copyright by Tunku Abdul Rahman University of Management and Technology.

All rights reserved. No part of this project documentation may be reproduced, stored in retrieval system, or transmitted in any form or by any means without prior permission of Tunku Abdul Rahman University of Management and Technology.

# 

Declaration

The project submitted herewith is a result of my own efforts in totality and in every aspect of the project works. All information that has been obtained from other sources had been fully acknowledged. I understand that any plagiarism, cheating or collusion or any sorts constitutes a breach of TAR University rules and regulations and would be subjected to disciplinary actions.

TAN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TAN KE YU

Bachelor of Software Engineering (Honours)

ID: 22WMR01784

Abstract

As AI continues to transform industries, its role in recruitment is becoming increasingly vital. Traditional interview methods often suffer from human bias, inefficiency, and inconsistency, which can impact hiring quality. By using AI in interviews, organizations can assess candidate responses, evaluate skills objectively, and ensure fair assessments. This enhances the efficiency of the recruitment process and also provides data-driven insights to make better hiring decisions, to suits current growing need for accuracy in hiring talents without bias.

To resolve this issue, the developer proposed a NLP Interview Web Portal designed to analyze candidate performance across multiple attributes, including technical skills, communication skills, problem-solving abilities, cultural fit, and professionalism. By using natural language processing techniques including sentiment analysis, keyword extraction, and semantic similarity, the system provides an effective solution for evaluating interview responses.

The portal is developed using ReactJS for the frontend, Node.js for backend integration, and MySQL for database management. Python-powered Jupyter Notebooks are used for NLP analysis, with Streamlit serving as the prototyping platform. Agile methodology is employed throughout the development lifecycle to ensure iterative progress and adaptability to requirements.

The testing phase involves black-box testing to validate functionality and ensure reliability across use cases. There are multiple constraints for this project and further improvement, including enhancing NLP accuracy, more complete functionality can be further studied. This web portal act as an approach to automating and enhancing interview evaluation processes, which aim to provide a efficient and objective solution for recruiters and organizations.

Acknowledgement

This contains acknowledgement to those who have contributed directly or indirectly to the completion of the project. Usually the people to be acknowledged include the project supervisor(s), moderators, family, and those who have given assistance and supports to ensure the success of the project.

**Table of Contents**

[Declaration ii](#_Toc186062776)

[Abstract iv](#_Toc186062777)

[Acknowledgement v](#_Toc186062778)

[Table of Contents v](#_Toc186062779)

[1 Introduction 2](#_Toc186062780)

[1.1 Background 2](#_Toc186062781)

[1.2 Problem Context 2](#_Toc186062782)

[1.3 Rationale 3](#_Toc186062783)

[1.4 Potential Benefits 4](#_Toc186062784)

[1.5 Targeted Users 5](#_Toc186062785)

[1.6 Scope and Objectives 6](#_Toc186062786)

[1.6.1 Scope 6](#_Toc186062787)

[1.6.2 Objectives and functions 6](#_Toc186062788)

[1.7 Project Schedule 6](#_Toc186062789)

[1.8 Overview of this report 7](#_Toc186062790)

[1.9 Chapter Summary and Evaluation 9](#_Toc186062791)

[2 Literature Review 11](#_Toc186062792)

[2.1 Introduction 11](#_Toc186062793)

[2.2 Domain Research 11](#_Toc186062794)

[2.2.1 Interview scoring 11](#_Toc186062795)

[2.2.2 Natural Language Processing Techniques 12](#_Toc186062796)

[2.2.3 Speech-to-text 14](#_Toc186062797)

[2.2.4 Application Programming Interface (API) 16](#_Toc186062798)

[2.3 Similar Systems 17](#_Toc186062799)

[2.3.1 HireVue 17](#_Toc186062800)

[2.3.2 Sapia.ai 18](#_Toc186062801)

[2.3.3 Interviewer.AI 19](#_Toc186062802)

[2.3.4 Comparison between the existing system 20](#_Toc186062803)

[2.4 Technical Research 21](#_Toc186062804)

[2.4.1 Front-End Framework 21](#_Toc186062805)

[2.4.2 Back-End Framework 22](#_Toc186062806)

[2.4.3 Natural Language Processing model (NLP) 24](#_Toc186062807)

[2.4.4 Prototype 26](#_Toc186062808)

[2.4.5 Database Management System (DBMS) 27](#_Toc186062809)

[2.4.6 Web Server 28](#_Toc186062810)

[2.4.7 Development Hardware 29](#_Toc186062811)

[2.5 Chapter 2 Summary 29](#_Toc186062812)

[3 Methodology and Requirements Analysis 32](#_Toc186062813)

[3.1 Agile Development Methodology 32](#_Toc186062814)

[3.2 Core Target 33](#_Toc186062815)

[3.3 Requirement Analysis (Use Case Diagram) 34](#_Toc186062816)

[3.4 Functional Requirement 52](#_Toc186062817)

[3.4.1 Interview candidates 52](#_Toc186062818)

[3.4.2 HR recruiters 52](#_Toc186062819)

[3.4.3 Admin 53](#_Toc186062820)

[3.5 Non-Functional Requirement 53](#_Toc186062821)

[3.5.1 Qualitative Attributes 53](#_Toc186062822)

[3.5.2 Organizational Requirement 55](#_Toc186062823)

[3.5.3 Limitations and Constraints 55](#_Toc186062824)

[3.6 Chapter Summary and Evaluation 56](#_Toc186062825)

[4 System Design 58](#_Toc186062826)

[4.1 System Structure 58](#_Toc186062827)

[4.1.1 Object-Oriented Approach 58](#_Toc186062828)

[4.2 Process Design 59](#_Toc186062829)

[4.3 Software Architecture Design 61](#_Toc186062830)

[4.4 Algorithm Design 62](#_Toc186062831)

[4.4.1 Start Interview Activity Diagram 62](#_Toc186062832)

[4.4.2 Invite Candidate Activity Diagram 64](#_Toc186062833)

[4.5 Data Design 66](#_Toc186062834)

[4.6 Data Dictionary Table 67](#_Toc186062835)

[4.7 UI Design 70](#_Toc186062836)

[4.8 Chapter Summary and Evaluation 96](#_Toc186062837)

[4.9 Reference and Citations 97](#_Toc186062838)

[4.10 Originality report 103](#_Toc186062839)

Chapter 1

Introduction

# Introduction

This chapter provides a brief overview of the development of an AI interview web portal. It includes an analysis of the existing methods for conducting interviews and the challenges associated with these methods. We examine the current interview process and identifies the problems encountered. Furthermore, this paper outlines the proposed system designed to address these issues and the benefits it suggests in terms of efficiency and accuracy in evaluating interview performance.

## Background

GIT corporation, a leading technology solutions provider has encountered significant challenges in its interview process. With rapid growth of job applications, GIT need a more efficiently way to manage the high volume of candidates. The traditional manual interview process has caused inconsistencies and biases, resulting in the potential oversight of qualified candidates. Additionally, the time-consuming nature of scheduling and conducting interviews has strained HR resources, impacting overall productivity. These challenges underscore the need for a more efficient, consistent, and unbiased solution to streamline the interview process and ensure the selection of the best talent.

## Problem Context

In today's competitive job market, organizations strive to enhance the efficiency of their recruitment processes while ensuring they select the most qualified candidates. Traditional interview methods, often rely on subjective human judgment, can be time-consuming and prone to biases. Artificial Intelligence (AI) offers a solution by automating and improving various aspects of the interview process. A 2018 survey by Mercer, a U.S.-based human resource consulting firm, involving 7,300 business leaders and chief personnel managers from global enterprises, revealed that 36% of participating companies utilized AI in their hiring processes to assess applicants' potential for superior performance and long-term retention (Blacksmith et al., 2016).

Firstly, the current interview process is excessively time-consuming due to resume screening. Manually reading through a large volume of resumes to identify candidates with the right skills and experience is inefficient and prone to errors.

Evaluating a candidate's skills and competencies can be highly subjective, heavily relying on the interviewer's personal judgment. This subjectivity can lead to inconsistent and potentially unfair evaluations, as human interviewers may have unconscious biases based on factors like gender, race, age, or educational background. Many hiring decisions are made within the first five to ten minutes of an interview, primarily based on first impressions rather than the candidate's qualifications or capability to perform the job. This process, known as social categorization, often uses illogical, outdated, or even illegal criteria (University of Florida, n.d.).

Additionally, note-taking during interviews can be distracting for interviewers, leading to incomplete or inaccurate records of the conversation, which makes recalling and comparing candidates' responses complicating later.

By analyzing the language used by candidates, recruiters can gain insights into their attitude, enthusiasm, and overall performance, aiding in the assessment of their communication style, professionalism, and cultural fit (Poonam. S, 2024). However, determining a candidate's sentiment, such as enthusiasm, confidence, and honesty, based solely on verbal responses is challenging. These subjective aspects significantly impact the assessment of a candidate's suitability for the role.

Moreover, providing quick follow-up and personalized feedback to candidates is labour-intensive, often resulting in communication delays that negatively impact the candidate experience and the organization's reputation. Detecting inconsistencies in candidate information at different stages of the interview process is also challenging, leading to potential misjudgements about a candidate's honesty or reliability.

Finally, the subjectivity in skills and competency assessment, along with the lack of comprehensive data, make organizations' hard to analyse and improve their interview processes. Without a clear understanding of what works and what doesn't, optimizing and improving interviewing techniques is fairly difficult.

## Rationale

The said problems above can be improved by an AI-powered interview web portal. With a AI specialize in interview process, we can quickly and accurately process large volumes of applications. AI's relevance in recruitment and talent acquisition is growing, offering faster and more precise candidate selection (Zhang, 2024). NLP algorithms can be designed to automatically screen resumes, analyse interview responses, and rank candidates based on predefined condition. This significantly reduces the time and effort required by human recruiters, enabling them to handle a larger number of applications without compromising quality.

Furthermore, AI-driven evaluation systems help minimize human biases by standardizing assessment criteria. NLP tools evaluate candidates based on their responses, ensuring fair treatment regardless of gender, race, age, or background, thereby promote organizational diversity. AI-powered interview tools ask standardized questions to all candidates, ensuring consistent evaluation based on the same criteria and eliminating bias in the interviewing process (Hermele, 2023).

Similarly, NLP algorithms evaluate candidates' skills and competencies based on standardized criteria, reducing subjectivity and potential biases in the evaluation process, leading to more reliable and fair results. With AI’s ability to generate result in a relatively short time, it also help automate follow-up communications and provide personalized feedback to candidates in a timely manner, improving the candidate experience and enhancing the organization's reputation as a responsive employer.

Additionally, the web portal automates the transcription and summarization of interview conversations, eliminating the need for manual note-taking, reducing distractions for interviewers, and ensuring complete and accurate records of each interview. These records can be easily reviewed and compared later, which greatly assist in better decision-making. AI can generate detailed interview reports almost immediately, providing a complete, accurate, and consistently structured record for each candidate (AI Interviews: What Are They and Pros/Cons to Consider, 2019).

Furthermore, AI and NLP can provide valuable insights into the interview process by analysing data from multiple interviews. This data can reveal trends, identify areas for improvement, and help improve the recruitment process. Organizations can use these insights to continuously optimize their interview techniques and strategies.

Finally, automating various aspects of the recruitment process allows organizations to significantly reduce costs associated with manual screening, interviewing, and follow-up. The web portal can handle a larger volume of applications with fewer resources, leading to cost savings and higher efficiency (Doug, 2021). Virtual interviews save expenses related to in-person interviews, such as travel, parking, and accommodation costs, improving the interview experience from the start.

## Potential Benefits

1. The AI interview web portal automated the interview process, reducing the time required to screen and evaluate candidates. This allows recruiters to focus on more strategic tasks.
2. By using standardized criteria and algorithms, the portal ensures that all candidates are assessed fairly and consistently, minimizing human bias.
3. The portal can handle a large volume of applications, making it ideal for organizations with high recruitment needs, such as large enterprises and tech SMEs.
4. The AI provides detailed analytics and insights into candidate performance, helping recruiters make informed hiring decisions based on data rather than subjective judgment.
5. Automating the initial stages of the interview process can reduce the overall recruitment costs by minimizing the need for extensive HR resources and manual efforts.
6. Candidates receive feedback and a more structured interview process, enhancing their overall experience and perception of the organization.
7. The portal helps identify the best candidates more efficiently, ensuring that organizations hired top talent.
8. By relying on AI and predefined criteria, the portal helps reduce unconscious bias, promoting a more diverse and inclusive hiring process.
9. The portal automates repetitive tasks such as collecting candidate information, validating candidate resumes, and provide interview feedback. Saving up recruiters' time.

## Targeted Users

Large enterprises, which have high-volume recruitment needs can benefit significantly from an NLP interview web portal. These organizations often handle thousands of applications for multiple positions across various departments. Consequently, the AI-powered portal can carry out the screening and interviewing processes, saving time and resources while ensuring consistency and fairness.

Moreover, recruitment agencies that manage the hiring process for various clients can use this technology to enhance their service offerings. By quickly identifying top candidates, the web portal can improve client satisfaction and help these agencies maintain a competitive edge in the market.

Additionally, SMEs including our client, GIT, which often faced problem in lack extensive HR departments, can implement the NLP interview web portal to efficiently manage their recruitment processes. The automation and analytical capabilities of the portal allow SMEs to compete with larger companies in attracting and hiring talent.

Similarly, tech startups often face rapid growth and need to scale their teams quickly. The NLP interview web portal can help these companies efficiently manage their hiring processes, ensuring they attract and choose the best talent while maintaining hiring process efficiency.

Furthermore, universities and colleges can also utilize the web portal for faculty hiring, student admissions, and scholarship interviews. The NLP capabilities can assist in evaluating large numbers of applicants and ensuring a fair selection process based on predefined criteria. Additionally, universities can use it as a tool for students to practice their interview skills.

Besides, HR departments within any organization can greatly benefit from the efficiencies and insights provided by the NLP interview web portal. This tool can assist HR professionals in managing large volumes of candidates, reducing biases, and making data-driven hiring decisions.

Furthermore, although not direct users of the web portal, job seekers and candidates benefit indirectly from a more fair and efficient recruitment process. The AI interview web portal can provide feedback, enhance the candidate experience, and ensure that they are evaluated based on their true potential and skills. Moreover, candidates can use the demo portal for mock interviews, allowing them to become familiar with the interview process and practice their skills.

## Scope and Objectives

### Scope

Research, design and develop an AI-driven interview web portal. The web portal help GIT corporation (client) to automate and increasing efficiency for the interview process.

### Objectives and functions

1. Develop web portal that calculate comprehensive score of candidates interview performance.
2. Develop web portal that record candidate answer question-by-question.
3. Develop web portal that convert recorded video into text.
4. Implement AI model to do transcript analysis and generate marks for candidate’s interview performance in each interview scoring attributes.

## Project Schedule

|  |  |  |
| --- | --- | --- |
| **ACTIVITIES** | **EXPECTED OUTCOME** | **COMPLETION DATE** |
| Project I (Y3S1) | | |
| Project Proposal Submission | * Abstract * Problem * Solution * Target Market * Competition/Contribution * Milestones * References | 18/11/2024 (Week 1) |
| Chapter 1: Introduction | * Objective * Project Background * Advantages and Contribution * Project Plan * Project Team & Organization | 25/11/2024 (Week 2) |
| Chapter 2: Research Background | * Project Background and Related Work * Literature Review * Feasibility Study | 05/08/2024 (Week 4) |
| Chapter 3: Methodology and Requirement Analysis | * Methodology * Research Approach * Development Model * Requirement Analysis | 19/08/2024 (Week 5) |
| Chapter 4: System Design | * Process Design * Database Design * Interaction Models * Behavioural Models * System Design Specification * UI Design | 02/09/2024 (Week 6) |
| Project I Portfolio | * Prototype * Portfolio | 17/09/2024 (Week 7) |
| Project II (Y3S2) | | |
| Implementation (Backend Development) | * Model Development * System Development * Backend Development | 16/03/2025 (Week 5) |
| Implementation (Frontend Development) | * UI Design * Frontend Development | 30/03/2025 (Week 7) |
| Testing | * Test Planning * Test Cases * Testing | 06/04/2025 (Week 8) |
| Bugs Fixing and Improvements | * Bugs Fixing * Function improvement | 20/04/2025 (Week 10) |
| Discussion and Conclusion | * Summary * Achievements * Contributions * Limitations and Future Improvements * Issues and Solutions | 11/05/2025 (Week 13) |

Table 1.1: Project Schedule

## Overview of this report

**Chapter 1: Project Overview**

In this chapter, we discuss the project background, the problems associated with the current manual system, and the rationale behind the proposed AI Interview Web Portal. It also covers the potential benefits of the new system, identifies the target users, and outlines the scope and objectives of the project. Additionally, the chapter discuss the deliverables, focusing on the system's functionality, and concludes with a general overview of the project plan, summarizing the development process and expected outcomes.

**Chapter 2: Literature Review**

This chapter presents a literature review, focusing on researches related to the proposed system using resources such as journals, documents, and articles. The developer examines three similar systems, compare and conclude their strength and weaknesses. The chapter also discusses the technical research outcomes, covering the choice of programming languages, Database Management Systems (DBMS), web servers, and comparisons with similar systems.

**Chapter 3: Research Methods**

This chapter outlines the research methods used for data collection and analysis. It includes a detailed description of the questions posed and their objectives, explaining the rationale behind the selected research methods and how they contribute to the overall project development.

**Chapter 4: System Design**

In this chapter, the system design is discussed. It specifies the functional and non-functional requirements and includes diagrams that illustrate the system flow, such as use case diagrams, use case descriptions, activity diagrams, class diagrams, and database table structures. The chapter also covers the system development methodology, detailing the process of selecting the appropriate methodology by comparing various options suitable for the proposed system.

**Chapter 5: System Development and Testing**

This chapter covers the features and functionalities developed during system development, along with a description of the developer’s test-driven development strategy, including the test plan used. It details the implementation of code and ensures all functionalities meet the requirements. The chapter includes screenshots of the system and sample code snippets. Lastly, it discusses system validation to ensure the proposed system meets the requirements and operates without errors, incorporating feedback from initial testers for further enhancement.

**Chapter 6: Challenges Evaluation and Conclusion**

This chapter presents a critical evaluation of the challenges faced during development and provides a summarized version of the project documentation. It highlights the achievements throughout the development process and offers suggestions for future enhancements, reflecting on lessons learned and potential improvements.

## Chapter Summary and Evaluation

This chapter provides an in-depth analysis of the current challenges faced by GIT Corporation in their interview process, including inefficiencies, biases, and resource strain. It discusses the limitations of traditional interview methods and presents the potential of AI and NLP technologies to transform recruitment by automating resume screening, interview scheduling, evaluation, and follow-up. The chapter also list out the benefits of an AI-powered interview web portal in enhancing fairness, consistency, and productivity in the recruitment process. Additionally, it highlights the various targeted users and potential benefits of the system, emphasizing its capacity to handle high-volume recruitment needs efficiently and effectively. Finally, the scope and objectives of developing the AI-interview web portal for GIT Corporation are defined, identified the specific functionalities and goals of the proposed solution.

Chapter 2

Literature Review

# Literature Review

## Introduction

As the 4th Industrial Revolution advances, the importance of talented individuals has become more in demand. Organization, including GIT corporation are increasingly in need of the ability to cultivate skilled talent.

The development of Artificial Intelligence (AI) has significantly changed various sectors, including recruitment. Traditional methods of conducting interviews, which rely heavily on human judgment, are increasingly being seen biased. Therefore, it is suggested to be replaced by AI models. Most of these AI models implement natural language processing (NLP) and other AI technologies to assess candidates' suitability for specific roles based on their responses during interviews. (B C Lee & Kim, 2021)

## Domain Research

### Interview scoring

Interview scoring is normally based on a combination of quantitative and qualitative metrics to evaluate candidate performance. These metrics are derived from studies in industrial-organizational psychology and human resources management, such as Borman and Motowidlo's (1993) research on job performance dimensions //need to find actual research. Common scoring criteria include communication skills, problem-solving abilities, technical expertise, and cultural fit. These attributes are assessed using structured methods like competency-based scoring or behavioral interviews. Scoring frameworks ensure that evaluations are consistent, objective, and aligned with the role’s requirements. For example, STAR (Situation, Task, Action, Result) methodology is widely used to assess how candidates handle past scenarios relevant to the job. (Apple et al., 2020)

Additionally, many scoring systems also consider non-verbal cues and overall presentation. Studies such as Ibrahim et al. (2022) work on non-verbal communication highlight the impact of body language and tone on interpersonal interactions, making these crucial factors in interview assessments. Interviewers often score candidates on attributes like confidence, adaptability, and emotional intelligence based on observed behaviors and responses.

As a results, organizations often assess candidate in terms of their technical skills, communication skills, problem-solving skills, cultural fit, and professionalism.

### Natural Language Processing Techniques

Natural Language Processing (NLP) is a powerful tool for automating and enhancing the interview evaluation process. By analyzing textual responses, NLP systems can assess candidates on technical skills, problem-solving abilities, and interpersonal skills. Different NLP techniques are used for specific evaluation points, including analysis that improve objectivity, efficiency, and fairness. This section discusses the key NLP techniques and their application to various scoring points which are discussed in section 2.1.1.

**Keyword Extraction**

Keyword extraction identifies domain-specific terms and critical keywords in candidate responses to evaluate their technical knowledge and problem-solving abilities. Furthermore, for instance, in a software engineering interview, the system extracts and analyses terms like “database normalization,” “API integration,” or “algorithm optimization.” Techniques like Named Entity Recognition (NER) and Term Frequency-Inverse Document Frequency (TF-IDF) are employed to detect and rank relevant keywords. Initially, candidate responses are preprocessed by cleaning text (e.g., removing stop words and punctuation) and tokenising sentences into manageable units. Then NER models built on transformer architectures such as BERT, are applied to identify domain-specific terms by recognising entities relevant to the domain, such as technical concepts or methodologies. Simultaneously, TF-IDF models assign weights to words based on their frequency in the candidate's response relative to a broader corpus, highlighting terms that are uniquely significant. Moreover, these extracted terms are compared against a predefined domain-specific knowledge base to evaluate their relevance. The system matches the candidate's vocabulary with the required technical concepts, scoring their expertise accordingly. Additionally, dependency parsing is used to analyse grammatical structures, identifying relationships between keywords and their contexts. For instance, this helps assess how a candidate describes processes, such as linking "database normalization" with "efficiency improvement."

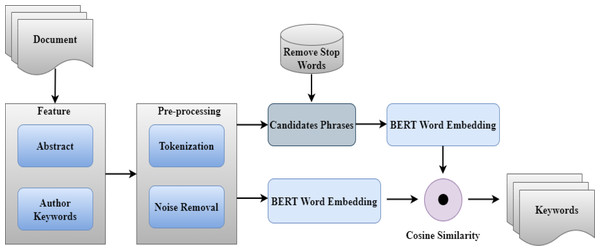


Diagram 2.1: Keyword extraction structure

**Sentiment Analysis**

Sentiment analysis evaluates emotional attributes in candidate responses, providing insights into interpersonal skills and cultural fit. Moreover, by analysing features such as word choice, clarity, and phrase structure, the system assesses traits like empathy, teamwork, and adaptability. Pretrained sentiment analysis models are often implemented, such as those based on transformer architectures (e.g., BERT or RoBERTa), are fine-tuned on datasets with labelled interview responses. These models analyse textual inputs to determine emotional polarity (e.g., positive, neutral, or negative). For example, during inference, candidate responses are tokenised into input embeddings compatible with the transformer model. Subsequently, the model processes these embeddings, leveraging its attention mechanism to understand contextual nuances. The output layer then assigns sentiment scores based on the trained parameters (G. Sri Harsh et al., 2024).

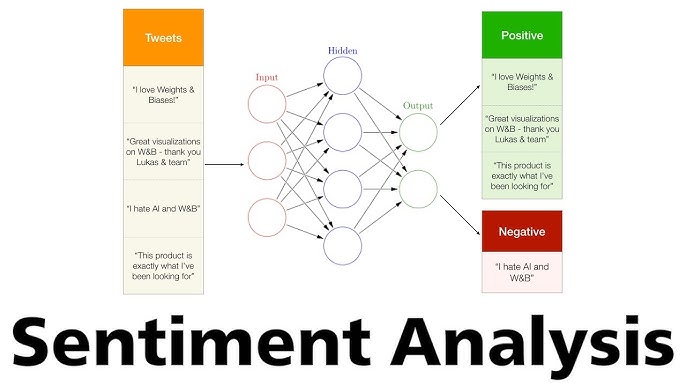


Diagram 2.2: Sentiment analysis structure

**Semantic Similarity**

Semantic similarity models compare a candidate's responses with ideal answers to evaluate technical expertise and problem-solving skills. Furthermore, implementation involves using transformer-based models like Sentence-BERT or embedding-based similarity measures to quantify how closely a response aligns with expected answers. In one of the technique, the system calculates cosine similarity between the candidate's response and predefined model answers, giving higher score for contextually aligned and logical explanations. Normally, pretrained models like Sentence-BERT are utilised to encode both candidate responses and ideal answers into dense vector representations. After text normalisation and tokenisation, the transformer-based architecture then generates sentence embeddings by capturing contextual and semantic relationships between words, ensuring nuanced understanding. Next, the system computes similarity metrics, such as cosine similarity, to determine the alignment between vectors. Cosine similarity measures the angle between the vector representations, where a smaller angle indicates a closer match in meaning. The similarity scores are then used to rank responses based on their relevance to predefined ideal answers. Additionally, embedding-based similarity measures can be enhanced by implementing domain-specific fine-tuning. For example, the model is trained on a dataset of technical questions and answers relevant to the role, improving its ability to recognise domain-specific terminology and logical reasoning. Finally, the output scores are generated to assess a candidate's understanding of technical concepts and problem-solving approach (Dhivya Chandrasekaran & Vijay Mago, 2021). Responses with higher semantic alignment and logical coherence are ranked favourably, ensuring a comprehensive evaluation of technical expertise and practical problem-solving ability.

### Speech-to-text

Speech-to-text technology plays an significant role in AI interview analysis systems by converting spoken language into text for further analysis. In this context, the primary focus is on accurately transcribing candidate responses. Several Application Programming Interfaces (APIs) and libraries are available for implementing speech-to-text functionalities, including Google Cloud Speech-to-Text, IBM Watson Speech to Text, and Microsoft Azure Speech.

Google Cloud Speech-to-Text is a popular choice due to its high accuracy, support for over 120 languages, and ability to handle background noise effectively. It implements advanced machine learning models, including deep learning techniques, to continuously improve transcription accuracy. One of its key advantages is the real-time streaming transcription which is suitable for the interview analysis system. However, its usage can be costly, particularly for high volumes of data. (IANCU, 2019)

IBM Watson Speech to Text also offers strong transcription capabilities with an emphasis on customizability. Watson allows users to train models specific to their domain or industry, improving transcription accuracy for specialized vocabularies. This flexibility is suitable for interview systems that deal with technical or industry-specific transcripts. Its limitation is the customization process can be complex and time-consuming. (Noel, 2019)‌

Microsoft Azure Speech provides comparable transcription accuracy and multilingual support, with smooth integration into the broader Azure ecosystem. It offers features such as speaker recognition and sentiment analysis, which may be beneficial for deeper interview analysis. Its disadvantage is that its performance can be less consistent with non-standard accents or in noisy environments, compared to Google’s solution. (Stoyan Stoitsev, 2023)

There are multiple challenges, such as handling various accents, speech impediments, and audio quality issues, which can affect transcription precision. Moreover, these APIs often involve high usage costs when processing large datasets, making budget considerations crucial for long-term use.

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Google Cloud Speech-to-Text | IBM Watson Speech to Text | Microsoft Azure Speech |
| Supported Languages | Over 125 languages and variants | Over 30 languages and variants | Over 90 languages and variants |
| Real-Time Transcription | Yes | Yes | Yes |
| Batch Processing | Yes | Yes | Yes |
| Customisation | Custom language models and pronunciation tuning | Custom language models and acoustic tuning | Custom Speech models with fine-tuning |
| Noise Handling | Advanced noise filtering | Moderate noise filtering | Advanced noise cancellation |
| Punctuation and Formatting | Automatic punctuation and basic formatting | Automatic punctuation with customisation | Automatic punctuation and full formatting |
| Speaker Identification | Supports speaker diarisation | Supports speaker diarisation | Supports speaker diarisation |
| Integration Support | REST API, gRPC, multiple SDKs | REST API, SDKs for major platforms | REST API, SDKs for multiple programming languages |
| Pricing Model | Pay-per-minute | Pay-per-minute or subscription plans | Pay-as-you-go or reserved plans |
| Offline Capabilities | Limited (via edge devices or partners) | Limited (requires Watson Studio Edge) | Available with on-device support |
| Region Availability | Global | Global | Global |
| Security and Compliance | GDPR, HIPAA-compliant | GDPR, HIPAA-compliant | GDPR, HIPAA-compliant |
| Unique Features | Domain-specific adaptation, multi-channel audio | AI-powered keywords spotting | Real-time translation, Speech SDK for edge |

Table 2.3: Comparison of multiple speech-to-text API

### Application Programming Interface (API)

APIs play an important role in web portal development by enabling smooth communication between the portal and external systems or databases. They allow the portal to fetch, create, update, or delete data seamlessly, offering essential functionality for dynamic user experiences. APIs also facilitate third-party integrations, such as payment processors, social media platforms, or analytics tools, which can enhance the portal’s features. Furthermore, APIs are crucial for enabling real-time functionality, like live notifications or data feeds, providing users with up-to-date information.

Different types of APIs serve different purposes in web portal development. RESTful APIs are the most commonly used, including HTTP methods like GET, POST, PUT, and DELETE to interact with data in a stateless manner, usually formatted in JSON or XML. GraphQL is another popular API type, allowing clients to request exactly the data they need, making it efficient for complex data queries. SOAP APIs, which is less flexible, are XML-based and often used in enterprise systems due to their rigidity and security. Webhooks, an event-driven APIs that send data to a specified URL when certain actions occur, making them ideal for real-time updates (AltexSoft, 2020).

APIs has several use cases in web portals, such as authentication, data management, third-party integration, and analytics. For authentication, APIs enable secure login and access control mechanisms using OAuth, JWT, or API tokens. They support data management by providing CRUD (Create, Read, Update, Delete) operations for handling user profiles, products, or other resources. APIs also enable integration with external services like payment gateways, email platforms, or social media sharing tools, broadening the portal’s capabilities. Additionally, APIs can be used for analytics, helping collect and visualize usage data with tools like Google Analytics (Michael Goodwin, 2024).

To use an API in a web portal, developers need to integrate it into both the back-end and front-end systems. On the back-end, APIs can be created or connected using frameworks like Node.js, Django, or ASP.NET, often linked to databases like MySQL or MongoDB. On the front end, JavaScript tools like `fetch` or Axios can be used to make API requests, allowing for dynamic interaction between the user interface and server. Testing APIs with tools like Postman or Swagger is essential to ensure proper functionality (Hall, 2024). Security measures such as using HTTPS and validating inputs, can protect the system from potential threats from API.

AI systems are often deployed as APIs to make their advanced functionalities accessible to other applications without requiring developers to handle the complexities of AI. By hosting AI models on cloud platforms like AWS, Google Cloud, or Microsoft Azure, AI APIs can process requests and return results efficiently. To integrate an AI API into a system, the portal sends data to the AI API’s endpoint (e.g., a user’s text input or uploaded file) in a structured format like JSON. The API processes this input using its model and returns the result, such as a sentiment analysis score, predictions, or summaries. For example, an NLP API like OpenAI’s GPT processes interview transcripts to generate insights, which the web portal then displays to users. The integration can be achieved using HTTP calls in the back-end or directly in the front-end with libraries like Axios or Fetch (Amazon Web Services, Inc., 2023).

## Similar Systems

### HireVue

HireVue is a widely used AI-driven platform that combines video interviewing with advanced assessment technology to enhance the recruitment process. It acts as a solution for hiring specialize for focusing on efficiency and fairness. The platform allows candidates to record responses to predefined interview questions through video, which are then analyzed by AI. The analysis evaluates verbal responses, tone, language usage, and non-verbal cues like facial expressions and eye movement. Additionally, HireVue offers game-based assessments to evaluate candidates' cognitive abilities, emotional intelligence, and problem-solving skills. These features provide a comprehensive view of a candidate's potential.

One of the outstanding aspects of HireVue is its use of natural language processing (NLP) to process candidates' language for fluency, clarity, and content. The platform generates detailed reports with scores and rankings, enabling recruiters to make data-driven decisions. Moreover, It uses predictive analytics to identify candidates that are likely to succeed based on historical hiring data. HireVue also support integrating with Applicant Tracking Systems (ATS) and is scalable across industries and job levels, making it suitable for enterprises of all sizes.

The platform’s benefits is as follows: It automates time-consuming tasks such as initial screenings and scheduling, significantly reducing the time-to-hire. By using data-driven assessments, HireVue aims to minimize unconscious bias, ensuring a fairer hiring process. It also enhances the candidate experience with flexible interview options, allowing individuals to complete assessments at their convenience. Companies like Nike, Hilton, Unilever, and Delta Airlines have adopted HireVue to handle high-volume recruitment needs, with use cases spanning industries like retail, hospitality, healthcare, and technology (HireVue, n.d.).

### Sapia.ai

Sapia.ai is an innovative AI-powered recruitment platform that focuses on enhancing hiring process by improving fairness and reducing bias. Unlike traditional recruitment methods, Sapia.ai uses conversational AI to evaluate candidates through a structured set of text-based interview questions. By relying solely on text communication, the platform eliminates the need for video, audio, or facial recognition, reducing the risk of bias based on appearance or accents. This unique approach ensures that candidates are assessed purely on their responses, creating a more inclusive hiring process.

The platform’s core functionality are mostly chat-based interviews. Candidates carry out in a conversation with the AI, then respond to open-ended questions related to the job role. Sapia.ai then analyzes these responses using advanced linguistic and sentiment analysis to assess personality traits, problem-solving abilities, and cultural alignment. The system generates scores and rankings for candidates based on these metrics, providing recruiters with candidate’s suitability for the position. Additionally, candidates receive personalized feedback after the interview, highlighting their strengths and areas for improvement. This feature enhances transparency and leaves candidates with a positive impression, regardless of the hiring outcome.

Sapia.ai offers several benefits for organizations. Since it is based on text-only responses, it eliminates biases related to gender, ethnicity, or appearance, promoting fairness. The platform is also highly scalable, capable of handling large volumes of candidates simultaneously, which makes it ideal for high-volume recruitment processes. Automated assessments save time and resources by reducing the need for initial screenings. Furthermore, its ability to integrate with Applicant Tracking Systems (ATS) allows recruiters to manage the entire process efficiently from one platform. By aligning candidates’ responses with the skills and attributes needed for the role, Sapia.ai helps organizations make data-driven hiring decisions that improve fairness and performance.

Many organizations have adopted Sapia.ai across industries such as retail, hospitality, and customer service. Notable companies like Qantas and Starbucks in Australia utilize the platform to modernize their recruitment processes. Its scalability also makes it suitable for hiring across all levels, from entry-level roles to executive positions. By identifying candidates who align with a company’s culture and values, Sapia.ai ensures better hiring outcomes and long-term success (Sapia, 2024).

### Interviewer.AI

Interviewer.AI is an AI-powered video interview platform designed to simplify the hiring process while providing recruiters with insights into candidates’ abilities and potential. It combines artificial intelligence with video technology to create a recruitment tool to simulate physical interview. The platform is suitable for roles requiring strong communication skills, problem-solving abilities, and cultural alignment, ensuring that all the skills are assessed.

The platform’s core functionality are automated video interviews, where candidates record responses to predefined interview questions, then analyzed by AI. The system evaluates verbal communication, tone, and non-verbal cues such as facial expressions and gestures. These responses are further analyzed using trained models, and candidates are assigned a performance score based on various factors, including clarity of thought, emotional intelligence, and alignment with role-specific requirements. This data-driven approach allows recruiters to make decisions efficiently.

Interviewer.AI offers wide range of customization options, enabling recruiters to create specialized interview questions to evaluate technical skills, industry knowledge, or soft skills specific to a role. It also generates detailed reports and rankings for each candidate, providing AI-driven insights and predictive analytics to help identify top talent. The platform is also able to integrate with Applicant Tracking Systems (ATS), connect the hiring process by centralizing candidate data and enabling recruiters to manage interviews from a single platform.

Interviewer.AI’s benefits are as follows: The platform improves hiring efficiency by automating the initial screening stages, saving time for both recruiters and candidates. It is scalable, making it suitable for enterprises handling high volumes of candidates. Additionally, the flexible interview process allows candidates to complete assessments at their convenience, enhancing their overall experience. By standardizing interviews and evaluations, Interviewer.AI minimizes unconscious bias, promoting a fairer hiring process. The focus on customizing specialize skills and competencies ensures high-quality hiring outcomes and lower the risk of mismatches (Interviewer.AI, 2024).

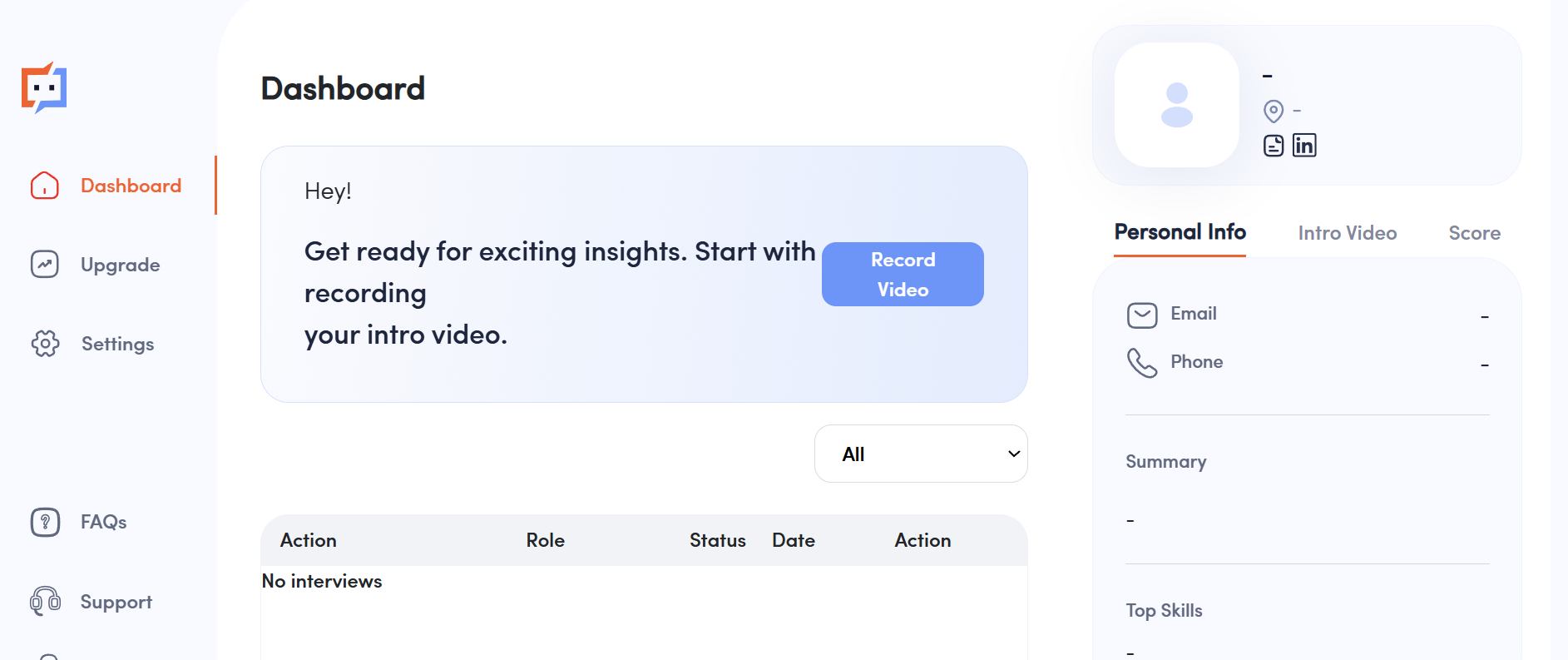


Diagram 2.4: Interviewer.AI website

### Comparison between the existing system

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature/Aspect** | **HireVue** | **Sapia.ai** | **Interviewer.AI** |
| **Overview** | Combines video interviews and assessments to enhance recruitment efficiency and fairness | Text-based interviews to eliminate bias from video, audio, or appearance | Video interview platform simulating physical interviews for comprehensive assessments |
| **Core Functionality** | Video interviews analyzed for verbal, tonal, and non-verbal cues | Chat-based interviews analyzed using linguistic and sentiment analysis | Automated video interviews evaluating verbal, tonal, and non-verbal cues |
| **Additional Features** | - Game-based assessments for cognitive abilities, emotional intelligence, and problem-solving  - Predictive analytics based on historical data  - ATS integration for centralized recruitment | - Personalized feedback for candidates  - ATS integration for centralized recruitment | - Customizable interview questions tailored to specific skills or industry needs  - Predictive analytics for identifying top talent  - ATS integration for centralized recruitment |
| **Key Technologies Used** | - Natural Language Processing (NLP) for fluency, clarity, and content evaluation | - Linguistic and sentiment analysis | - AI-driven performance scoring based on clarity, emotional intelligence, and role alignment |
| **Benefits** | - Enhances efficiency by automating initial screening  - Minimizes unconscious bias  - Enhances candidate experience with flexible interview options | - Enhances efficiency by automating interview assessments  - Promotes fairness by eliminating biases related to gender, ethnicity, or appearance  - Scalable for high-volume recruitment | - Enhances efficiency by automating initial screening and standardizing evaluations  - Customization ensures high-quality matches  - Flexible interview process enhances candidate convenience |
| **Industries/Use Cases** | - Retail, hospitality, healthcare, and technology | - Retail, hospitality, and customer service | - Suitable for roles requiring communication, problem-solving, and cultural alignment |

Table 2.5: Comparison of existing system

## Technical Research

### Front-End Framework

ReactJS is a popular JavaScript library developed by Facebook, designed for building user interfaces, particularly for single-page applications. Its component-based architecture allows developers to create reusable UI components, making the development process efficient and the code more maintainable. It is widely used due to its high performance, flexibility, and a large ecosystem of tools and libraries. ReactJS follows a component-based architecture, where the entire application is divided into small, reusable, and independent components (React, n.d.). These components are organized hierarchically, achieving a clean separation of concerns. The architecture consists of components, props, state, context, lifecycle methods and Virtual DOM.

Components are the building blocks of a React application, each representing a specific part of the UI. Components can be stateful (managing their own data) or stateless (receiving data via props). Props, which is short for "properties," are read-only inputs passed from parent to child components (unidirectional). State is a component’s internal data that can change over time, which it triggers re-renders to reflect updates in the UI. Context is a structure that provides a way to share data across components without passing props manually at every level. Lifecycle Methods are hooks provided by React to handle component lifecycle events. For examples, `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount` in class components. Virtual DOM acts as a lightweight copy of the actual DOM. React updates the Virtual DOM first, compares it with the previous version through a process called "diffing," and applies minimal updates to the real DOM for optimal performance (Vora, 2024). This architecture ensures modularity, reusability, and scalability, making ReactJS suitable for building complex, high-performance applications.

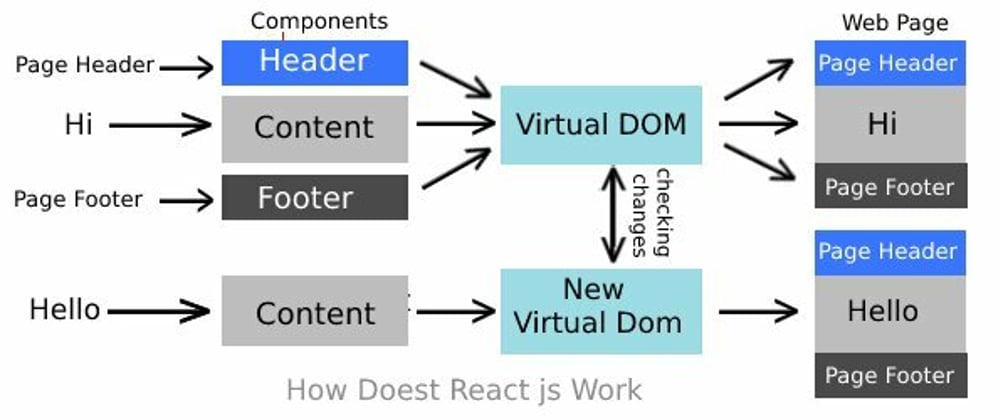


Diagram 2.6: Architecture of ReactJS

ReactJS is a suitable choice for AI interview analysis system. Firstly, its ability to create dynamic and interactive UIs makes it an ideal choice for an AI interview analysis system. Features like real-time performance dashboards, visual analytics, and dynamic feedback are supported by ReactJS. Moreover, AI-driven technologies and analytics often involve in frequent updates. React’s Virtual DOM ensures these updates are efficiently rendered, providing a smooth user experience even with frequent update operations. Additionally, React can also be integrated with existing backend services, such as APIs for processing interview data or NLP models. This flexibility also makes it compatible with a range of technologies, including Node.js, Python (Flask/Django), or Java-based APIs. Which is suitable for current scenario since the backend for this system is planned to use Node.js. Furthermore, since the system requires advanced visualizations or specialized tools for data representation, React’s ecosystem supports libraries like Chart.js, D3.js, and Recharts. These libraries can be used to create compelling data visualizations for AI models. Moreover, React’s component reusability and clear syntax reduce development time and complexity. Lastly, the large React community have a large number of resources, from tutorials to third-party libraries, enabling rapid development and troubleshooting. Regular updates from Facebook make React follow up-to-date features and solutions (Angelina, 2024).

### Back-End Framework

Node.js is selected as backend framework of AI interview analysis web portal. Node.js is an open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside a web browser. It is built based on Google Chrome's V8 JavaScript engine, which can achieve quick runtime and achieve high performance. It uses an event-driven, non-blocking I/O model, which makes Node.js a lightweight and efficient solution. It includes several features: event-driven, single threaded, non-blocking, modules, core modules and npm. Node.js is an event driven framework, it listens for events and executes functions when those events occur. For example, when a file finishes loading, an event triggers the callback function to process it. Moreover, it is single-threaded which handle all the request in one thread. It doesn't wait for operations like file reading or database queries to finish; instead, it moves on to handle other tasks, making it fast and efficient. Furthermore, Node.js uses non-blocking I/O model, which the execution of other code is not blocked while waiting for tasks like I/O operations. This allows it to handle many requests simultaneously. Modules are used to organize code into separate files. Each module can export and import functionality, making the code more manageable. For Core Modules, built-in modules like http, fs, and path help developer to perform common tasks without needing extra libraries. Node.js also uses Node Package Manager (npm) to manage library or packages in projects. It makes it easy to add functionality to application by downloading packages (Adedimeji, 2023).

Basic Structure of a Node.js App:

* app.js or server.js: The main file that starts the server and routes requests.
* routes/: Contains files for defining the paths and what happens when those paths are accessed.
* models/: Defines how data is structured, like in a database.
* controllers/: Contains logic for processing requests and interacting with the models.
* public/: Stores static files (images, stylesheets, etc.).
* node\_modules/: Where external packages installed via npm are stored.

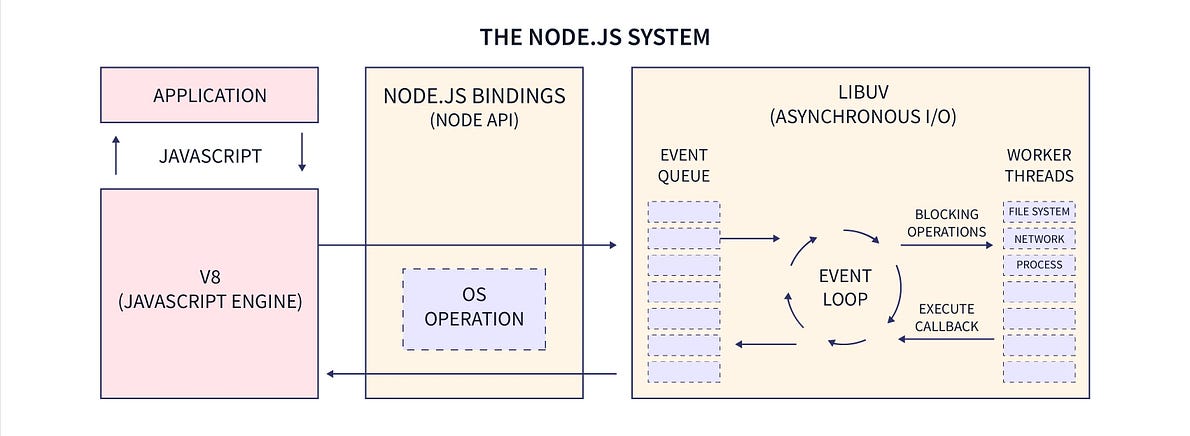


Diagram 2.7: Architecture of Node.js

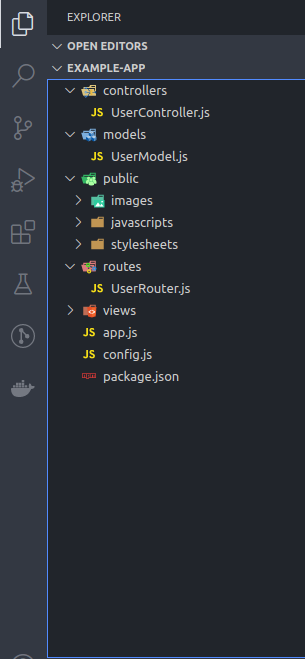


Diagram 2.8: File structure of Node.js

Node.js is normally combined with frameworks like Express.js, which can help make API development easier and quicker. With this, developers can set up RESTful APIs to connect the ReactJS frontend with the backend in shorter time. Middleware for input validation, authentication, and logging, ensuring secure and organized API operations are also easy to implemented using Express.js. Furthermore, it can integrate well with Python-based NLP services via REST APIs. Node.js also have a large ecosystem supported by npm, which offer variety of pre-built libraries and modules. For example, Express.js for server-side routing, Mongoose for managing database interactions and Axios for handling API communications (mdn, 2024).

### Natural Language Processing model (NLP)

The Natural Language AI model is built in python. It is built based on features including: text-to-speech, keyword extraction, semantic similarity and sentiment analysis.

**Text-to Speech**

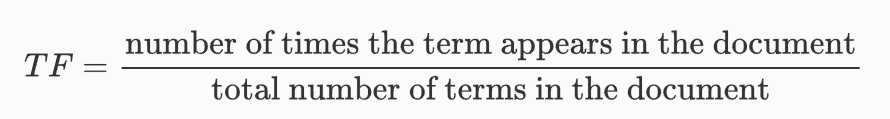
Google Speech Recognition API is used to achieve text-to speech transcript generation of interview. The Google Speech Recognition API is a cloud-based service provided by Google that converts spoken language into text. It is based on Google’s advanced machine learning models to perform accurate speech-to-text transcription in real-time or from pre-recorded audio files. It supports over 120 languages and variants including English, Indonesian, Arabic, Chinese and many more. Google Speech Recognition handles pre-recorded audio files which can be used for offline analysis or large-scale transcription. Moreover, it automatically adds punctuation marks like periods and commas to transcriptions, which help to improve readability. Moreover, it also support contextual biasing which allow customization for specific use cases by giving context and hints about expected words or phrases, improving transcription accuracy in specific domains. It also works well in noisy environments using its advanced filtering techniques and improving recognition accuracy. However, it is dependency on Internet which will require a stable internet connection for communication with the cloud. Furthermore, free tier which is used in this development have a limitation of 10Mbs file size or 1 minute length for uploaded audio file, whichever condition is reached first (Google, n.d.).

**Keyword Extraction**

For keyword extraction, scikit-learn is used. Scikit-learn is one of the most popular open-source machine learning libraries for Python. It is built on top of NumPy, SciPy, and matplotlib, and is designed for a variety of applications including classification, regression, clustering, preprocessing, and feature extraction. It provides a unified and consistent API for many machine learning models. Additionally, wide range of algorithms is provided in scikit-learning. Supporting supervised, unsupervised, and semi-supervised learning. Most importantly, it includes tools for text feature extraction like TfidfVectorizer. Finally, it work well with other Python libraries like Pandas and NumPy.

The TfidfVectorizer is a feature extraction tool within Scikit-learn that converts a collection of raw text documents into a matrix of TF-IDF (Term Frequency-Inverse Document Frequency) features. It is used for text mining and natural language processing (NLP) tasks, including extracting important terms (keywords) from a document and preparing text data for machine learning models. TF-IDF is a measure of how important a word is in a document relative to a collection (corpus) of documents. It emphasizes unique and meaningful words while downplaying common words (Karabiber, n.d.). Simplified formula used is as follows:

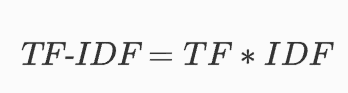
Term Frequency (TF):



Inverse Document Frequency (IDF):



TF-IDF Score:



Finally, the system then ranks the keywords according to TF-IDF Score calculated.

**Sentiment Analysis**

Hugging Face library is used for sentiment analysis, it is based on a pre-trained models hosted on the Transformers library. These models allow developers to integrate advanced sentiment analysis models into projects with minimal effort. Hugging Face host many pre-trained models from state-of-the-art research, including distilbert-base-uncased-finetuned-sst-2-english (for English) and nlptown/bert-base-multilingual-uncased-sentiment (for multilingual support). Moreover, Hugging Face offers customizability by allowing developers to fine-tune pre-trained models on domain-specific data to achieve improved accuracy in specialized tasks. This flexibility allows models to adapt to unique requirements, enhancing their performance (Growth Acceleration Partners, 2024).

**Semantic Similarity**

The semantic similarity model is also based on Hugging Face library. The model "sentence-transformers/all-MiniLM-L6-v2" contains 22.7 million parameters and is capable of mapping sentences and short paragraphs into a 384-dimensional dense vector space. It is specifically designed for tasks like clustering and semantic search. This model serves as an encoder for sentences and short paragraphs, producing a vector that represents the semantic content of the input text. The resulting sentence vector can be used for information retrieval, clustering, or assessing sentence similarity. By default, any input text exceeding 256 word pieces is shortened (Pi, 2024).

### Prototype

Streamlit is an open-source Python library for creating interactive, data-driven web applications quickly and easily. Streamlit applications are Python scripts, enabling developers to define logic, layout, and interactivity using Streamlit’s API. A single Python file typically serves as both the frontend and backend, allowing integration with Python libraries for AI model. The Streamlit server executes the Python script, manages user sessions, handles dependencies, and ensures that the app re-runs dynamically whenever users interact with widgets like sliders or text inputs. Widgets maintain their state in script re-runs, which help support communication between the frontend and backend (Streamlit, n.d.). Streamlit offers ease of use as it simplifies building web applications by allowing developers to write python code without requiring knowledge of HTML, CSS, or Javascript. Instead of spending time building a complex frontend, developers can focus on building AI model outputs. It also supports interactivity including widgets like sliders, dropdowns and input boxes to make application interactive. Real time updates is also one of the feature. Changes of python script reflects immediately on interface. It also work seamlessly with python libraries like NumPy, pandas, matplotlib and Tensorflow, make it suitable for AI related projects.

### Database Management System (DBMS)

A database is an organized collection of data that allows for efficient storage, retrieval, and management of information. It is designed to handle large volumes of structured or unstructured data, making it accessible for various operations like querying, updating, and analyzing. Databases ensure data integrity, consistency, and security, enabling users and applications to work with information. Developer chooses relational database for this scenario due to its ability to handles structured data with well-defined relationships, such as candidate profiles, interview details, questions, responses, and sentiment analysis results. The system requires the ability to perform complex queries across multiple tables, which relational databases provide through features like SQL.

MySQL is a popular open-source relational database management system (RDBMS) due to its high performance, reliability, and ease of use. It is widely used because it efficiently handles structured data for systems that require complex queries and relationships between data entities. Additionally, its active community, well-structured documentation, and compatibility with various programming languages make it a popular choice for developers.

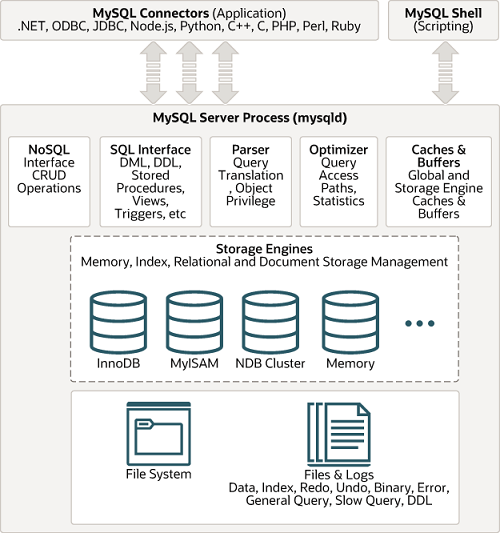


Diagram 2.8: Architecture of MySQL

MySQL’s architecture is a layered design that consists of three main components: the Client Layer, Server Layer, and Storage Layer. The Client Layer manages interactions between users or applications and the database through SQL queries, providing connections for high performance. The Server Layer handles core functionalities such as query parsing, optimization, caching, and execution. It also includes features like stored procedures, triggers, and user management. The Storage Layer is responsible for data storage and retrieval, supporting multiple storage engines like InnoDB (for transactions and ACID compliance) and MyISAM (for faster read-heavy operations). This modular design allows MySQL to be efficient, flexible and scalable for various use cases (Oracle, n.d.).

### Web Server

Node.js web server itself supports web server component. Different from traditional web servers like Apache or Nginx, which are designed to handle HTTP requests and serve static files, Node.js is an event-driven, non-blocking I/O model that is particularly suited for real-time applications and API services. Node.js itself can act as both the application and web server, normally it is used together with frameworks like Express.js for development. Express.js can simplify the process of building web servers and APIs, such as routing, middleware and error handling. Routing manage HTTP request and define routes (e.g. GET, POST, PUT). Middleware will perform tasks like authentication, logging, or parsing JSON data. While error handling automatically handles errors that occur during request processing (Express, n.d.).

WebSockets are built into Node.js and provide a full-duplex communication channel. This is useful for applications requiring constant updates, such as notifying interviewers about new analysis results or displaying real-time sentiment analysis to the user during an interview (Campbell, 2024).

### Development Hardware

Window operating system is used for development of the AI interview analysis web portal. It is a personal computer operating system and is equipped with specification of Windows 11, 64-bit operating system, x64-based processor.

Specifications:

* AMD Ryzen 9 5900HS with Radeon Graphics 3.30 GHz
* 8GB DDR5 on board
* 8GB DDR5-4800 SO-DIMM
* NVIDIA® GeForce RTX™ 3070 Ti Laptop GPU
* Windows 11 Home

## Chapter 2 Summary

The chapter discuss the development and analysis of the NLP Interview Web Porta. First, the developer analyze the key attributes assessed during interviews, through a review of literatures and conclude that technical skills, communication skills, problem-solving skills, cultural fit, and professionalism are the attributes that are used widely. These attributes form the foundation for the portal's scoring system. For analysis to calculate these attributes’ score, Natural Language Processing (NLP) techniques is used to analyze insights from candidate responses, including techniques like keyword extraction, sentiment analysis, and semantic similarity. The chapter then discuss speech-to-text technologies. Google Cloud Speech-to-Text, IBM Watson Speech-to-Text, and Microsoft Azure Speech-to-Text is compared to assess their performance for transcription in this scenario. APIs implementation and work is discussed to enable integration of AI advanced features to the system. Then the developer did a comparative analysis of similar systems, including HireVue, Sapia.ai, and Interviewer.AI, analyzing their strengths and weaknesses. Then the technical framework of the portal is analyzed, using ReactJS for the frontend, Node.js as both the backend and web server, and MySQL for database management. The report also discuss their architecture, how it works to get a better understanding of the tech stacks. The NLP model implement Google Speech Recognition API for transcription, Scikit-learn for keyword extraction, and Hugging Face libraries for sentiment analysis and semantic similarity. Then for the prototype, the developer use Streamlit for rapid iteration development. This study help developer to research and get theoretical and technical knowledge on how the system should be built and how the development should be carried out.

Chapter 3

Methodology and Requirements Analysis

# Methodology and Requirements Analysis

This chapter will discuss on what development methodology is used for development aspect and analyze the requirement of client of the system.

## Agile Development Methodology

Agile is a widely used project management and development methodology designed to enhance flexibility, collaboration, and efficiency in software development and other domains. This methodology is suitable for NLP Interview Analysis Web Portal, due to the development being and Research and Development based project, which iterative development and agile practice for continuous improvement is important.

Agile is a set of principles and practices aimed at delivering improvements incrementally through iterative development. It emphasizes customer collaboration, enabling close interaction with stakeholders to ensure the product meets their requirements. Agile is also highly adaptable, suitable for evolving requirements, making it suitable for NLP Interview Analysis Web Portal’s innovative and dynamic characteristic. Additionally, its frequent delivery of small, functional increments allows for quicker feedback and adjustments. Agile development iterate around core elements like iterations or sprints, where work is divided into short cycles, each about 2-4 weeks, focusing on achieving specific goals and features.

Normally, it goes through multiple steps when designing the agile development methodology, including defining goals, team assembly, periodizing features, execute iterative development, maintain agile practices and iterating continuous improvement. The vision is firstly defined. Goals including analyzing interviews using NLP, scoring candidates' answers from multiple perspectives, generating performance marks, and providing detailed feedback. Following with team assembly, since this is a one-person full-stack project, we skip team assembly and proceed to create a product backlog that includes features like user registration, NLP functionalities, and deployment.

Then we prioritize features in collaboration with company representatives, starting with the basic architecture and login system, followed by core NLP modules and dashboard analytics. Then plan sprints, each lasting 2–4 weeks, to focus on specific functionalities. This will be further talked in the project planning section. Then iterative development is being executed. In between sprints, we enhance NLP functionalities like adding visual report to dashboard if it suits client’s needs. In final sprint, the required functionalities will mostly be complete and we can focus on improving performance like UI design and debugging.

Agile practice is maintained in this development process with weekly stand-ups to share progress, discuss challenges, and plan tasks. At the end of each sprint (2-4 weeks), present deliverables to stakeholders and gather feedback during sprint reviews. Then we will evaluate the sprint process, find out what went well and identifying areas for improvement. Continuous improvement is also maintained by implementing feedback for UI components based on usability test and stakeholder opinion. Carry out regular test components, workflows, and NLP model accuracy using unit and integration tests. Also release working versions incrementally at the end of each sprint, deploying updates iteratively to get feedback early.

## Core Target

No requirement gathering process is being carried out, since core requirement is defined and being instructed directly from GIT corporation. From requirement gathered from multiple meeting, the final system is described as a NLP system that can analyze candidate’s performance in interview. From candidate’s side, the candidate can read the question. Then they can record video themselves and upload to the system, then the video will be analyzed by the system and saved in database. From HR sides, they can select candidate and view candidate’s performance. Including their marks on each interview scoring attribute, like technical skills, communication skills, problem solving skills, cultural fit and professionalism. The system should be able to calculate all the scores, and give a final score for interviewer’s reference. It should also generate report according to report template given by GIT corporation.

## Requirement Analysis (Use Case Diagram)

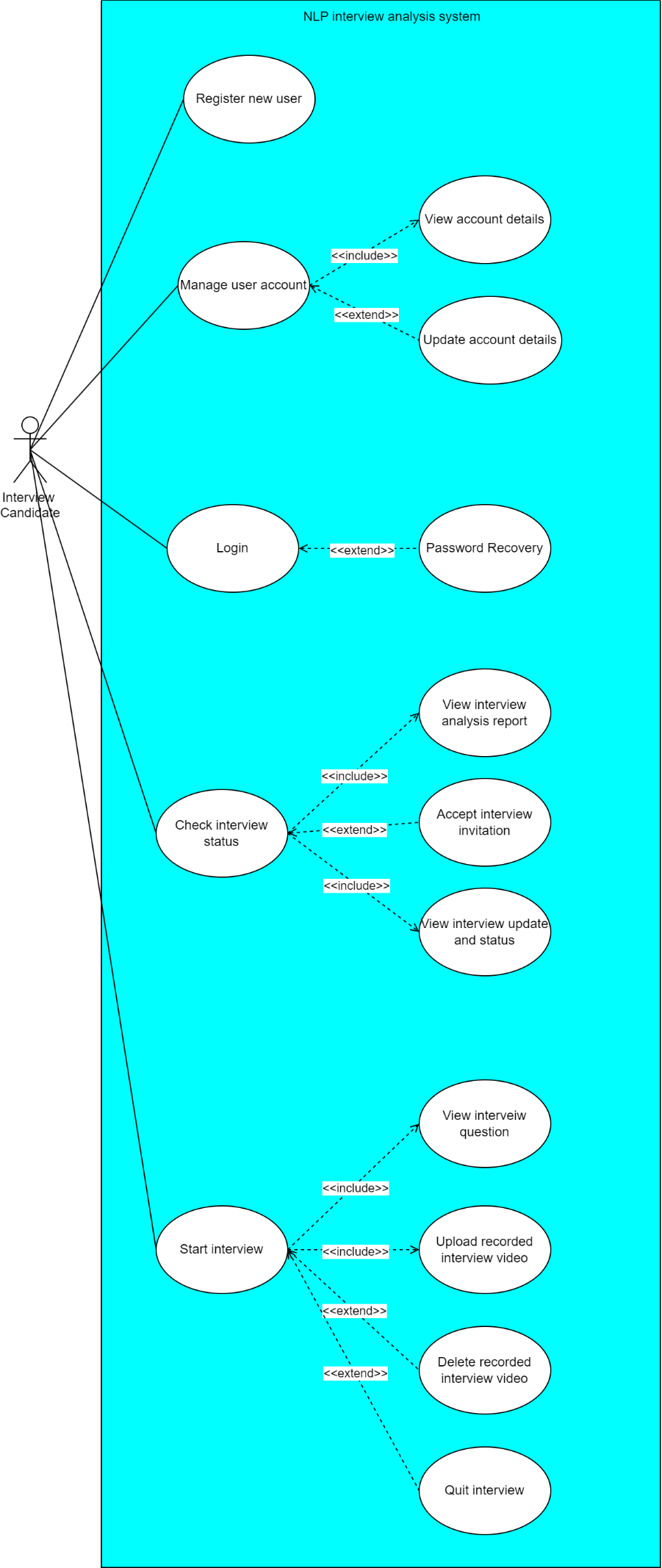


Diagram 3.1: Use case diagram of interview candidate

|  |  |  |
| --- | --- | --- |
| Use case name: | Register new user | |
| Actor: | Interview candidate | |
| Description: | Candidate create a new account, setting up username and password. | |
| Pre-condition: | - | |
|  | Actor action | System response |
| Main flow of events: | 1. Click register button | 1. Navigate user to a form to fill in username, password and details. |
| 1. Fill the user details, password | 1. Validate user input 2. If the information are valid, create account and navigate user back to login page |
| Alternative flows: | Step 4:  A1: if the password does not contain at least 1 symbol and 1 number, show error “password should include at least 1 symbol and 1 number” | |
| Post-condition: | Account created and saved into database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Manage user account | |
| Actor: | Interview candidate | |
| Description: | User able to see their account details if it is correct or not | |
| Pre-condition: | User must be logged in | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “manage user account” menu | 1. Navigate user to user details page |
| 1. Click “exit” | 1. Navigate user to main page |
| Alternative flows: | - | |
| Post-condition: | No change in system | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Update account details | |
| Actor: | Interview candidate and HR interviewer | |
| Description: | User update account details, including uploading resume. | |
| Pre-condition: | User must be logged in | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “manage user account” | 1. Navigate user to account profile details |
| 1. Click “edit account details” | 1. Show user editable page of account details |
| 1. Edit section that want to edit | 1. Validate if the new edit is valid 2. If it is valid, show “edit successful” and navigate user to account detail page |
| Alternative flows: | Step 6:  A1 if the new edit is invalid show error message “please check format” | |
| Post-condition: | User details updated and is displayed, details in database is updated. | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Login | |
| Actor: | Interview candidate and HR interviewer | |
| Description: | User is able to login into the system with user ID and correct password | |
| Pre-condition: | User must have registered their account | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “Login” | 1. Show login page |
| 1. Enter user ID and password | 1. Validate user ID and password 2. If the user ID and password is correct, navigate user to his account main page |
| Alternative flows: | Step 5:  A1 if the user ID and password is incorrect, show “incorrect user ID and password” | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Password recovery | |
| Actor: | Interview candidate and HR interviewer | |
| Description: | User is able to change his password by using link sent to his email address link | |
| Pre-condition: | User must have registered their account | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “Forget password” | 1. Send reset password link to users registered email account 2. Show “password reset link have been sent to your email” |
| 1. Click reset password email link | 1. Show form for user to enter new password |
| 1. Enter new password | 1. Validate new password is valid or not 2. If valid, show successfully reset password |
| Alternative flows: | Step 7:  A1 if the password is invalid, show “invalid password, please try again” | |
| Post-condition: | Password is updated in database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | View interview details and analysis | |
| Actor: | Interview candidate | |
| Description: | User is able to view interview completed and analysis of performance in interview | |
| Pre-condition: | User must have registered their account and at least 1 interview is done | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “Check interview status” in main page | 1. Navigate user to page showing their interview done |
| 1. Choose one and click the interview | 1. Show interview details and candidate’s performance in the interview |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Accept interview invitation | |
| Actor: | Interview candidate | |
| Description: | User is able to accept interview request from HR | |
| Pre-condition: | User must have registered their account and an interview request must be sent by HR | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “Check interview status” in main page | 1. Navigate user to page which include showing pending interview invitation |
| 1. Click interview invitation | 1. Ask user whether to accept or decline invitation. |
| 1. Accept invitation | 1. System display successfully joined interview. |
| Alternative flows: | Step 5:  If user decline invitation, display reject joining interview, notify in HR side that the candidate rejected the interview | |
| Post-condition: | In database, user is added to the interview, user can see that he is added to the interview. | |

|  |  |  |
| --- | --- | --- |
| Use case name: | View interview update and status | |
| Actor: | Interview candidate | |
| Description: | User is able to see notifications, updates and status of the interview | |
| Pre-condition: | User must have registered their account and have pending or completed interview | |
|  | Actor action | System response |
| Main flow of events: | 1. Click “Check interview status” in main page | 1. Navigate user to page which include showing interview update and status |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Start interview | |
| Actor: | Interview candidate | |
| Description: | User is able to read interview questions | |
| Pre-condition: | User must have registered their account and have been added to an interview | |
|  | Actor action | System response |
| Main flow of events: | View interview question | |
| 1. Click “Start interview” in main page | 1. Show confirmation message “Confirm to start interview?” |
| 1. Click “confirm” | 1. Navigate user to interview page and shows user interview questions |
| Upload recorded interview video | |
| 1. Record and upload interview video | 1. Validate uploaded file format 2. If format is correct proceed to show “video is uploaded” |
| 1. Complete questions and click submit | 1. Validate if all question is answered 2. If yes, show “interview completed, all files is submitted, pending for results” and send the videos to analysis NLP model |
| Alternative flows: | Step 4:  A1: If user click “No”, take user back to step 1  Step 7:  A1: If uploaded file is in wrong format, show “wrong format” go step 5 | |
| Post-condition: | The video is uploaded and saved to database, the interview status in user is changed to done interview, and user is not allowed to do the same interview again | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Delete uploaded interview video | |
| Actor: | Interview candidate | |
| Description: | User is able to delete interview video recorded | |
| Pre-condition: | -User must have registered their account  -Have been added to an interview  -Must have uploaded video | |
|  | Actor action | System response |
| Main flow of events: | View interview question | |
| 1. Click “Start interview” in main page | 1. Show confirmation message “Confirm to start interview?” |
| 1. Click “confirm” | 1. Navigate user to interview page and shows user interview questions |
| Delete uploaded interview video | |
| 1. Record and upload interview video | 1. Validate uploaded file format 2. If format is correct proceed to show “video is uploaded” |
| 1. Click delete video | 1. Show “Confirm to delete the video?” |
| 1. Click “confirm” | 1. Delete video and show “video deleted” |
| Alternative flows: | Step 4:  A1: If user click “No”, take user back to step 1  Step 7:  A1: If uploaded file is in wrong format, show “wrong format”  Step 11:  A1: If user click “cancel”, keep the video and go step 5 | |
| Post-condition: | Video is deleted in the interview page. | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Delete uploaded interview video | |
| Actor: | Interview candidate | |
| Description: | User is able to delete interview video recorded | |
| Pre-condition: | -User must have registered their account  -Have been added to an interview  -Must have uploaded video | |
|  | Actor action | System response |
| Main flow of events: | View interview question | |
| 1. Click “Start interview” in main page | 1. Show confirmation message “Confirm to start interview?” |
| 1. Click “confirm” | 1. Navigate user to interview page and shows user interview questions |
| Delete uploaded interview video | |
| 1. Record and upload interview video | 1. Validate uploaded file format 2. If format is correct proceed to show “video is uploaded” |
| 1. Click delete video | 1. Show “Confirm to delete the video?” |
| 1. Click “confirm” | 1. Delete video and show “video deleted” |
| Alternative flows: | Step 4:  A1: If user click “No”, take user back to step 1  Step 7:  A1: If uploaded file is in wrong format, show “wrong format”  Step 11:  A1: If user click “cancel”, keep the video and go step 5 | |
| Post-condition: | Video is deleted in the interview page. | |

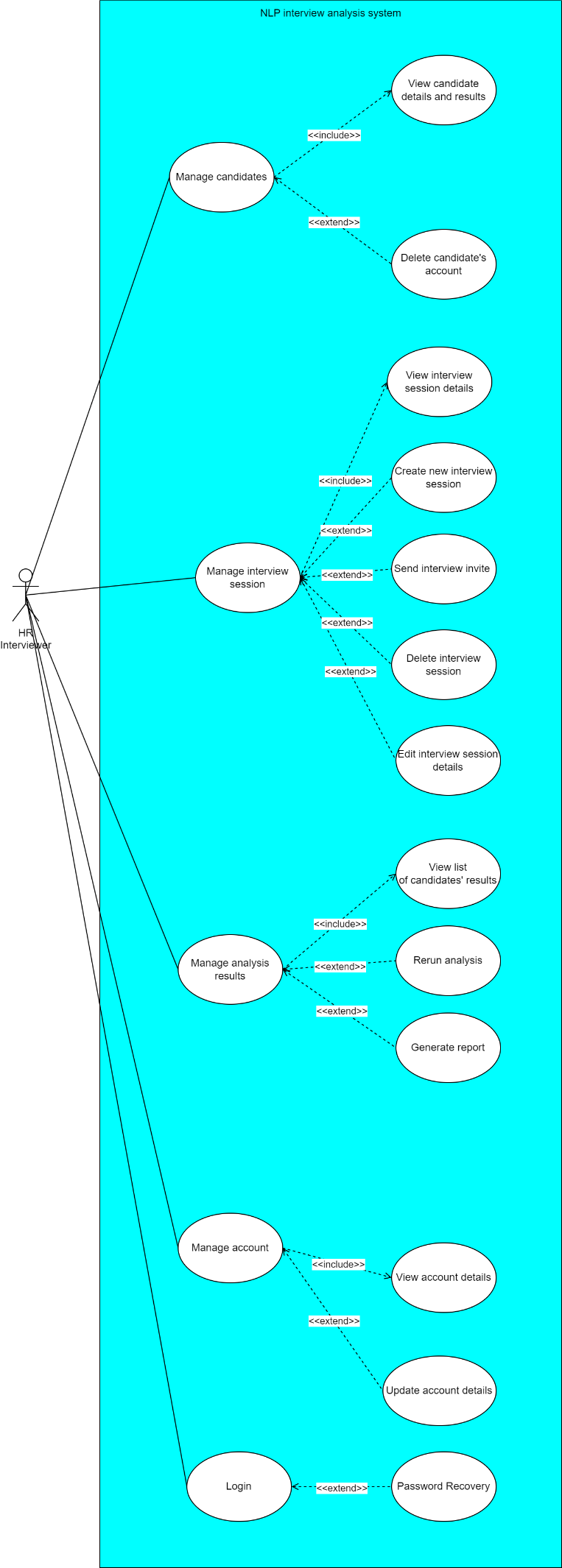


Diagram 3.2: Use case diagram of HR Interviewer

|  |  |  |
| --- | --- | --- |
| Use case name: | View candidate details and results | |
| Actor: | HR interviewer | |
| Description: | Can check candidate details like name, work experience, resumes and check their interview performance. | |
| Pre-condition: | User must be logged in in admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage user” | 1. Shows list of candidates |
| 1. Click on one candidate | 1. Shows selected candidate details and performance results |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Delete candidate’s account | |
| Actor: | HR interviewer and Admin | |
| Description: | Can delete candidate’s account. | |
| Pre-condition: | User must be logged in in admin role account  Candidate account must exists | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage user” | 1. Shows list of candidates |
| 1. Click on one candidate and select delete | 1. Show “confirm to delete?” |
| 1. Choose “Confirm” | 1. User deleted from database and shows “successfully deleted” |
| Alternative flows: | Step 6:  A1: If user choose “no” the popup is hidden and go back step 3 | |
| Post-condition: | User is deleted from the database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | View interview details | |
| Actor: | HR interviewer | |
| Description: | View specific interview details | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage interview” | 1. Shows list of interviews |
| 1. Click on one interview | 1. Show interview details |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Create new interview | |
| Actor: | HR interviewer | |
| Description: | Create a new interview and put questions in the interview | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage interview” | 1. Shows list of interviews |
| 1. Click on create interview | 1. Show “create interview” form |
| 1. Fill interview form and click submit | 1. Validate interview form 2. If valid, create new interview and add to database |
| Alternative flows: | Step 6:  A1: If information filled in interview form is invalid, show “invalid information, please try again” and go step 4 | |
| Post-condition: | New interview is added to database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Send interview invite | |
| Actor: | HR interviewer | |
| Description: | Send interview invite to candidates to join the interview | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “send invitation” | 1. Shows list of candidates |
| 1. Select candidates to invite | 1. Send invitations to selected candidates |
| Alternative flows: | - | |
| Post-condition: | Update candidate notification in database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Delete interview | |
| Actor: | HR interviewer | |
| Description: | Delete an interview session | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage interview” | 1. Shows list of interviews |
| 1. Select interview session and click on delete interview | 1. Deleted interview from database and show successfully deleted interview |
| Alternative flows: | - | |
| Post-condition: | The interview is deleted from the database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Edit interview details | |
| Actor: | HR interviewer | |
| Description: | Edit and update interview details such as questions, sessions | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage interview” | 1. Shows list of interviews |
| 1. Click on edit interview | 1. Show “edit interview” form |
| 1. Edit interview section and click submit | 1. Validate interview form 2. If valid, update new interview and add to database and go back main page |
| Alternative flows: | Step 6:  A1: If information filled in edit interview form is invalid, show “invalid information, please try again” and go step 4 | |
| Post-condition: | Interview details are edited and updated in database | |

|  |  |  |
| --- | --- | --- |
| Use case name: | View list of candidates’ results | |
| Actor: | HR interviewer | |
| Description: | View list of candidates result that is sorted by marks and job role | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “view results” | 1. Shows list of performance results |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Rerun analysis | |
| Actor: | HR interviewer | |
| Description: | Run the analysis again in case HR think the analysis result is incorrect | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “rerun analysis” | 1. Shows list of candidates |
| 1. Select candidate to rerun analysis | 1. Rerun analysis and generate analysis result, and update the result in database |
| Alternative flows: | - | |
| Post-condition: | The database of candidate performance and score is updated. | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Generate report | |
| Actor: | HR interviewer | |
| Description: | Generate a report of a candidate’s performance according to templates | |
| Pre-condition: | User must be logged in in HR admin role account | |
| Main flow of events: | Actor action | System response |
| 1. Click “generate report” | 1. Shows list of interview session |
| 1. Select an interview session and click generate report | 1. Generate all candidates’ performance report of the interview session |
| Alternative flows: | - | |
| Post-condition: | - | |

Note: Other repeated modules use case is included in Candidate’s side.

**Admin**

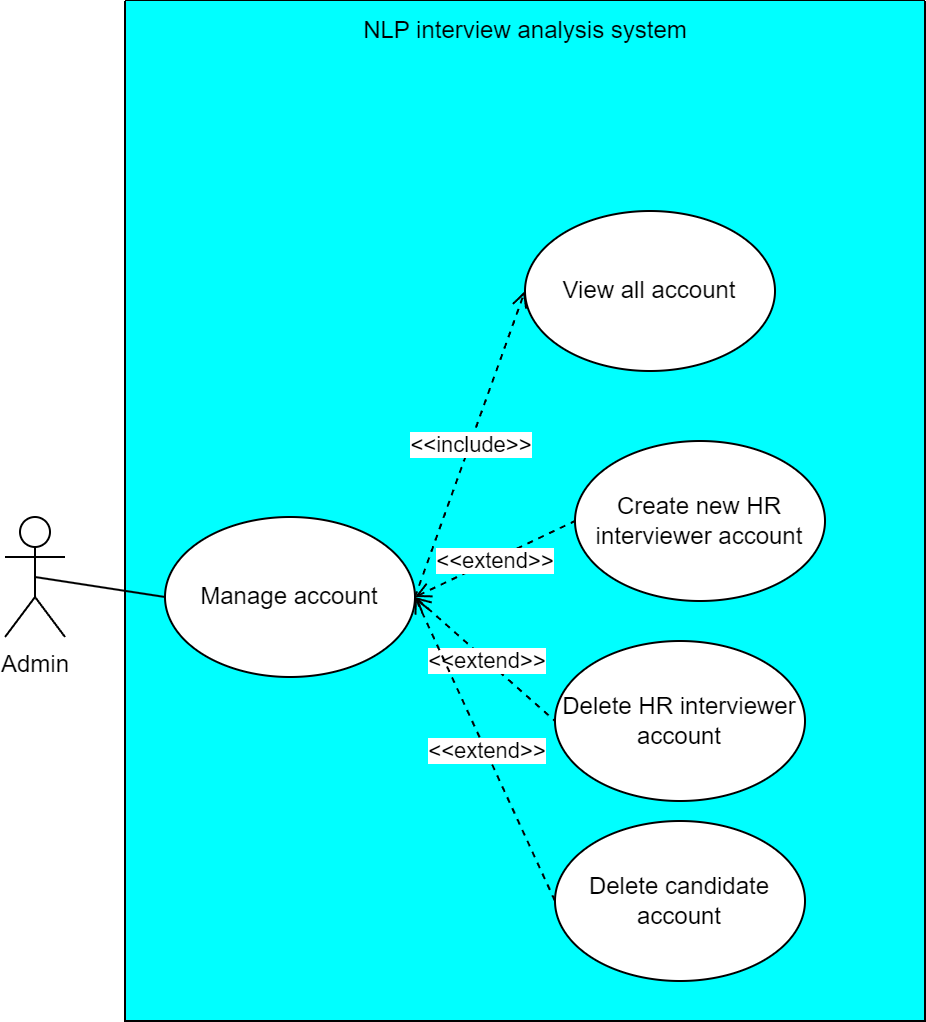


Diagram 3.3: Use case diagram of admin

|  |  |  |
| --- | --- | --- |
| Use case name: | View all account | |
| Actor: | Admin | |
| Description: | Generate list of account and click to see each account details | |
| Pre-condition: | User must be logged in as Admin | |
| Main flow of events: | Actor action | System response |
| 1. Click “manage users” | 1. Shows list of all users |
| Alternative flows: | - | |
| Post-condition: | - | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Create new HR interviewer account | |
| Actor: | Admin | |
| Description: | Generate list of account and click to see each account details | |
| Pre-condition: | User must be logged in as Admin | |
| Main flow of events: | Actor action | System response |
| 1. Click “create new HR account” | 1. Show “create account” form |
| 1. Complete the form and click submit | 1. Validate the form 2. If information entered is valid, create new account accordingly |
| Alternative flows: | Step 4:  A1: If information given is invalid, show “invalid information, please try again” and go back step 3 | |
| Post-condition: | New account is added and database updated. | |

|  |  |  |
| --- | --- | --- |
| Use case name: | Delete HR interviewer account | |
| Actor: | Admin | |
| Description: | Generate list of account and click to see each account details | |
| Pre-condition: | User must be logged in as Admin  HR interviewer account must exist | |
| Main flow of events: | Actor action | System response |
| 1. Click “delete HR account” | 1. Show list of HR account |
| 1. Select a HR interviewer account | 1. Show “confirm delete account?” |
| 1. Choose “confirm” | 1. HR interviewer account selected is deleted |
| Alternative flows: | Step 6:  A1: If user choose “cancel”, nothing changed and go back step 3. | |
| Post-condition: | Account is deleted and database is updated. | |

Note: Delete candidate account is included in HR recruiter side.

## Functional Requirement

### Interview candidates

1. The system should allow user to register new user
2. The system should allow user to view their account details
3. The system should allow user to update account details including edit name, upload resume and more
4. The system should allow user to login and logout
5. The system should allow user to let user to do password recovery in case user forgot their password
6. The system should allow user to view interview details and the performance
7. The system should allow user to accept interview invitations from HR of the company
8. The system should allow user to view interview update and status like HR change interview time or HR deleted the interview.
9. The system should allow user to start doing their interview, and the interview must have a time limit to complete once start.
10. The system should allow user to view interview questions
11. The system should allow user to upload recorded interview video according to questions
12. The system should allow user to delete uploaded interview video
13. The system should allow user to quit interview after they started the interview

### HR recruiters

1. The system should allow recruiters user to view candidate details and their results
2. The system should allow recruiters user to delete a candidate’s account
3. The system should allow recruiters user to view interview session details
4. The system should allow recruiters user to create new interview session
5. The system should allow recruiters user to send interview session invite to candidates
6. The system should allow recruiters user to delete interview session
7. The system should allow recruiters user to edit interview session details
8. The system should allow user to view list of candidates’ result
9. The system should allow recruiters user to rerun NLP analysis
10. The system should allow recruiters user to generate analysis report by selecting an interview session, then the system will generate all report under the interview session
11. The system should allow recruiters user to view their account details
12. The system should allow recruiters user to update account details such as edit name.
13. The system should allow recruiters user to login and logout
14. The system should allow recruiters user to let user to do password recovery in case user forgot their password

### Admin

1. The system should allow admin user to create new recruiter account
2. The system should allow admin user to delete account of any role

## Non-Functional Requirement

### Qualitative Attributes

Attributes are listed and prioritized according to its significance to the system.

**Functionality**

Functionality will be the most important attribute for an industrial project, since it is designed to be used by professionals to complete tasks efficiently. We must ensure that the system delivers on complete features and meets user needs effectively. Core capabilities like able to transcript multi language, scoring of interviews and supporting features such as generating reports and visualizing data must be implemented completely in its functionality and is able to generate result without failing. If the functionality is incomplete or unreliable, users will be unable to trust or rely on the system for decision-making, causing the whole system to be abandoned. As a result, developer need to implement thorough requirement analysis to ensure the portal meets all user needs. Moreover, carrying out regular test features to identify and fix bugs, and update the system to fulfill new functional requirements.

**Accuracy**

Accuracy is one of the important attributes for an interview analysis system. It ensures that the analysis and insights derived from interview data are precise and meaningful. For example, accurate transcription of interviews is crucial for further sentiment analysis, keyword extraction, or scoring. Similarly, the models used for evaluating performance must correctly interpret verbal and non-verbal cues, such as word choice, or pauses to generate reliable feedback. If the system's accuracy is not controlled, the insights it provides may mislead users, potentially resulting in poor hiring decisions, misjudgements of candidate potential, or a loss of trust in the portal. For example, inaccurate sentiment analysis might incorrectly label a candidate's response as negative due to a minor error in understanding their tone or choice of words, causing interviewer to misjudge candidate’s ability, leading to poor hiring decision. To achieve this attribute, it requires well-built data processing pipelines, high-quality training datasets, and regular refinement of NLP models. Additionally, the system should account for diverse linguistic features, such as accents, dialects, and cultural nuances, ensuring fairness and inclusivity across different candidates.

**Usability**

Usability is one of the most important attributes since it directly affect how easy for a user to interact with a system. HR professionals, recruiters and candidate often have varying levels of technical expertise, they may not be familiar with AI technical terms but familiar with human resource terms. For example, in showing semantic similarity, “Accuracy of answer” is better than “TF-IDF score”. A user interface that is designed surrounding usability requirement ensures that users can easily upload interviews, access analysis results, and interpret insights without requiring extensive training. System with poor usability may lead to frustration, errors, and wasted time, making the system unlikely to be well-used by human resources department. To achieve high usability, the system should have a user interface (UI) that follow design principles, like clear navigation, minimal clutter, and visual aids. Additionally, conducting user testing to identify pain points and refine the interface based on feedback can also be done.

**Learnability**

Learnability is important as it determines how quickly new users can understand and become proficient with the portal. Learnability is often follow with usability, since both attribute are normally achieved by a good UI design. Organizations may have large teams who need to adapt to the system without spending excessive time on training. A platform with high learnability offers clear instructions, minimal complexity, and intuitive workflows, enabling users to start using its features effectively after deployed. For example, simple navigation, helpful tooltips, and easy-to read help section can make a big difference in user experience. When users feel confident in their ability to use the system quickly, it boosts their productivity and ensures a smoother integration into their daily tasks. To achieve high learnability, onboarding guides, tutorials, and tooltips must be provided. Moreover, developer can simplify workflows by ensuring actions follow logical steps, making the system easier to understand for new users.

### Organizational Requirement

**Implementation Requirements**

GIT specify the system should be web based. However, the tech stack is not limited. Since this project is a collaborative project, the developers use the same tech stack for AI model development, which is python as language and Jupyter Notebook as code editor and computing platform.

**Report Standards Requirements**

Report template is specify by GIT corporation. According to report template, the final output of the report that is required is to design the system to calculate mark for each interview attributes, then calculate a final score for the candidates’ interview performance. The report must show each interview attributes and its marks and how the final marks is calculated. The system analysis will be designed based on the final report to be generated.

### Limitations and Constraints

**Time constraint**

The whole project has one-year timeline, which limits the scope of what can be achieved. Creating and fine-tuning AI models to achieve the highest accuracy is time-intensive, the focus must be on developing a functional MVP (Minimum Viable Product) with moderate accuracy. We have to prioritize essential features on working on MVP ensures the project progresses steadily.

**Budget constraint**

With no financial resources or sponsors, all tools, frameworks, and services must fall under free or open-source tiers. This restricts access to high-quality, paid datasets or premium AI APIs, which can improve accuracy. Instead, the project must rely on free datasets and pre-trained models that offer reasonable performance. While this saves costs, it also limits the accuracy of the AI features.

**Resource constraint**

As a one-person developer working full-stack, the workload is huge, leaving limited effort to optimize AI models to the highest accuracy. Prioritizing tasks, manage efficiency, balancing development, testing, and deployment tasks is important under these constraints. Pre-trained models or AI libraries with minimal customization can help achieve functional results without overwhelming the developer. Tech stacks with simple, large community are also chosen to ensure efficiency. Advanced AI capabilities and features may need to be postponed for further development.

## Chapter Summary and Evaluation

This chapter discuss the development process and requirements analysis for the NLP Interview Web Portal. Agile development methodology was implemented for its flexibility and iterative approach, allowing the developer to adapt to changes and change based on feedback effectively. This industrial collaboration with GIT Corporation did not involve a requirement collection phase since the core target is clearly defined by GIT, which is to automate the interview process and generate reports. Use case diagrams was used to visualize and clearly analyze the system's requirements. Functional requirements were listed based on user, including login, delete user, generate report and more. Non-functional requirements like system performance, scalability, and usability is also analyzed and listed. Additionally, limitations and constraints were also analyzed, including time constraints, budget limitations, and resource constraints. From discussing and researching on these sections, developer have a better concept of what requirements should be achieved, how the development will be carried out, and what to be caution on when assigning resources on developing the system.

Chapter 4

System Design

# System Design

## System Structure

### Object-Oriented Approach

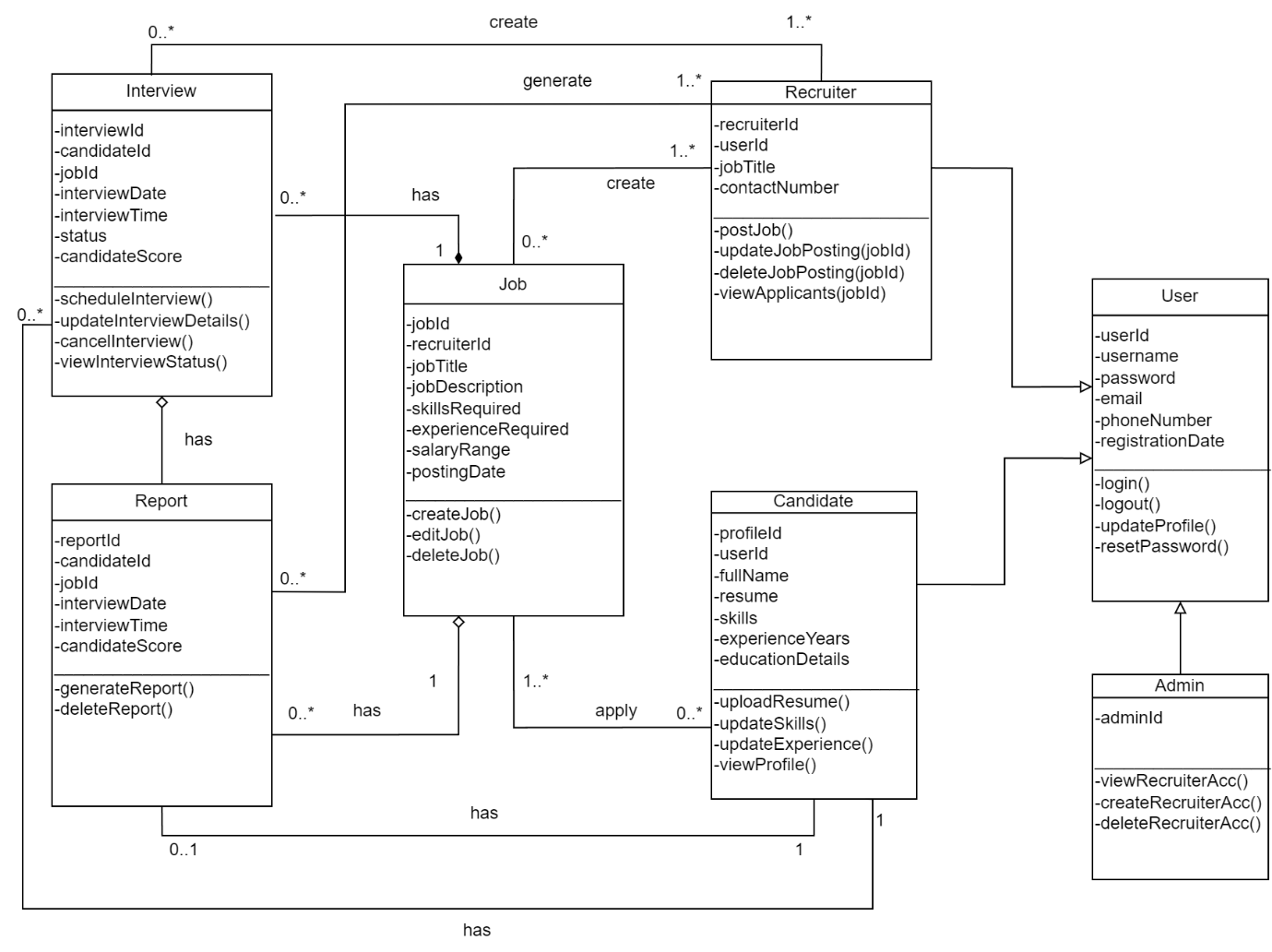


Diagram 4.1: Class Diagram

Referring to diagram, there are 3 types of users: Admin, Candidate and Recruiter. Each type of users has their own functionality, but they share the same attributes and function of user class, since all of them have same feature to login and check account details. Those are saved as a parent class to inherit to all the three users. For job, it will include the details for job which are posted, this job class. Each job will have multiple interviews and the interviews can be categorized by job to collect as one, so the job class also act like an interview session that handle the interviews. Each interview is able to generate 1 report and each candidate can have zero or many reports, these 2 aspects is important for recruiters to view report results in actual implementation.

## Process Design

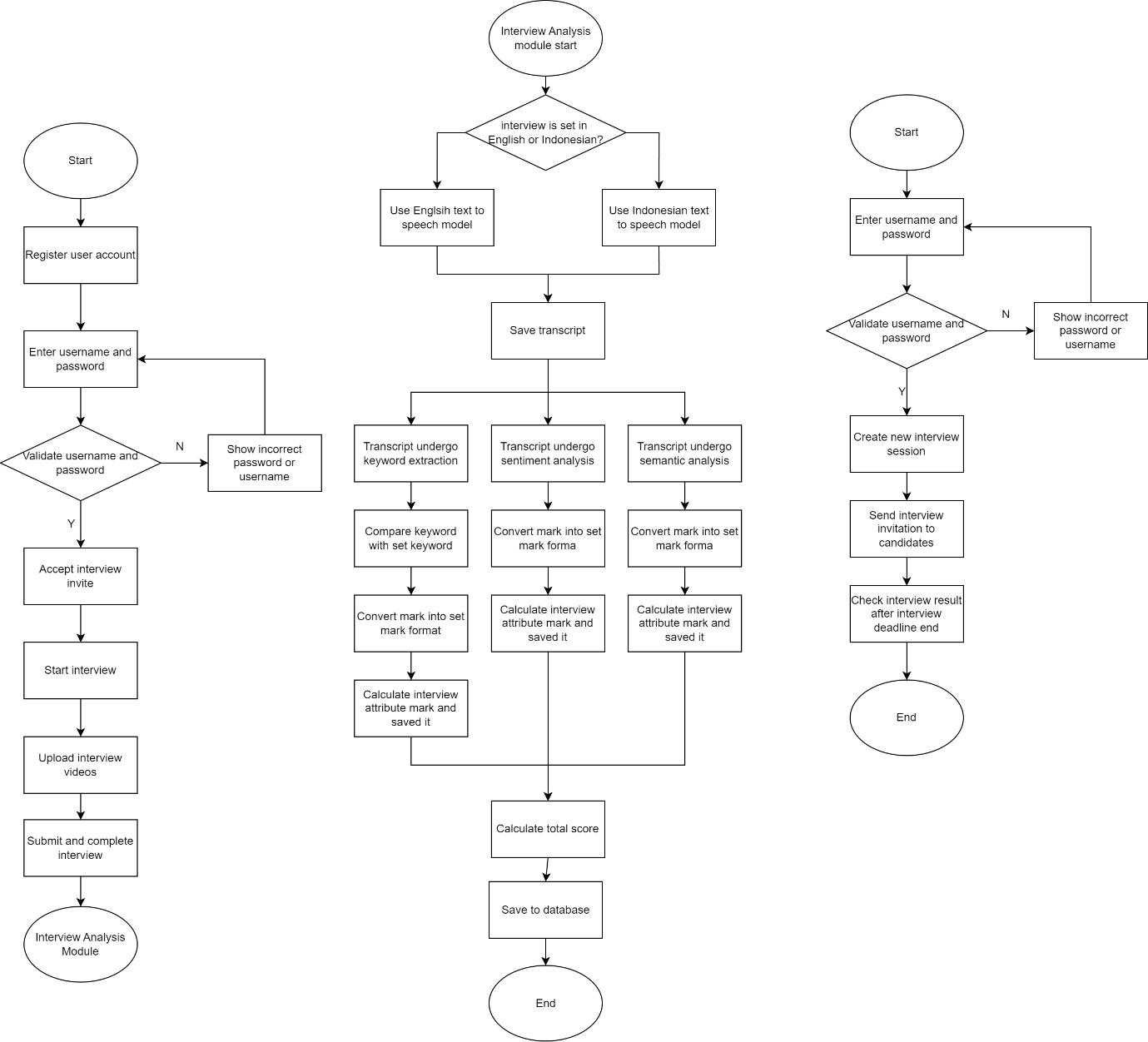


Diagram 4.2: Flowchart for main process of system

The figure shows the main flow of the system, one from candidate’s side and another from recruiter’s side. For candidate, they will start with registering an account and login into the account with passwords. Then they will need to accept interview invitation sent by the recruiter. After that, they will be able to start the interview. In the interview, candidate are required to record and upload interview videos in a limited time. Then the video will be submitted for analysis.

For analysis side, the system will check if the video is in Indonesian or English. Since we will need different module for speech to text to translate speech to text to translate different language. After the transcript is generated and saved, it will go through keyword extraction, sentiment analysis and semantic analysis for analysis and scoring. Each score is calculated and saved for total score calculation; the system will use formula to calculate total score according to weightage of each interview attributes. Finally, the scores and total score is saved to database to be later generated as report and show on candidates’ result page.

For admin side, the main flow of recruiter’s task is to create interview session and send invitation to candidates and check interview results. The process started as recruiter login into the system. Then creating a new interview session and send interview invitation to candidates and wait for results. After interview deadline, recruiter will check on candidates scoring which will be sorted accordingly by system to make hiring decisions for best talents.

## Software Architecture Design

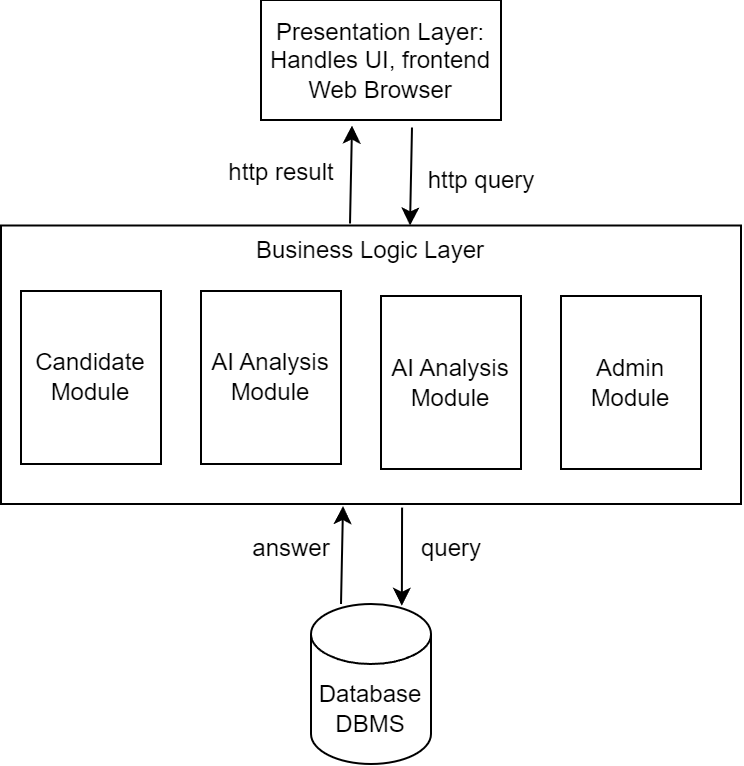


Diagram 4.3: Software architecture

Three-tier architecture is architecture is widely used in web application design due to its clear separation of concerns, maintainability, and flexibility for future growth. Three-tier architecture divides the system into three distinct layers including presentation layer, business logic layer and data layer. Each layer is responsible for a specific aspect of the application, ensuring a modular design (insightsoftware, 2024). For this project, the presentation layer will handle the user-facing interface, allowing candidates and recruiters to interact seamlessly. The business logic layer will manage processes like interview scheduling, feedback collection, and report generation. While data layer will efficiently store and retrieve information such as user profiles, jobs, and interview details. This architecture supports future scalability as the system grows. For example, if web portal has increase users or needs advanced features like analytics, each layer can be rescaled independently without need to redesign. Moreover, the three-tier structure achieve maintainability by allowing independent updates or replacements of layers. For instance, switching from ReactJS to another frontend framework can be done without affecting the backend logic or database. This is particular useful for this research and development project since developer are keep trying multiple technologies to achieve tasks. It is also useful for beneficial for a single-developer project, as it reduces the overall complexity of updates and feature additions.

## Algorithm Design

This section analyzes how modules algorithm is design, the developer only analyzes relevant modules that are complex and skip simple modules.

### Start Interview Activity Diagram

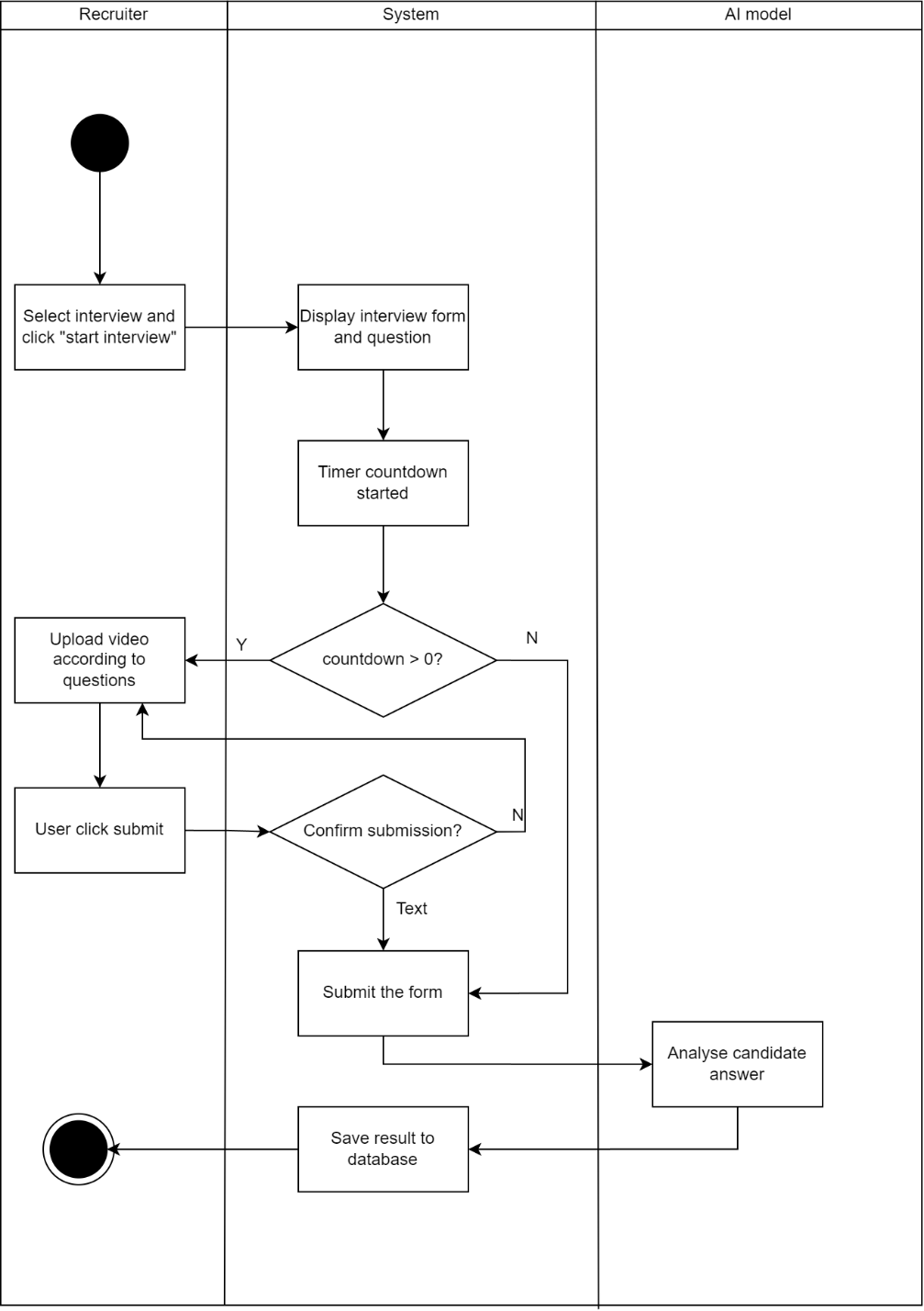


Figure 4.4: Start Interview Activity Diagram

The process is break down as diagram, the user start the interview and is able to view the questions and answers the form. The countdown timer is started as user started, they will have limited time to complete the form. If the timer reached 0, the submission time is up and system will automatically upload the form either user have completed it or not. After the form is submitted, recorded videos are sent to AI models for analysis and scoring for report generation. The scoring is then saved and the module ends here.

### Invite Candidate Activity Diagram

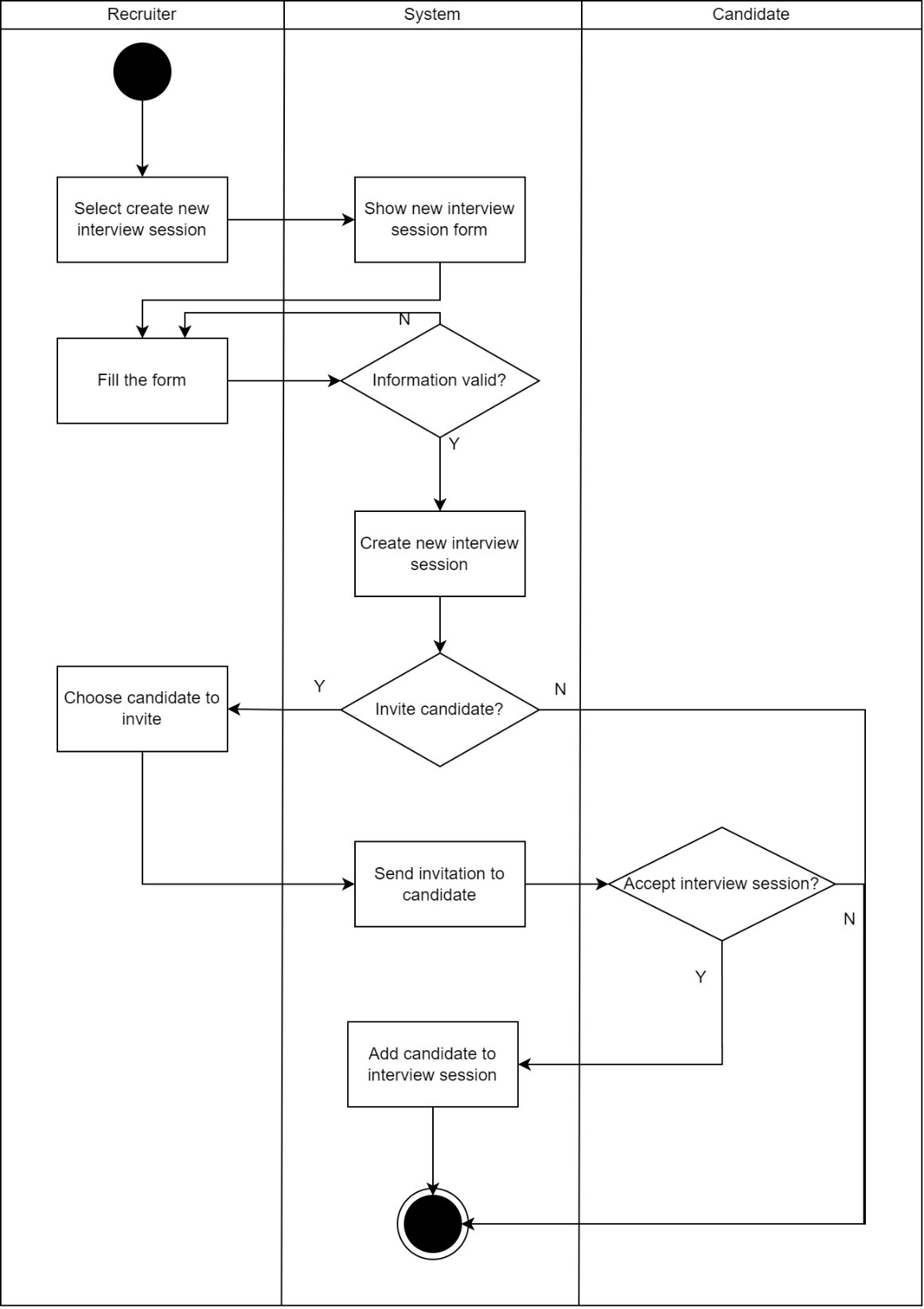


Figure 4.5: Activity diagram for invite candidate module

The recruiter will start with creating an interview session if there is no existing interview session. Then they will fill the details and question for the interview session. After validating, the session will be created. Then the system will prompt a question to ask whether recruiter want to invite candidate now, they can also invite later. If they choose to invite, the system will give recruiter a list of candidate to invite then send invitations to candidate. On candidate side, they will see an invitation in their notification bar and if they accept the invitation they will be added to the job interview session and complete their interview. Otherwise, they will not be added. This is design as invitation since the system would like to inform candidate via invitation and candidate can reject the interview if they make their decision to not join the interview.

## Data Design

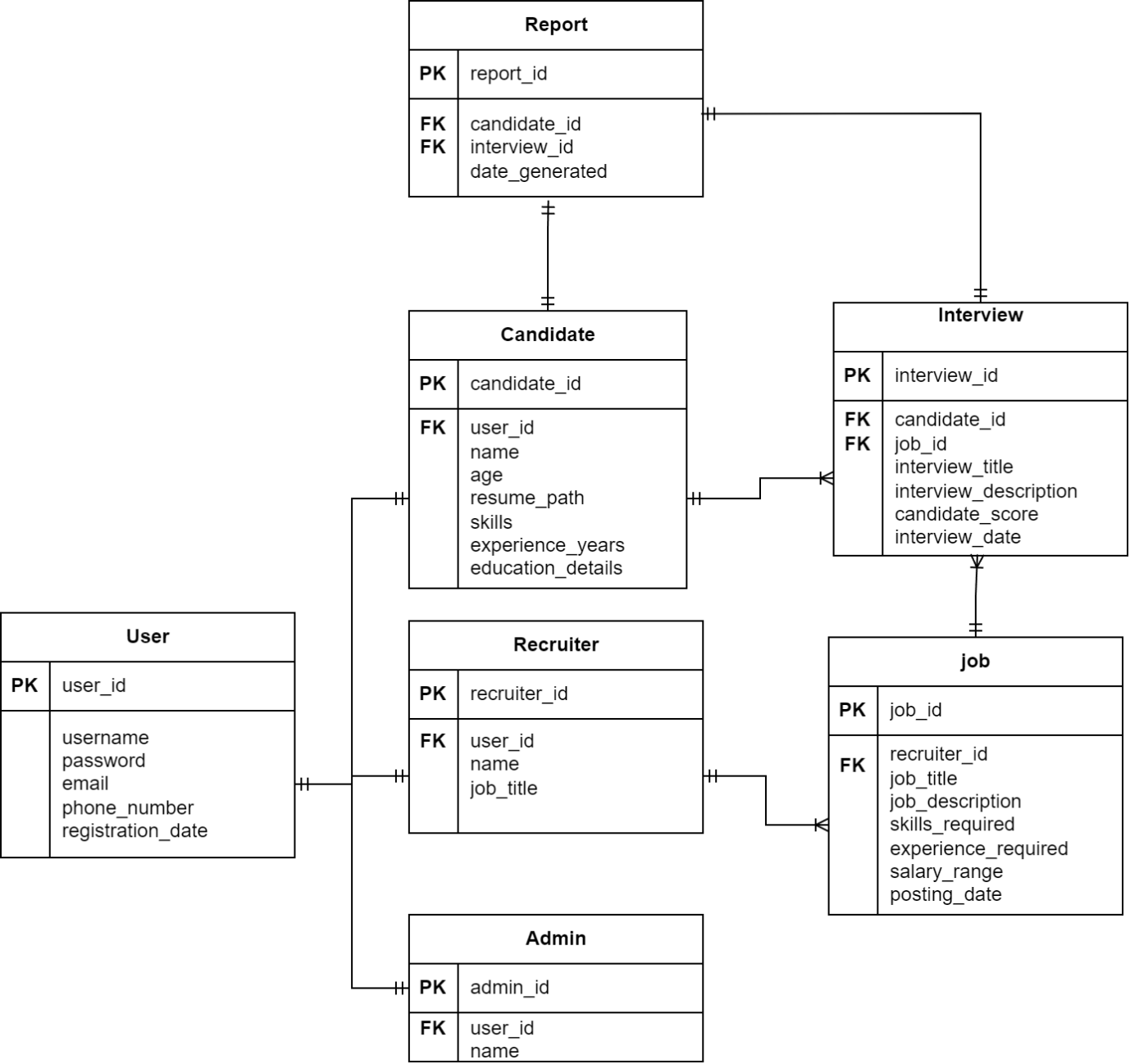


Figure 4.6: Database Relationship Diagram

From the diagram, we use the same inheritance relationship for three type of users, admin, recruiter and candidates. User parent table will store the sharing information of these three entities, including username, password, email and more for login purpose and account management modules. They are one-to-one relationship and they will have user foreign key to connect with user table. Job table act as an interview session to store all interviews entities, so they will form a one-to-many relationship with interview entity. Moreover, it will need recruiter id as foreign key to connect with recruiter table for feature like giving only recruiter with specific interview session to make changes. Then the interview table will have connection with job and candidate table so the scoring module can access the performance score via this connection. Finally report table will have connection with interview table and candidate table.

## Data Dictionary Table

Table Name: User

Table Description: To store user information and passwords for login purpose

Primary Key: user\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | user\_id | INT | Primary Key | To store user id |
| 2 | username | VARCHAR(50) | Not null | To store user name |
| 3 | password | VARCHAR(255) | Not null | To store user password |
| 4 | email | VARCHAR(255) | Not null | To store user email |
| 5 | phone\_number | VARCHAR(15) | Not null | To store user phone number |
| 6 | Registration\_date | DATETIME | Not null | To store registration date |

Table Name: Admin

Table Description: To store admin information

Primary Key: admin\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | admin\_id | INT | Primary Key | To store admin id |
| 2 | user\_id | VARCHAR(50) | Foreign Key | To connect user id |

Table Name: Recruiter

Table Description: To store recruiter information

Primary Key: recruiter\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | recruiter\_id | INT | Primary Key | To store recruiter id |
| 2 | user\_id | VARCHAR(50) | Foreign Key | To connect user id |
| 3 | name | VARCHAR(50) | Not null | To store recruiter name |
| 4 | job\_title | VARCHAR(50) | Not null | To store recruiter job title |

Table Name: Candidate

Table Description: To store candidate information

Primary Key: candidate\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | candidate\_id | INT | Primary Key | To store candidate id |
| 2 | user\_id | VARCHAR(50) | Foreign Key | To connect user table |
| 3 | name | VARCHAR(50) | Not null | To store candidate full name |
| 4 | age | INT | Not null | To store candidate age |
| 5 | resume\_path | VARCHAR(255) | - | To store url link to resume |
| 6 | skills | VARCHAR(50) | Not null | To store candidate skill |
| 7 | experience\_years | VARCHAR(255) | - | To store candidate working experience details |
| 8 | Education\_details | VARCHAR(255) | - | To store candidate education details |

Table Name: Job\_id

Table Description: To store job details for interview session

Primary Key: job\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | job\_id | INT | Primary Key | To store job id |
| 2 | recruiter\_id | INT | Foreign Key | To connect recruiter table |
| 3 | job\_title | VARCHAR(50) | Not null | To store job title |
| 4 | job\_description | VARCHAR(255) | Not null | To store job description |
| 5 | skills\_required | INT | Not null | To store skills required to apply for job |
| 6 | experience\_required | VARCHAR(255) | - | To store experience required to apply for job |
| 7 | salary\_range | VARCHAR(50) | Not null | To store salary range of job |
| 8 | posting\_date | DATETIME | - | To store date job posting posted |

Table Name: Interview

Table Description: To store interview for single candidate performance and details

Primary Key: interview\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | interview\_id | INT | Primary Key | To store interview id |
| 2 | candidate\_id | INT | Foreign Key | To connect candidate table |
| 3 | job\_id | INT | Foreign Key | To connect job table |
| 4 | interview\_title | VARCHAR(50) | Not null | To store interview title |
| 5 | interview\_description | VARCHAR(255) | Not null | To store interview description |
| 6 | candidate\_score | VARCHAR(50) | - | To store candidate performance score in interview |
| 7 | interview\_date | DATETIME | Not null | To store interview date and time |

Table Name: Report

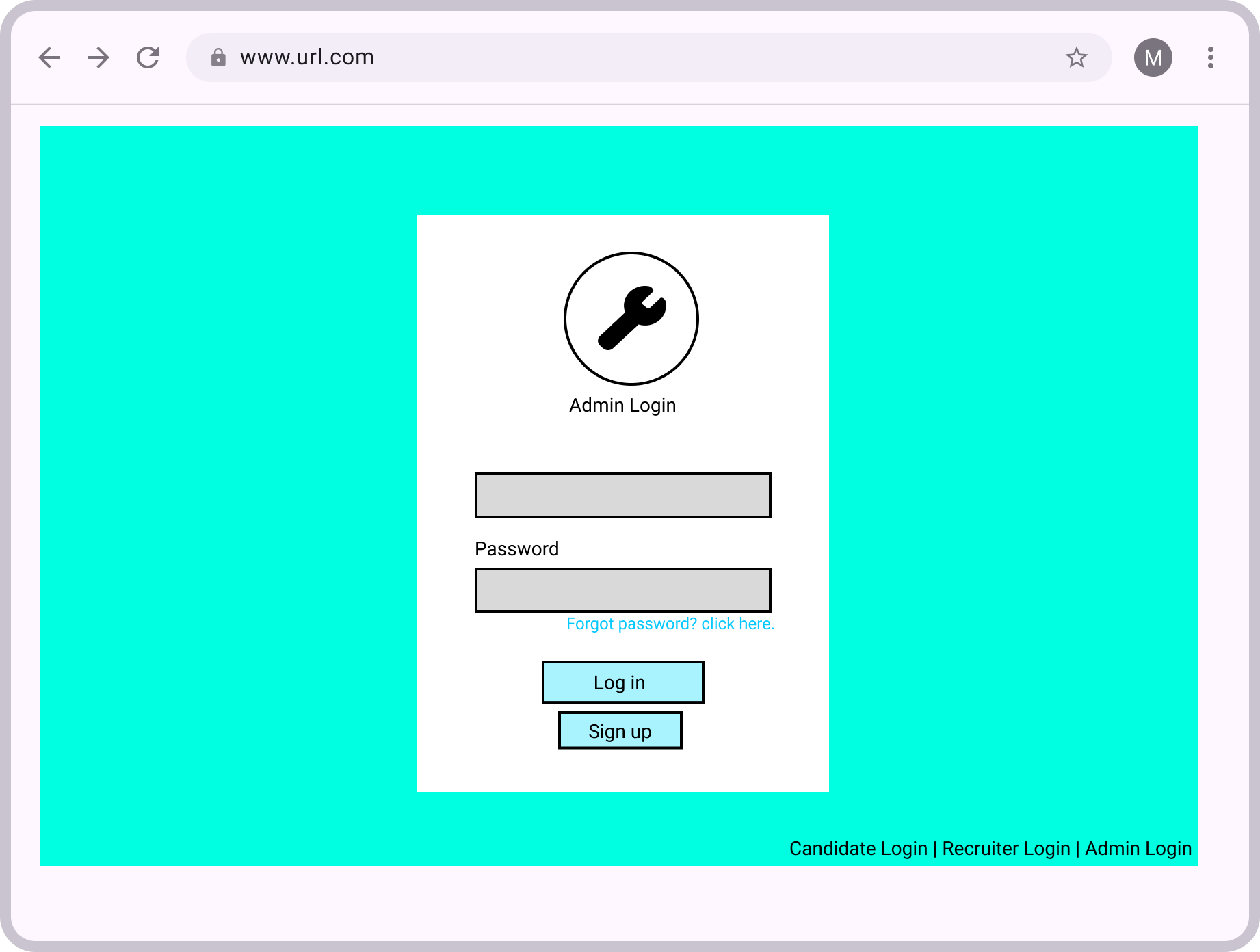
Table Description: To store interview report information

Primary Key: report\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Field name | Data type | Constraint | Description |
| 1 | report\_id | INT | Primary Key | To store report id |
| 2 | candidate\_id | INT | Foreign Key | To connect candidate table |
| 3 | interview\_id | INT | Foreign Key | To connect interview table |
| 4 | date\_generated | DATETIME | Not null | To store report date generated |

## UI Design

**Admin Module**



Email

Figure 4.7: Login page

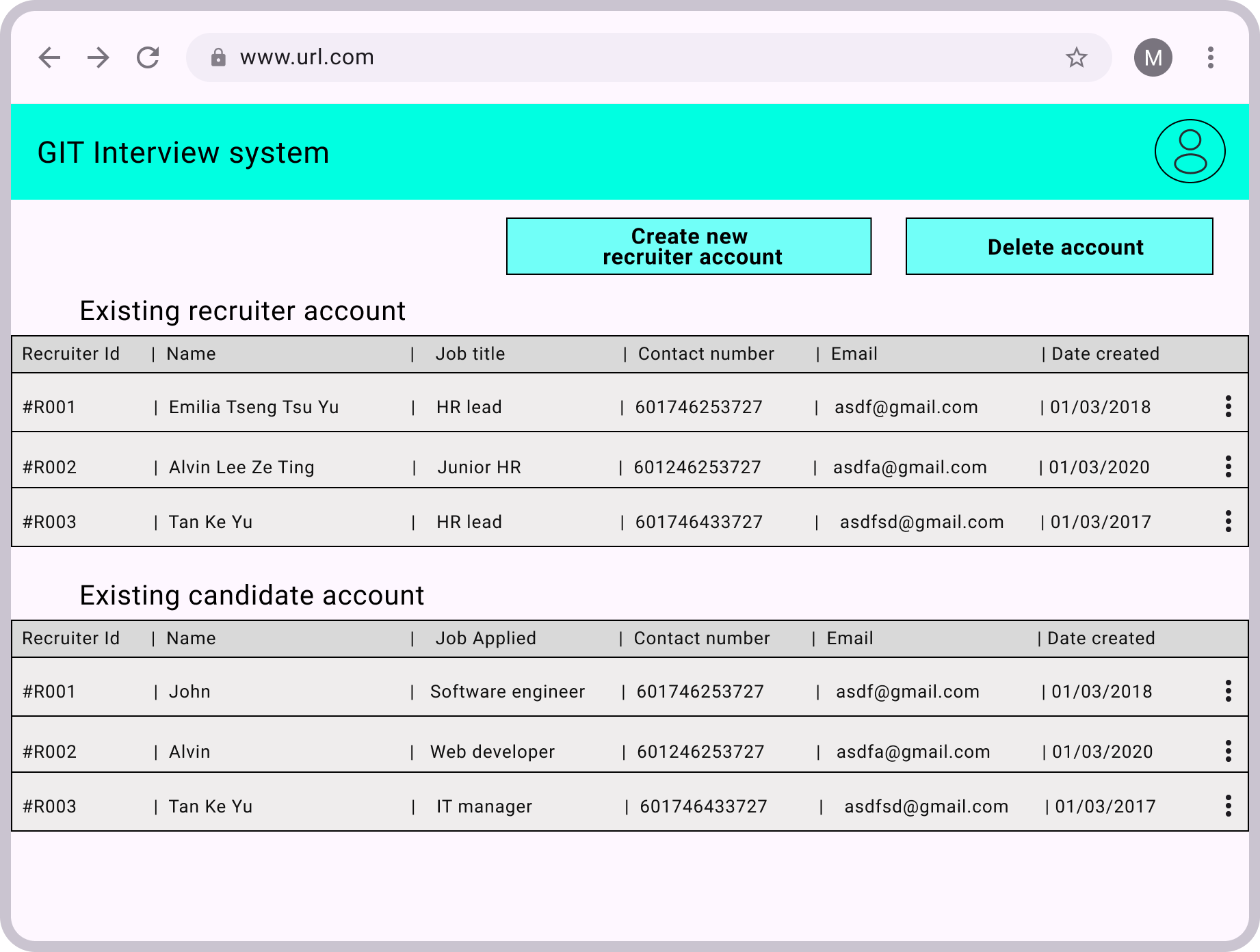


Figure 4.8: Admin manage account page



Figure 4.9: Admin manage account page

This page is for admin login, typing admin email and password to login, the admin account has highest access to manage accounts.

**Candidate Module**

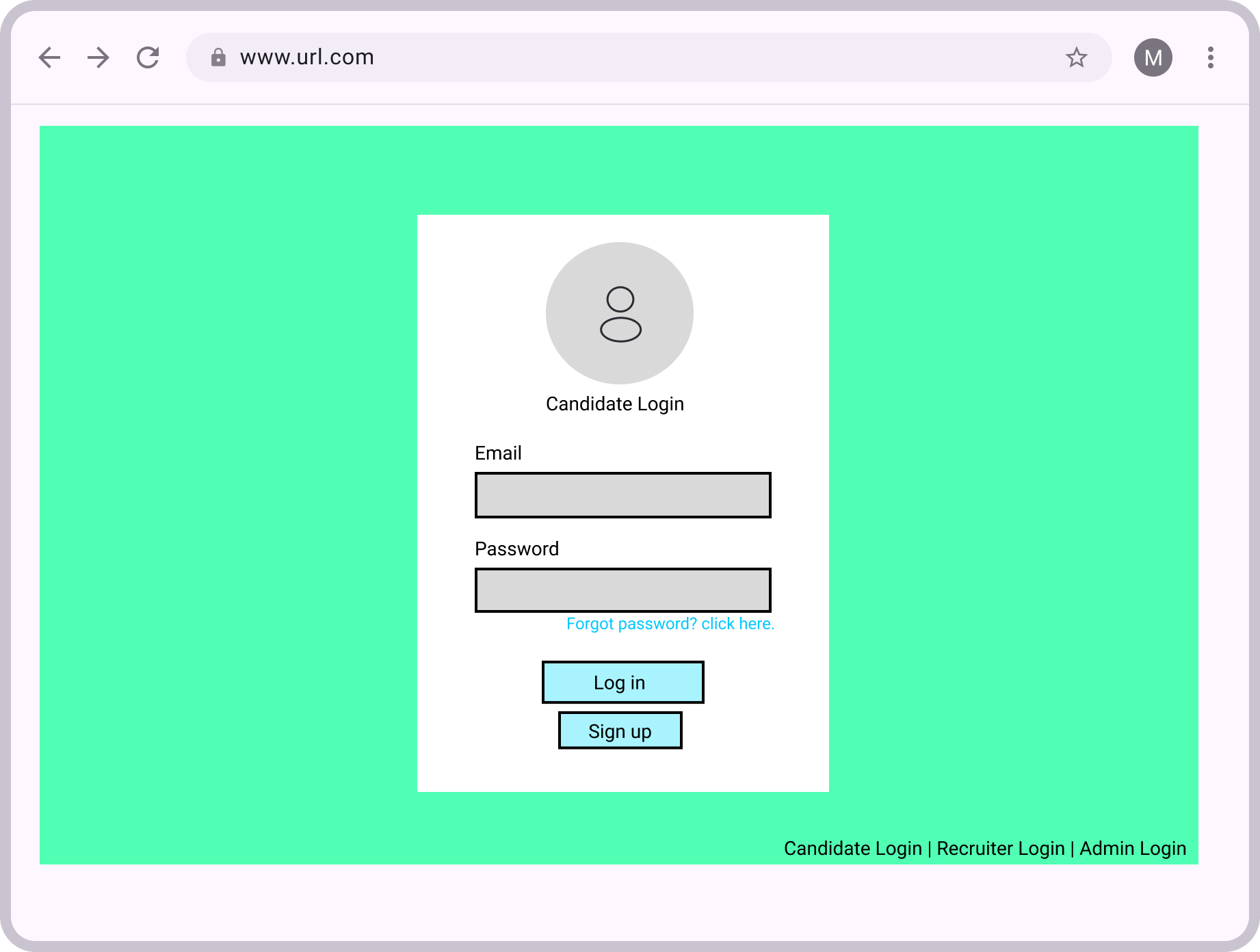


Figure 4.10: Candidate login page

Login page for candidate via email or password. User can choose to access other login page in bottom right corner.

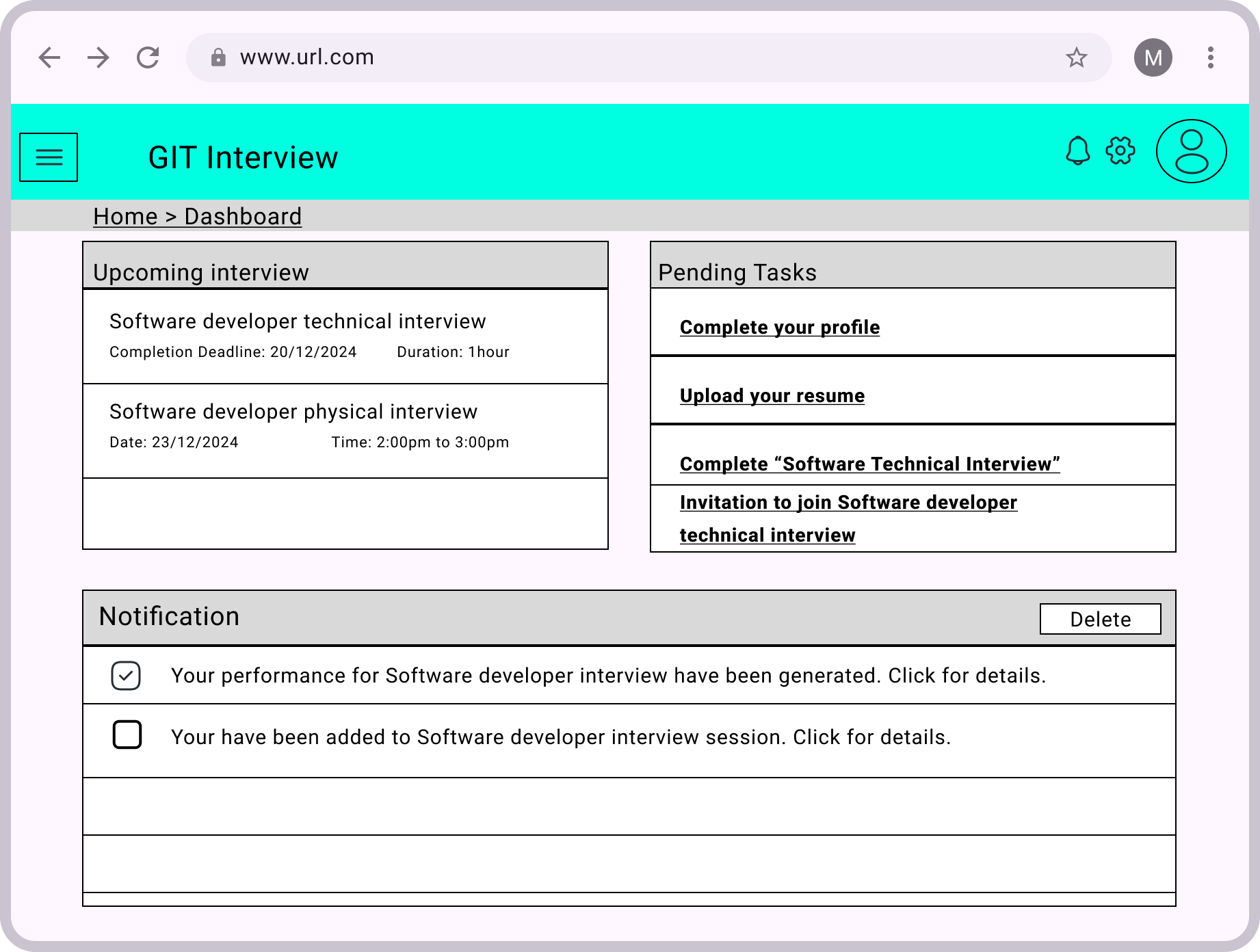


Figure 4.11: Dashboard page

In dashboard, upcoming interview is displayed too candidate to track their interviews, it also show pending task which require user to complete for interview evaluation. There is also notification to inform update for their interview.

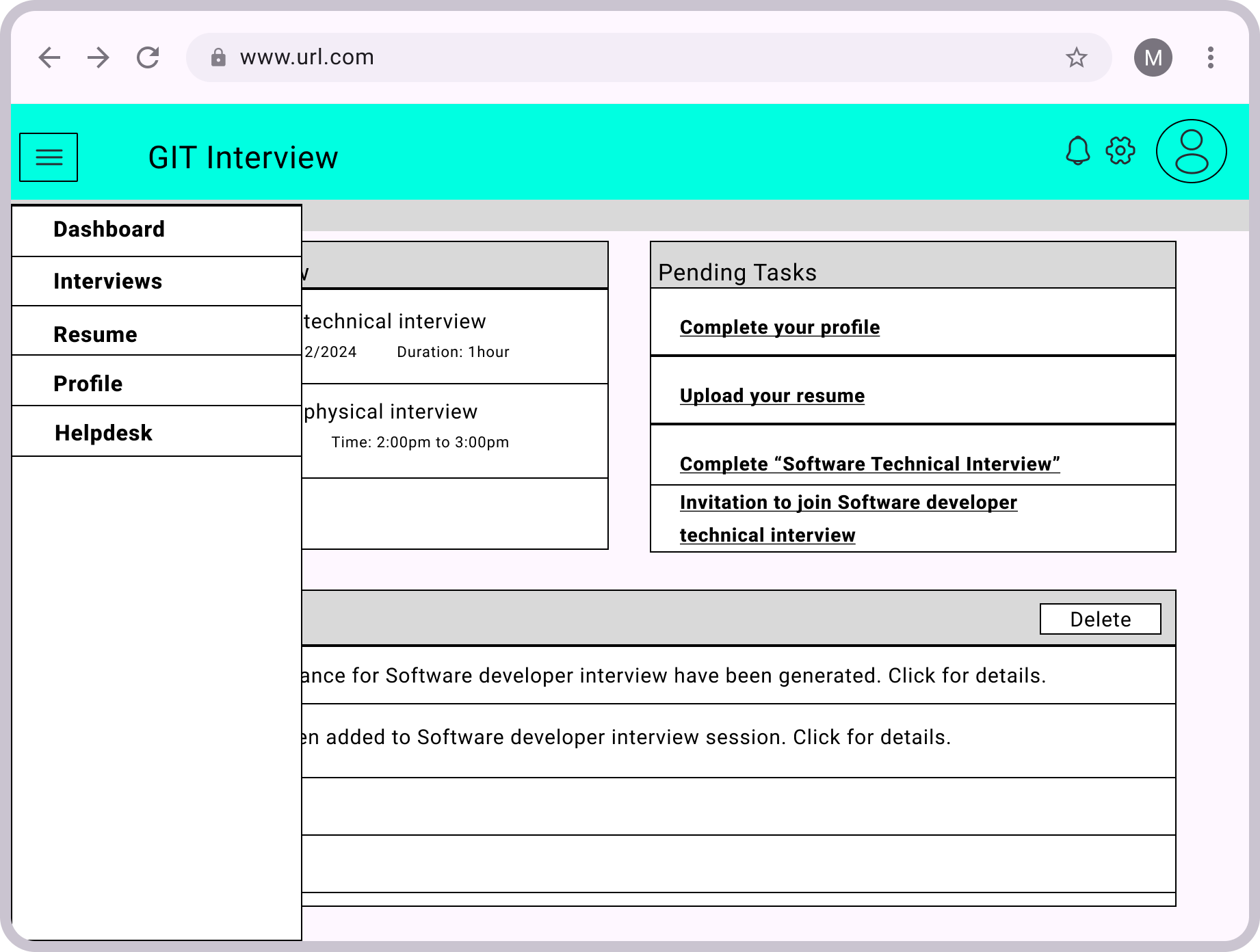
****

Figure 4.12: Navigation bar

Navigation menu is designed at the left side for easy access.

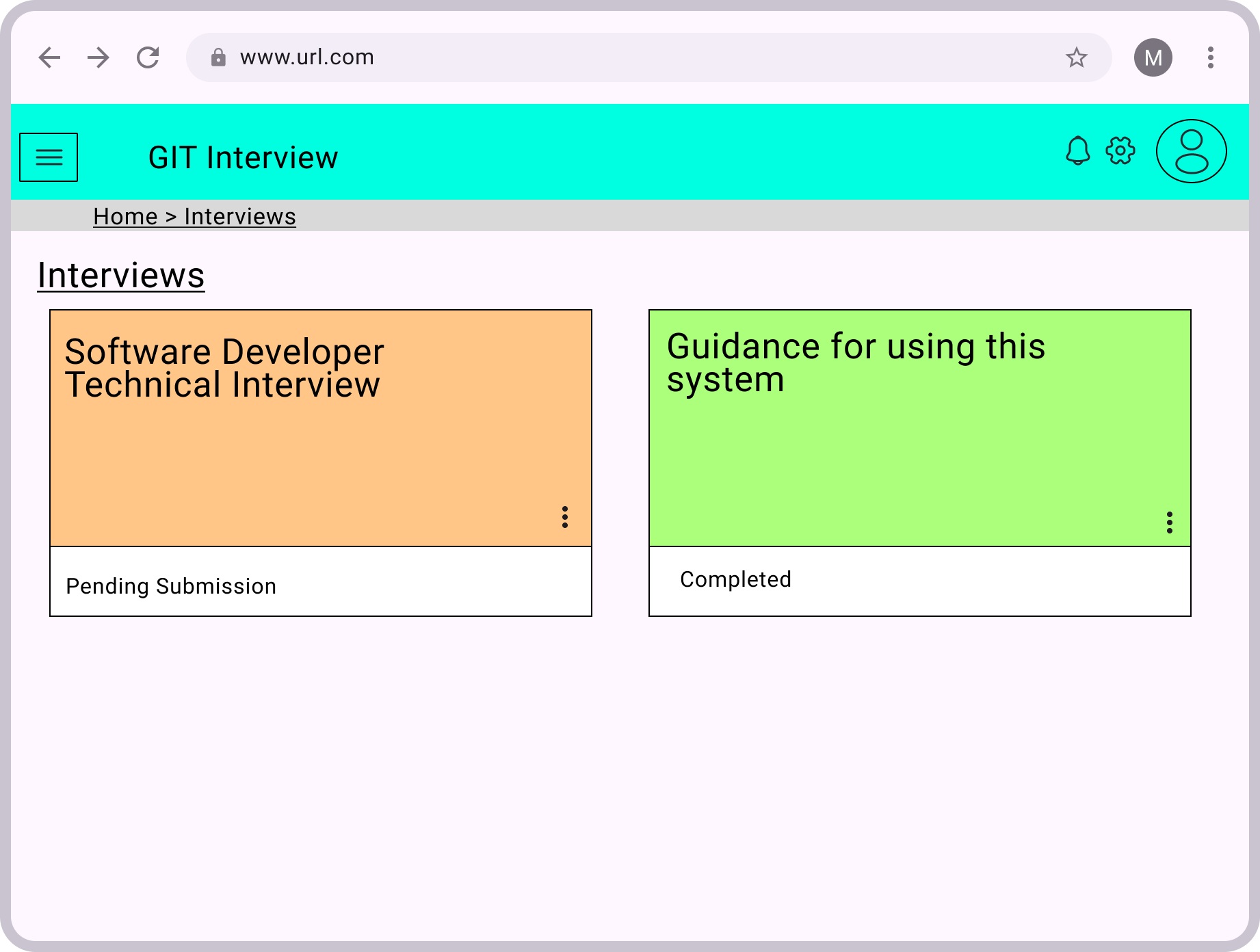
****

Figure 4.13: Interview session page

These are interview session that are joined by user, the system will include a default guidance session as tutorial to guide user how to use the system and what to prepare for the interview.

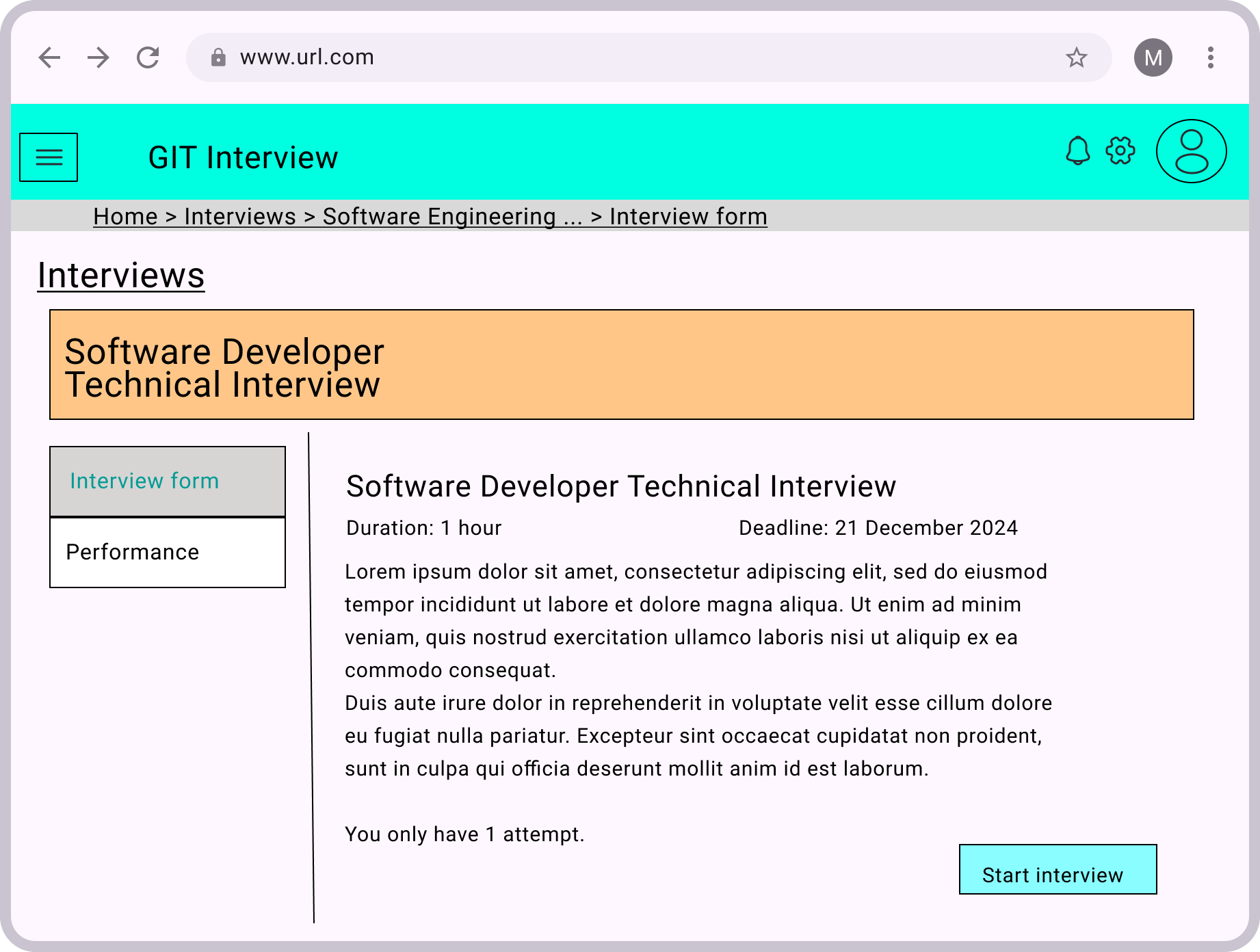
****

Figure 4.14: Interview form page

System will show user instructions to prepare and interview rules.

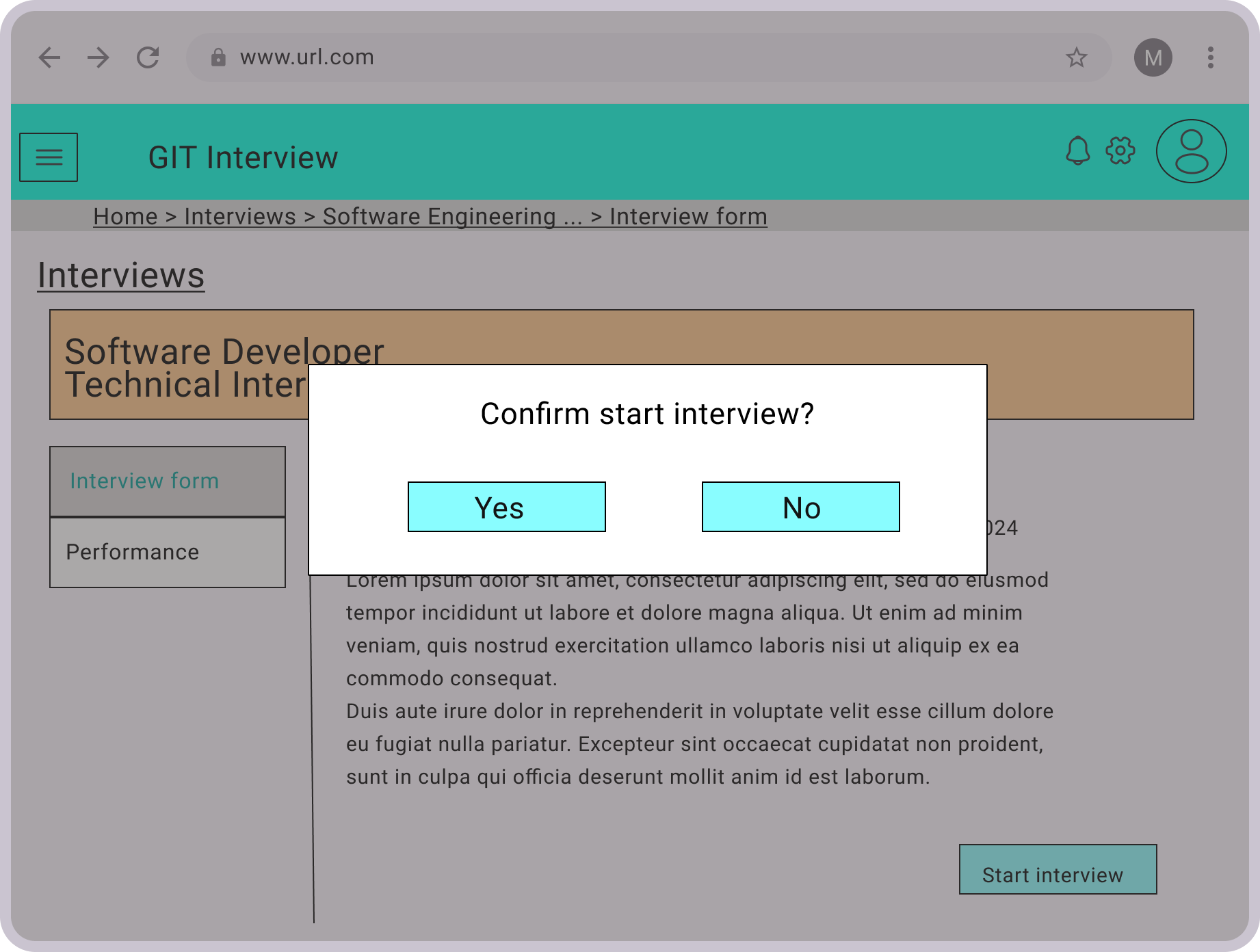
****

Figure 4.15: Confirm start interview prompt

Prompt user to confirm to prevent clicking mistake.

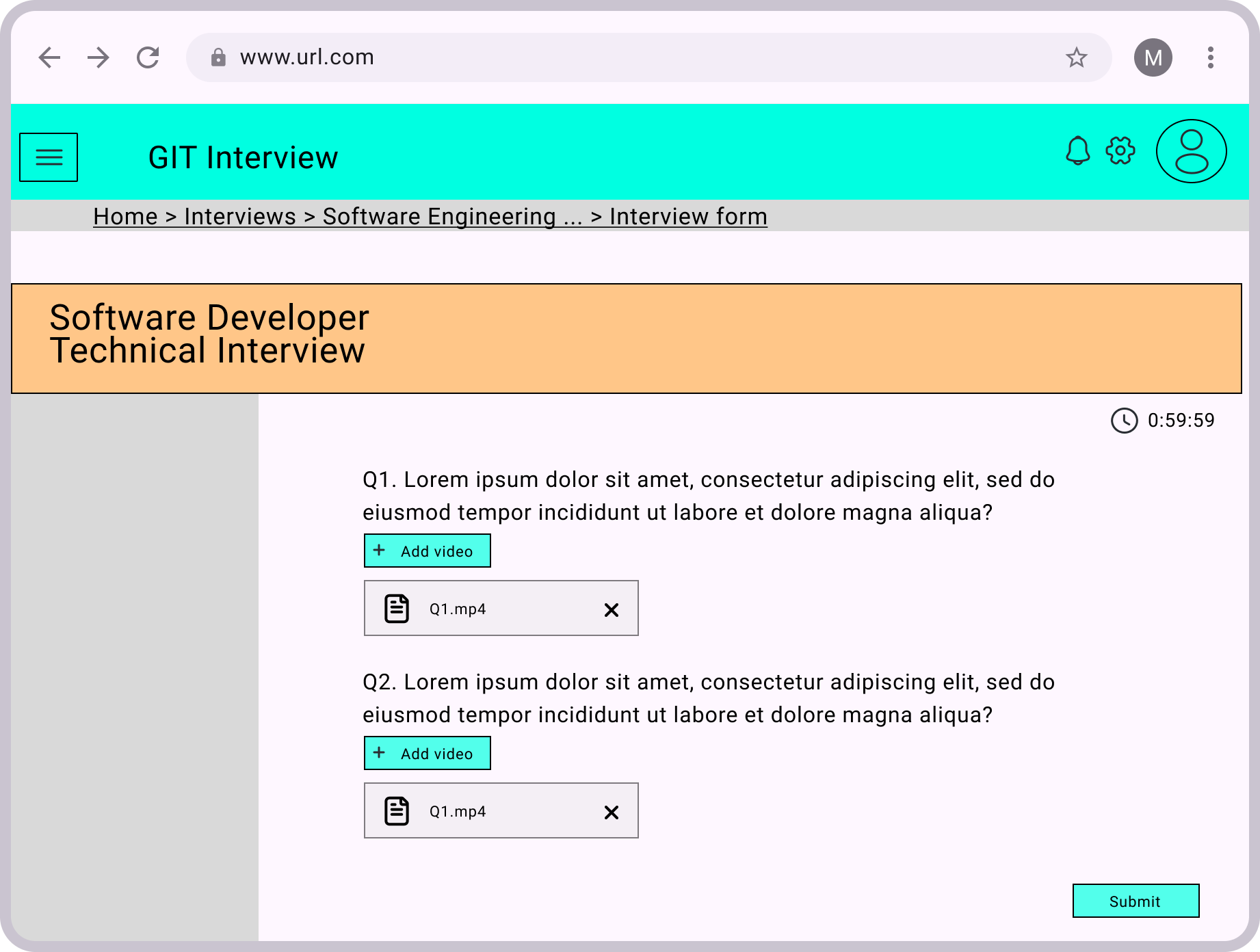
****

Figure 4.16: Interview form

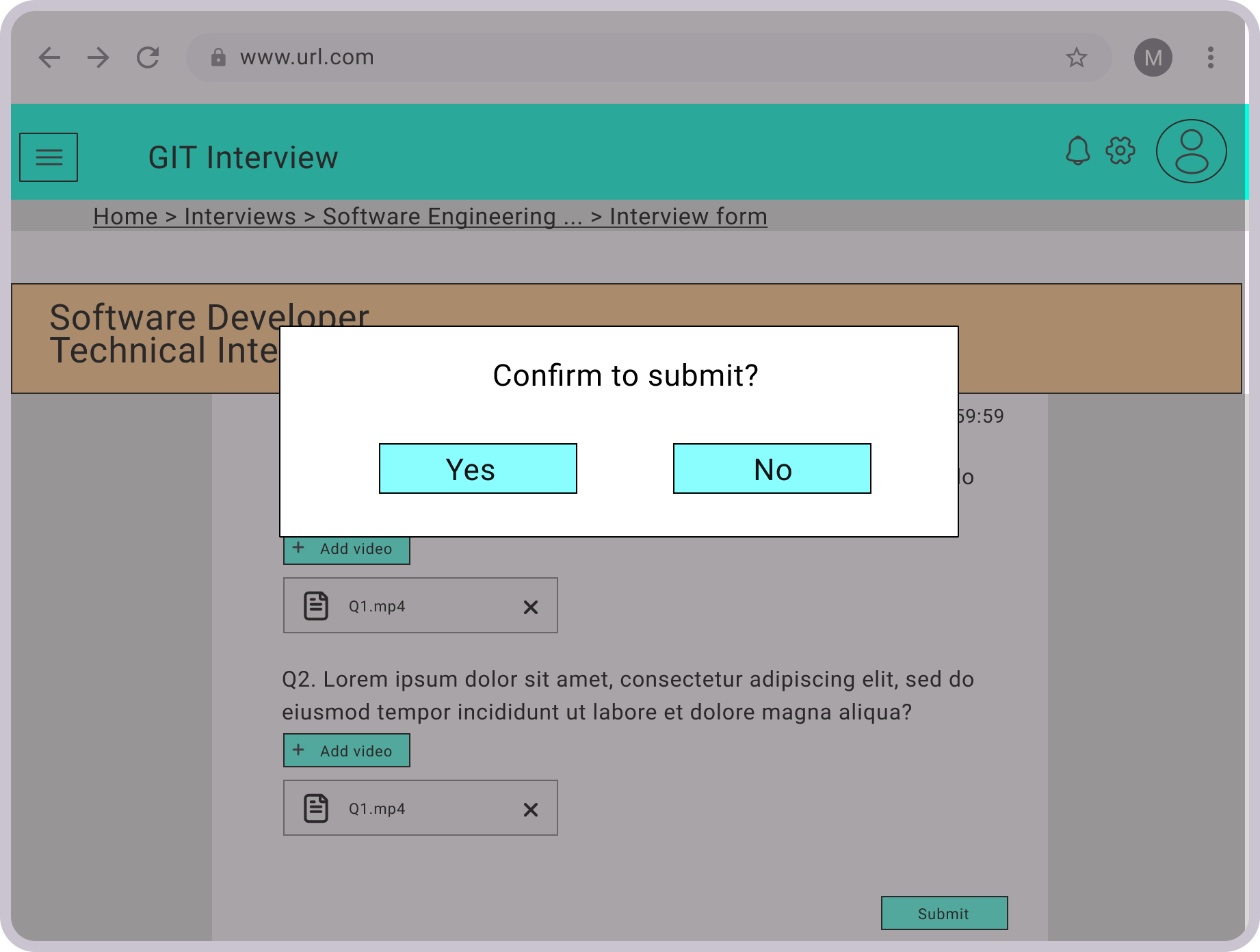
****

Figure 4.17: Confirmation before submit

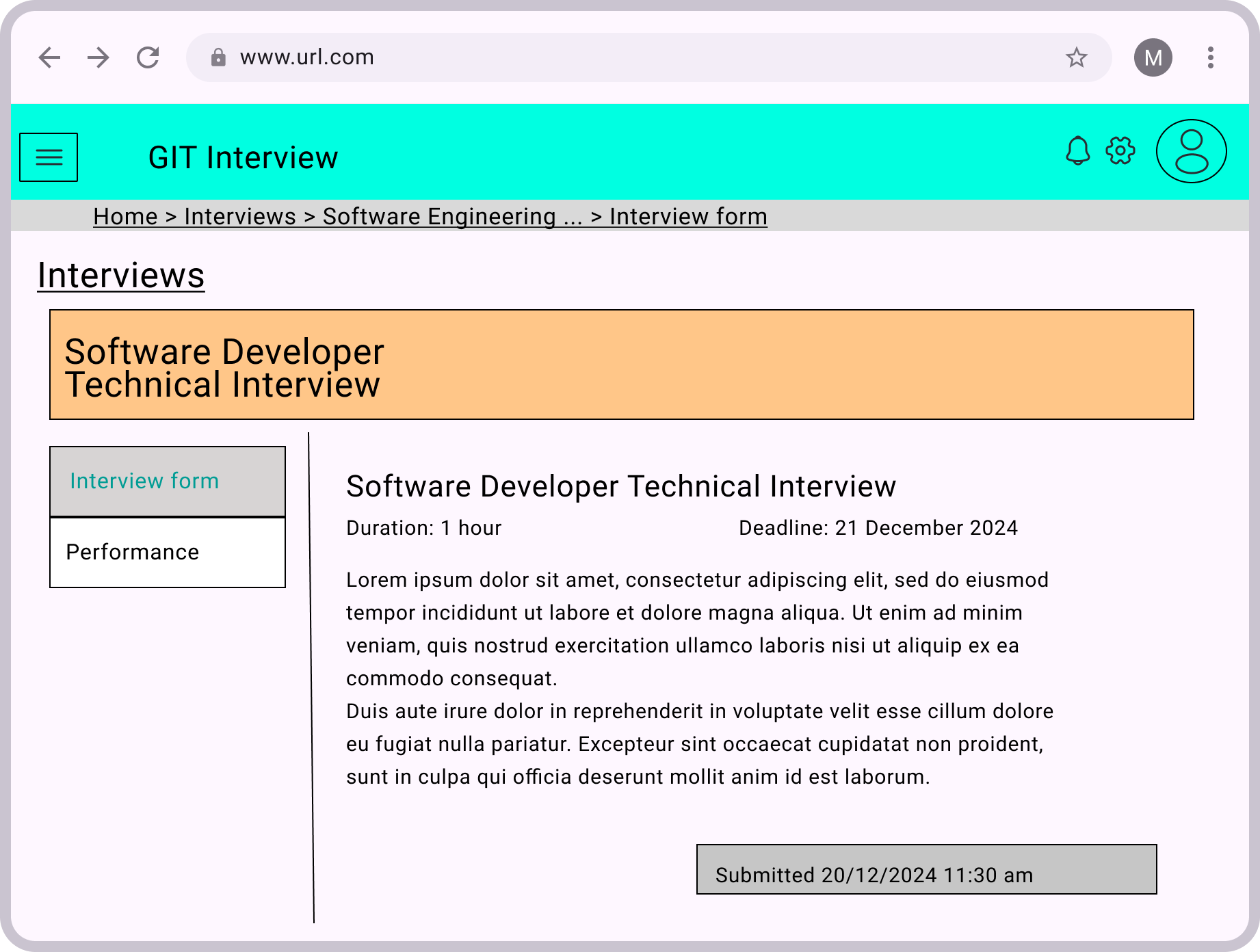
****

Figure 4.18: Interview page after it is submitted

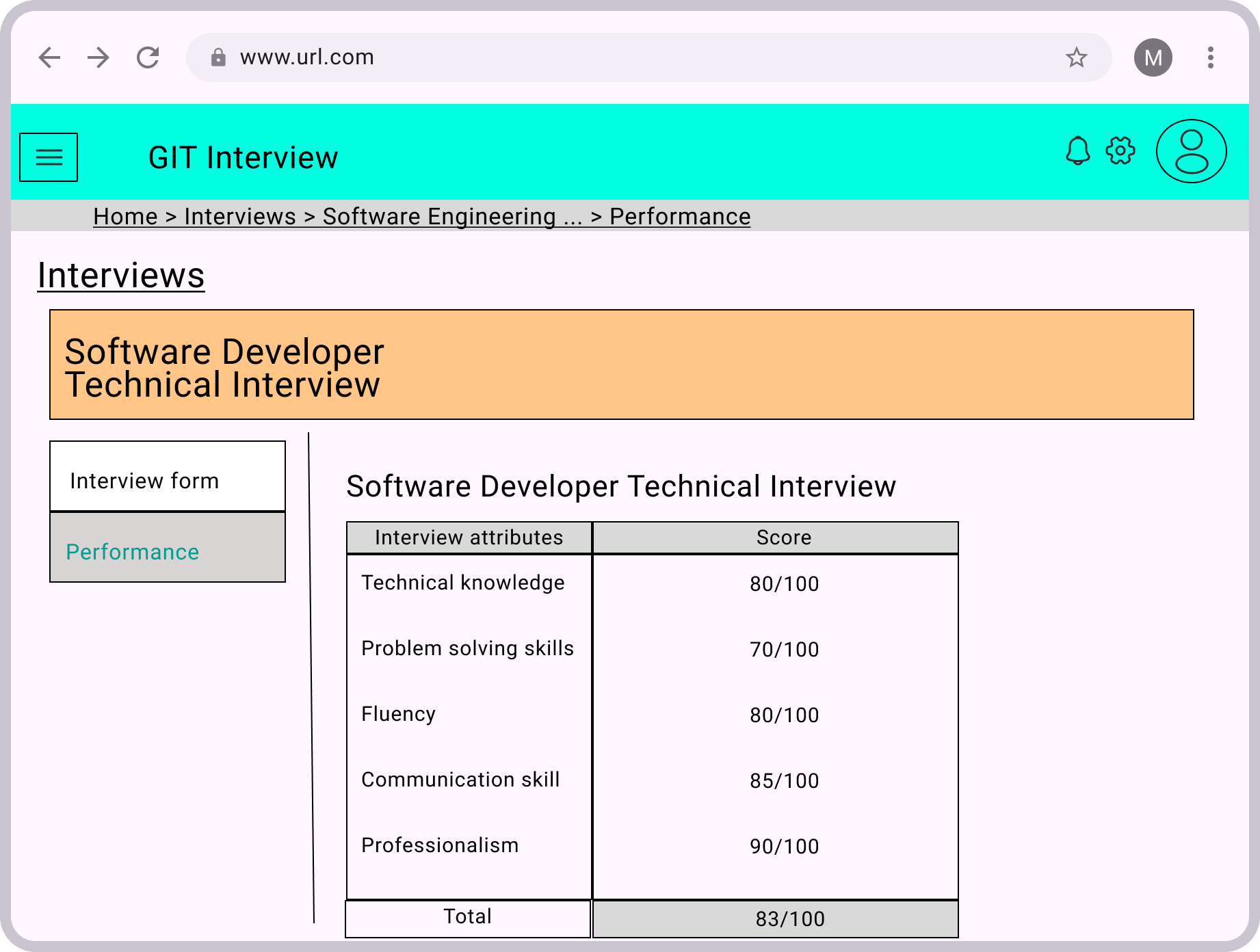
****

Figure 4.19: Candidate performance in interview page

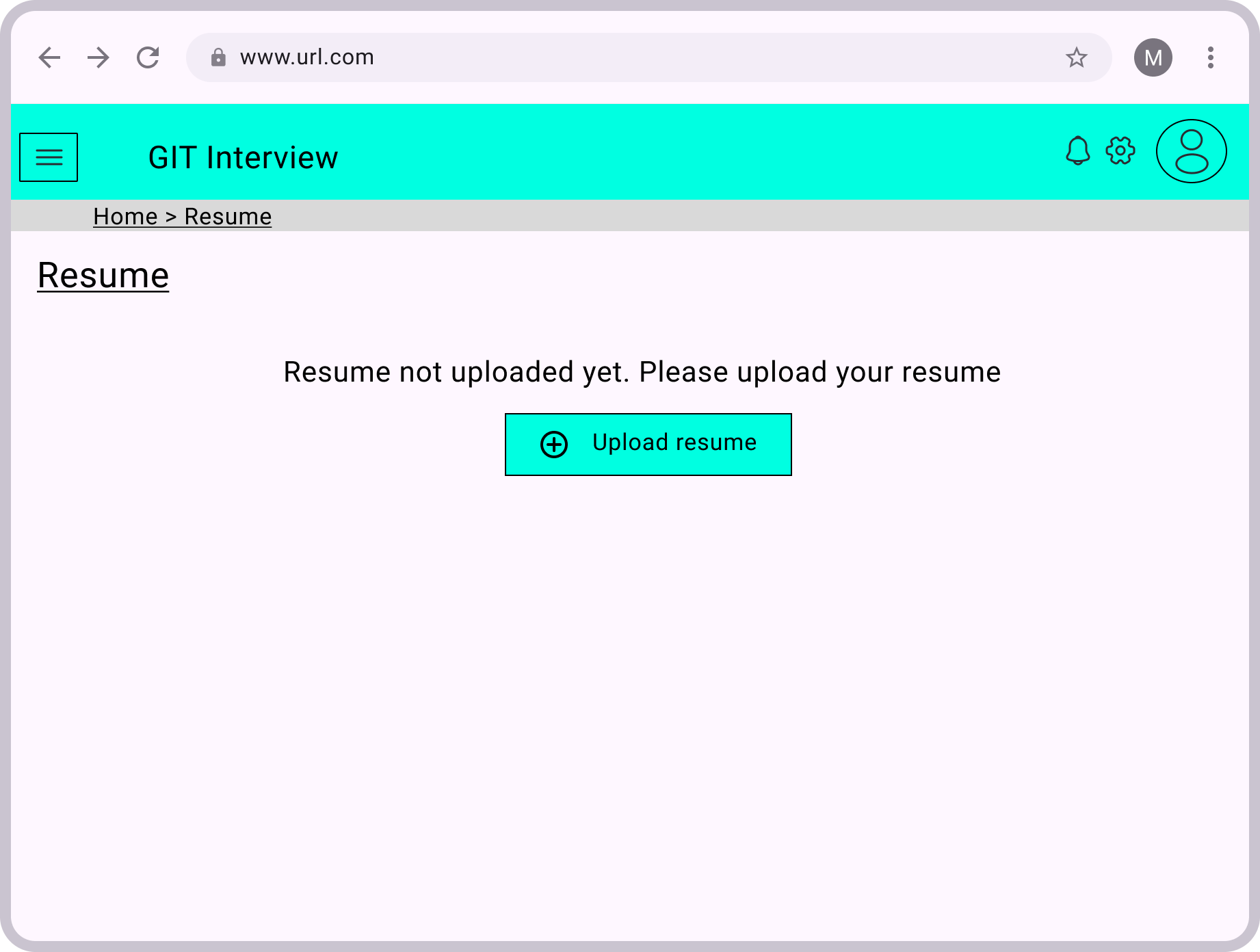
****

Figure 4.20: Resume upload page

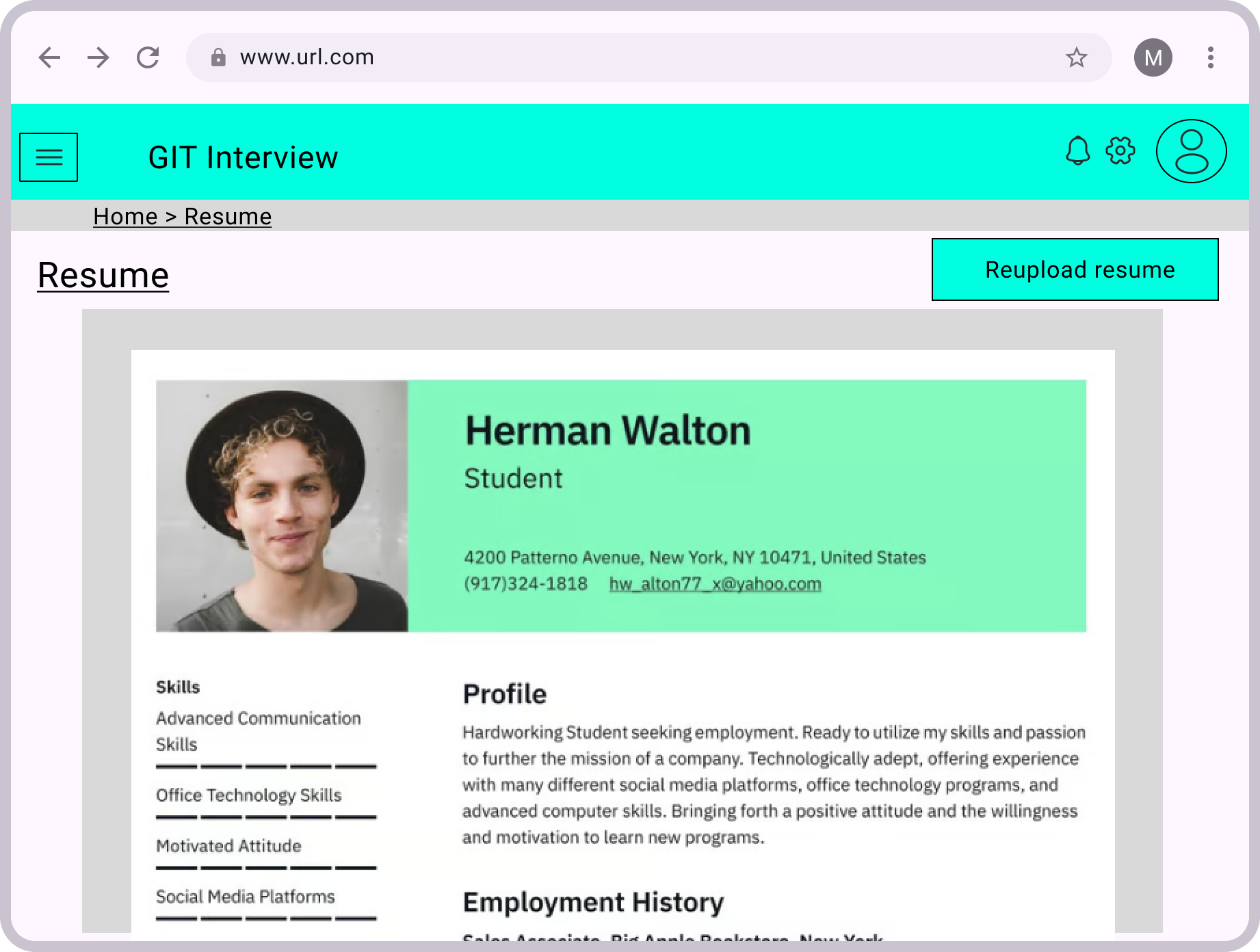
****

Figure 4.21: Resume page after resume is uploaded

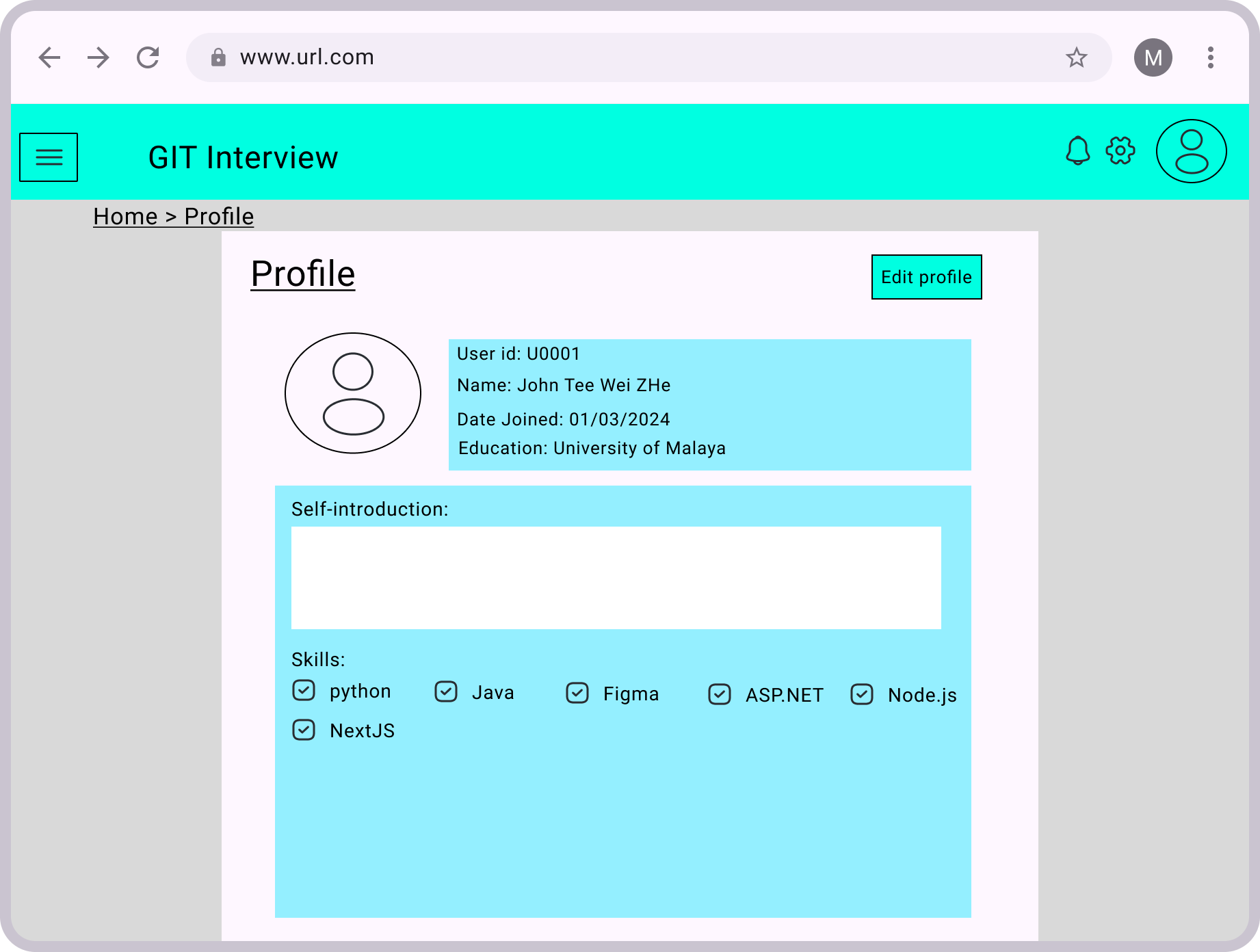
****

Figure 4.22: Profile page

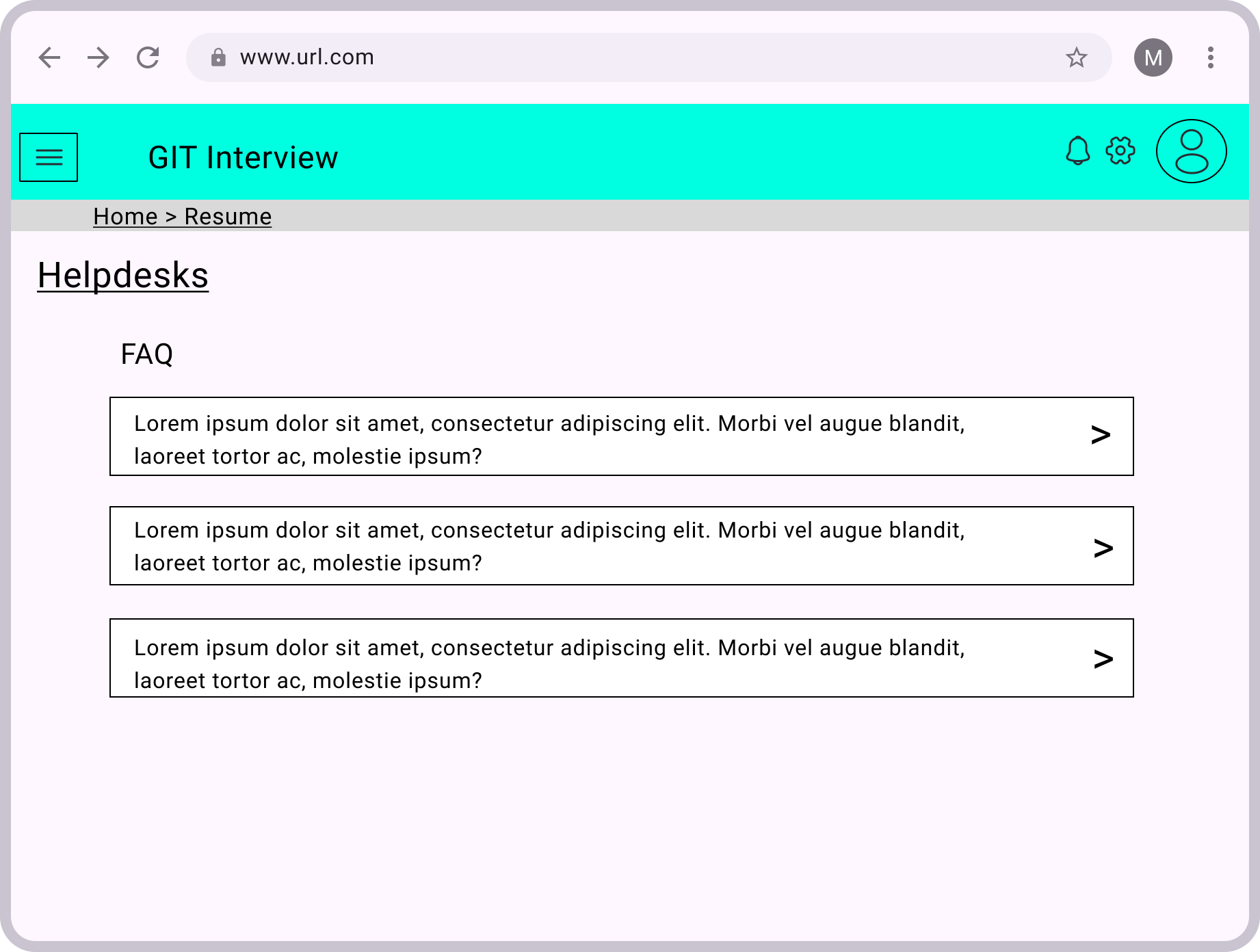


Figure 4.23: Helpdesk page

**Recruiter Module**

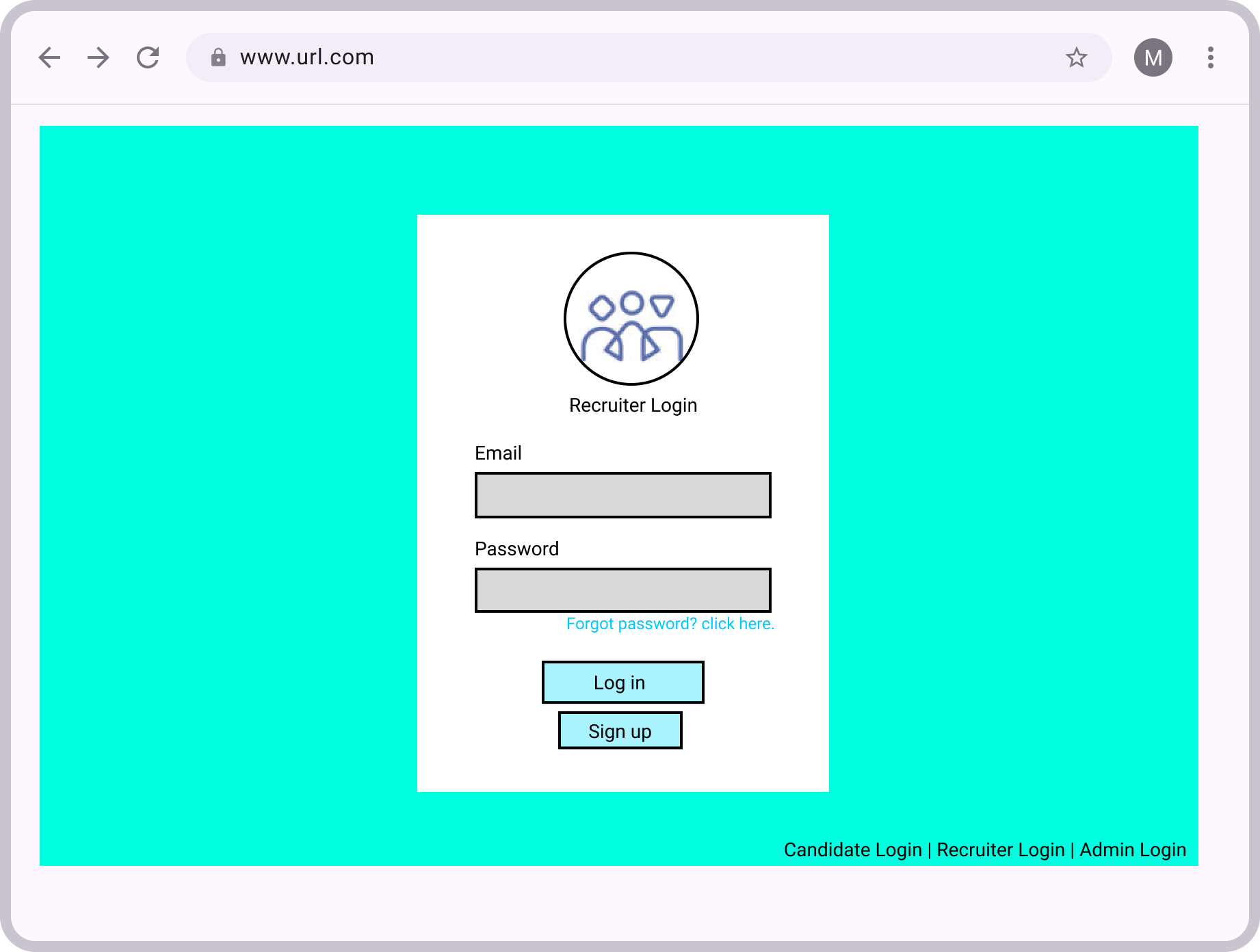
****

Figure 4.24: Recruiter login page

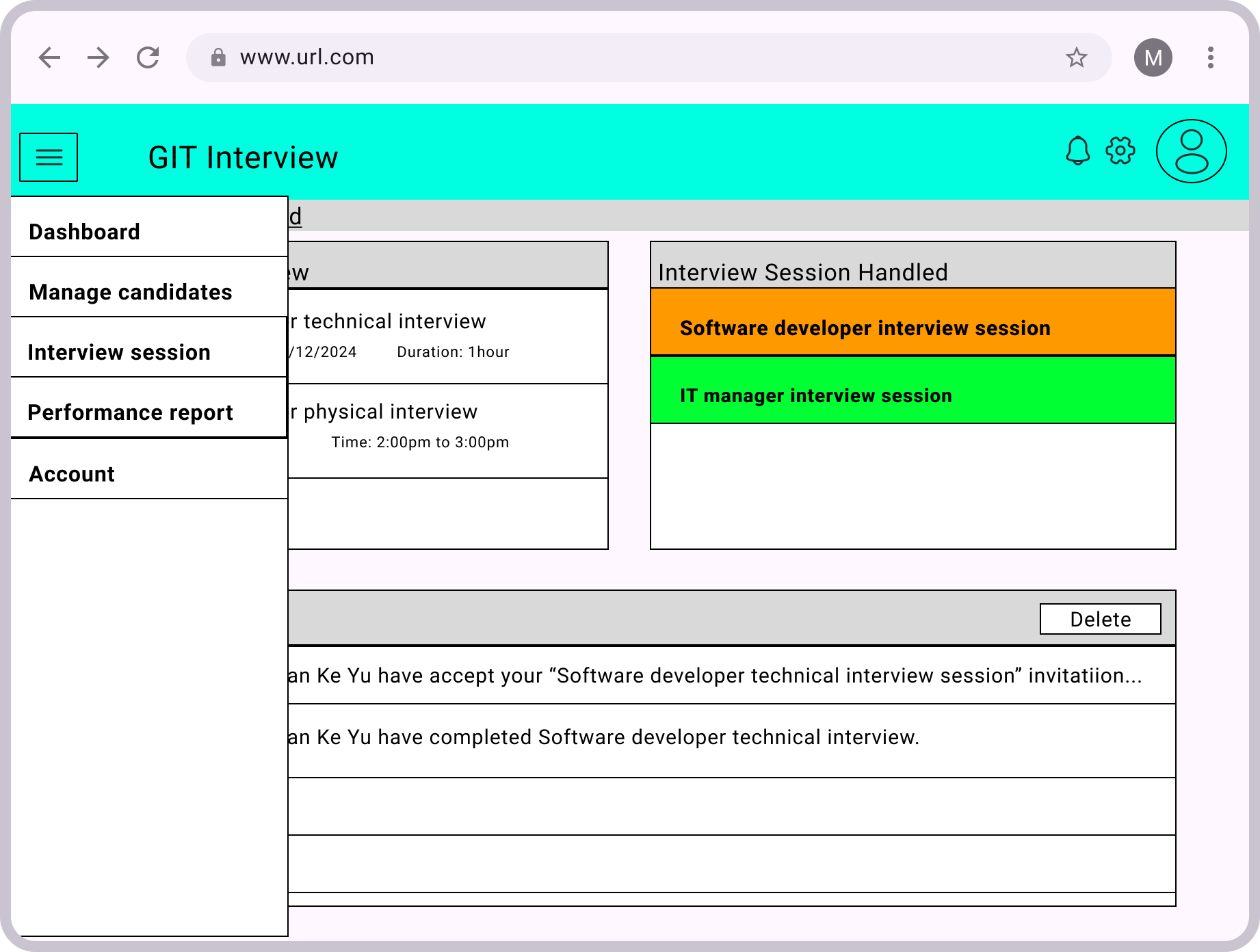
****

Figure 4.25: Recruiter dashboard and navigation menu

The difference with user dashboard is the interview session handled, recruiter can directly access from dashboard.

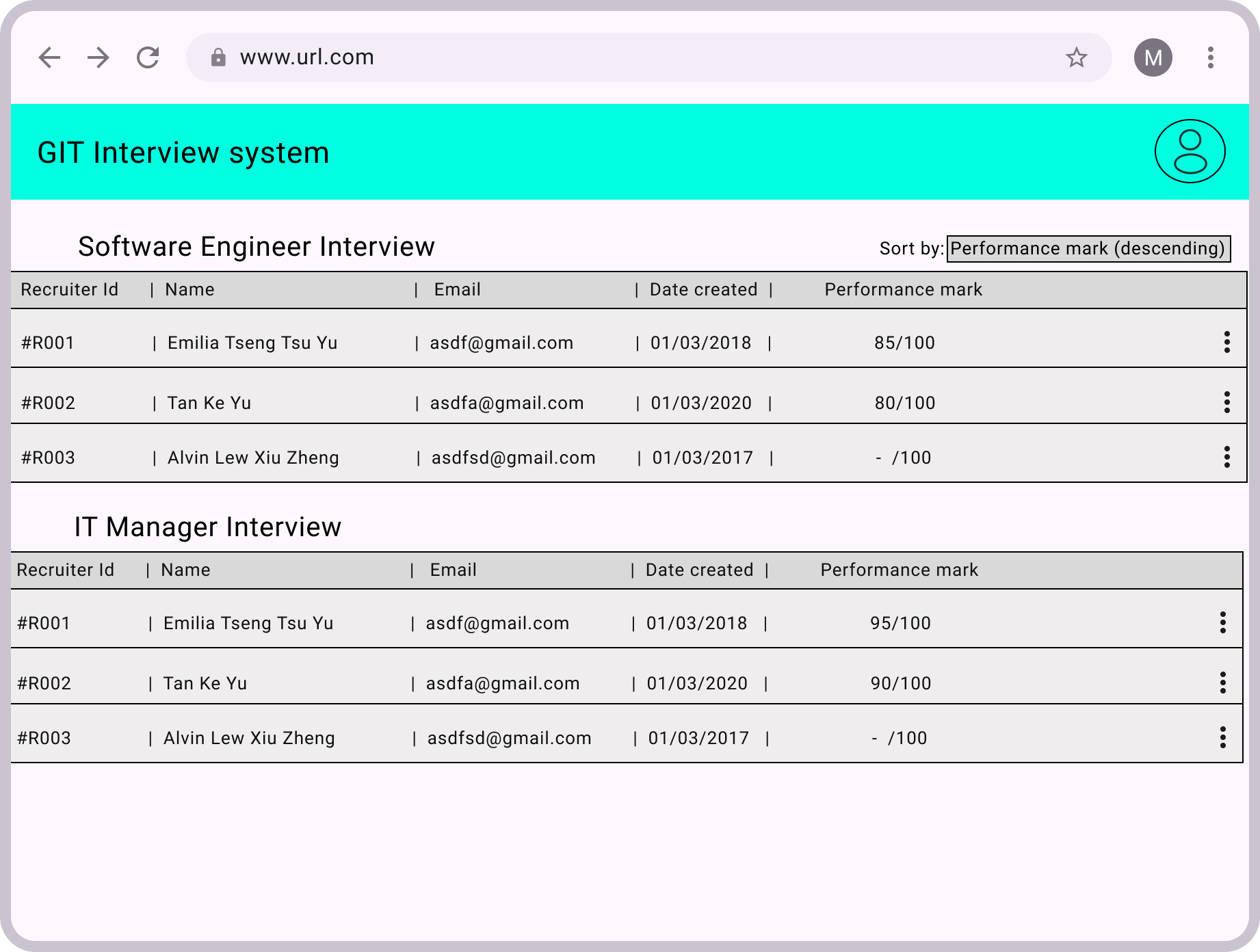
****

Figure 4.26: Manage candidate page

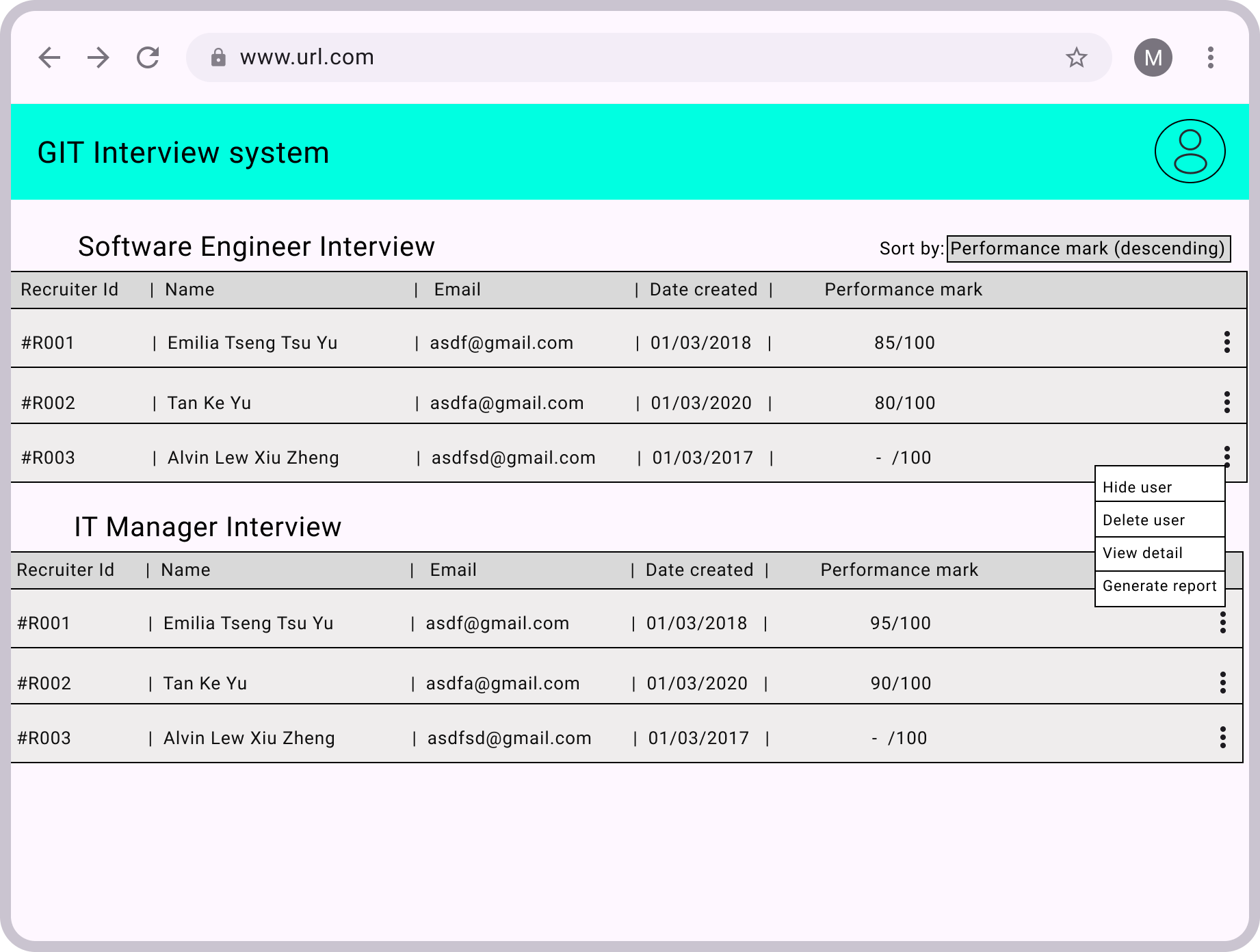
****

Figure 4.27: Candidate performance in interview page

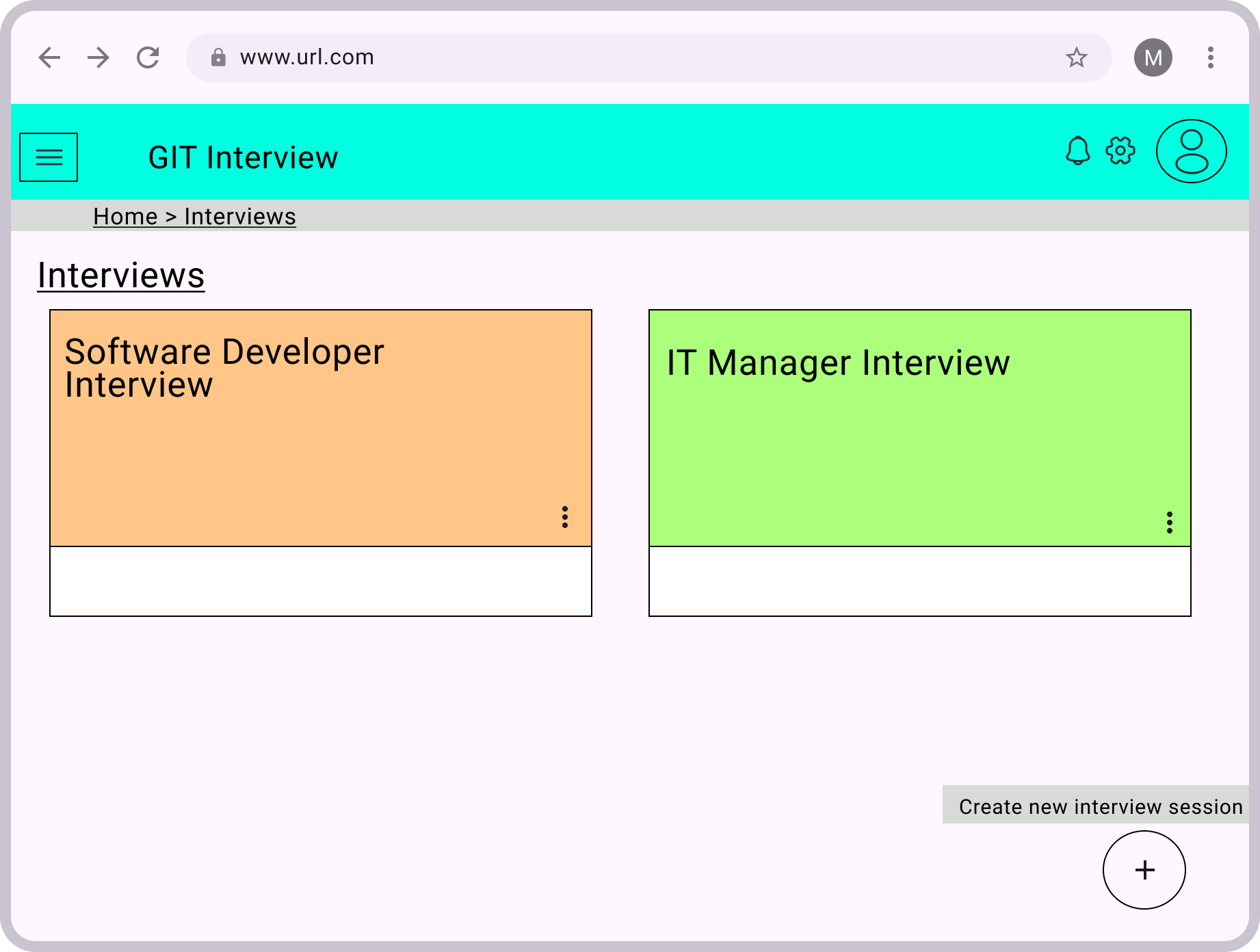
****

Figure 4.28: Manage interview session page

By using the button at bottom right, recruiter can create new interview session.

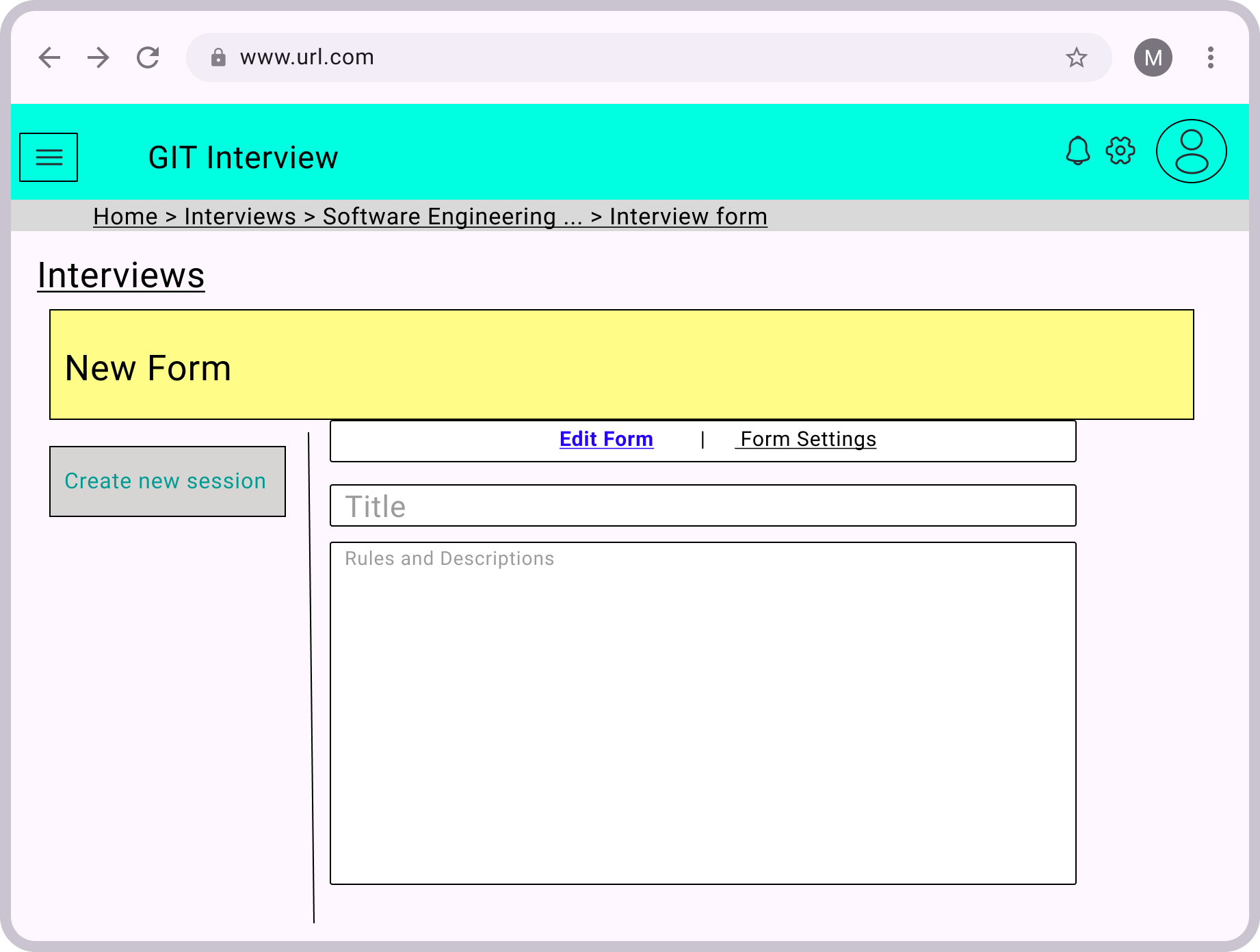
****

Figure 4.29: Create new session form

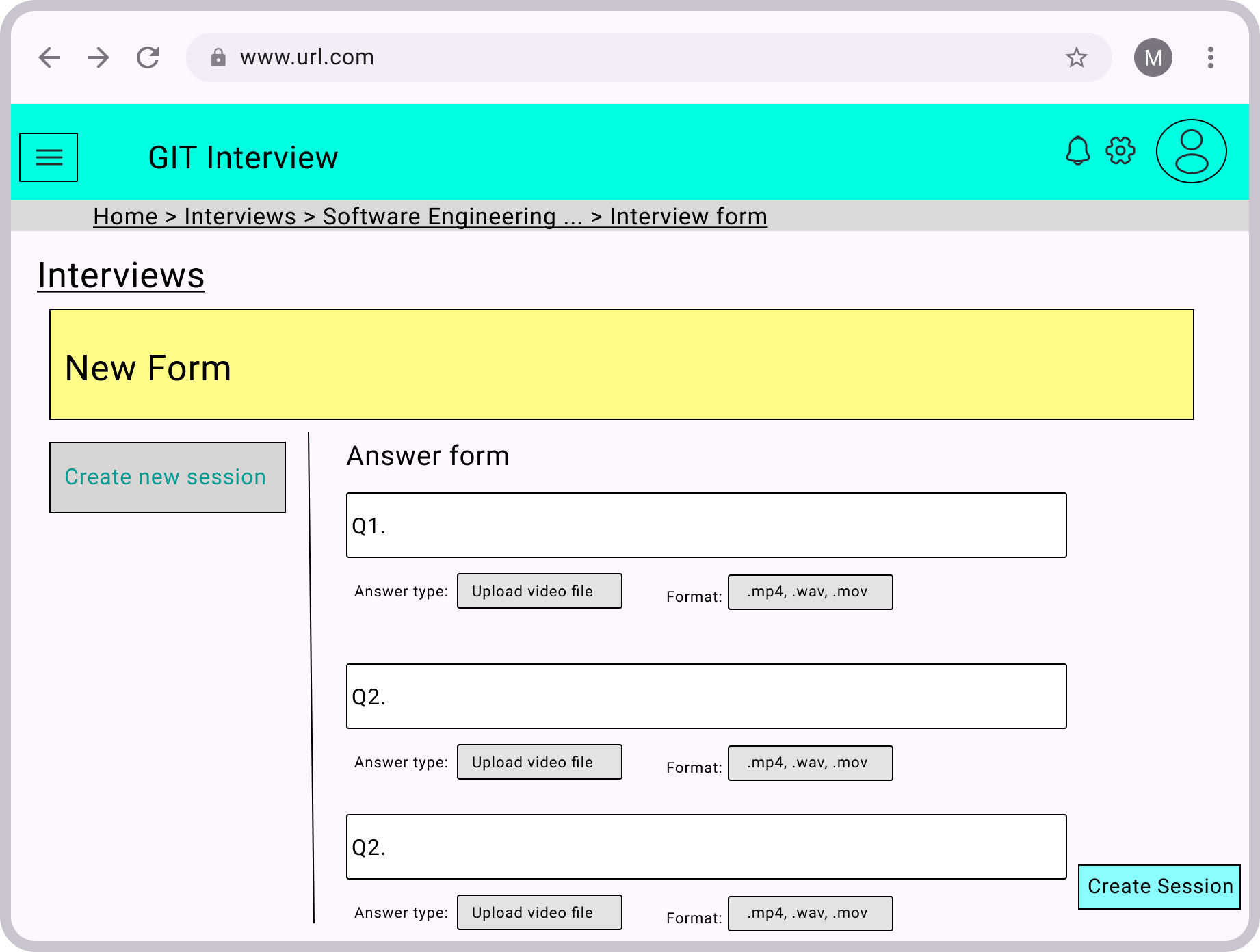
****

Figure 4.30: Create new interview session form

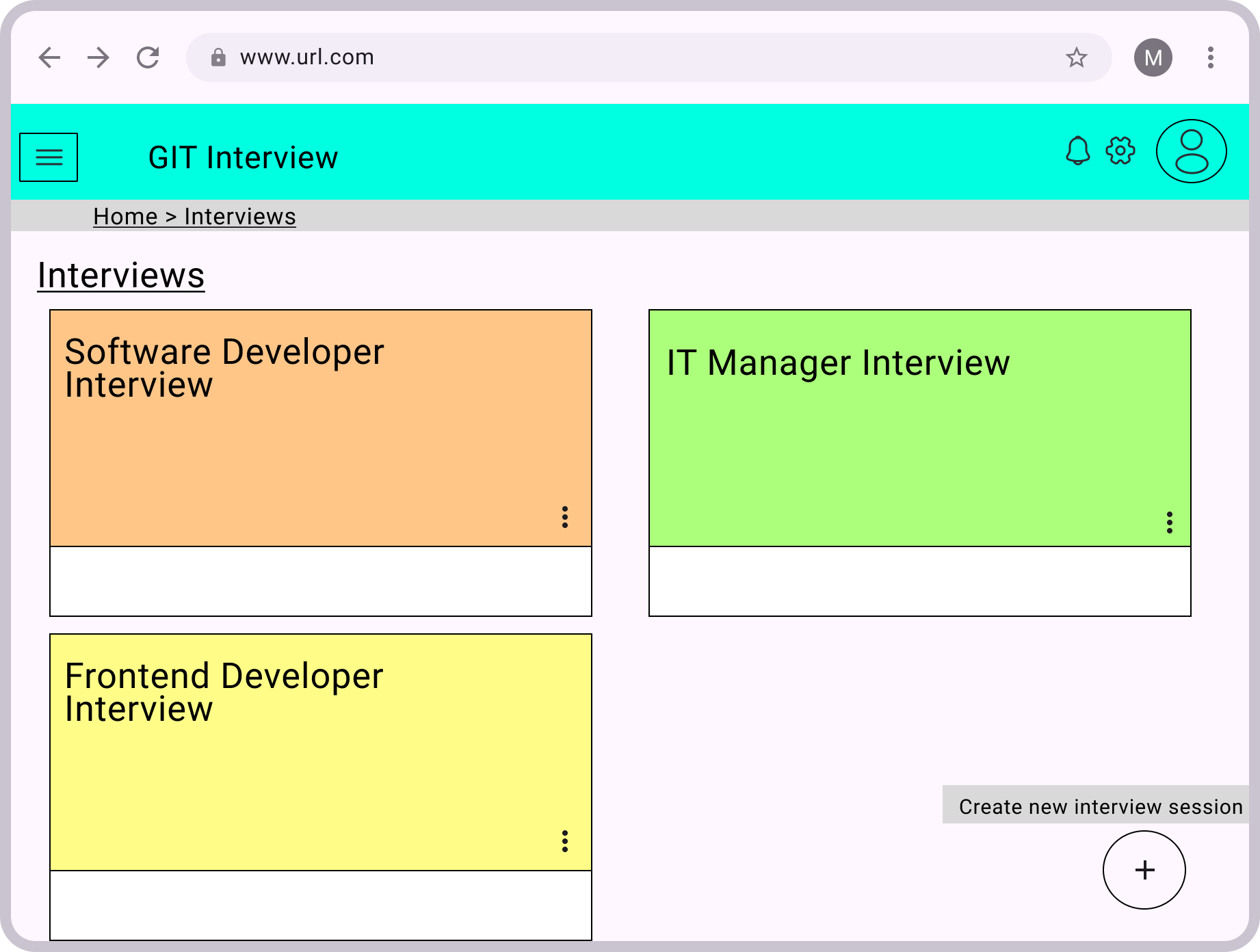
****

Figure 4.31: New interview session created page

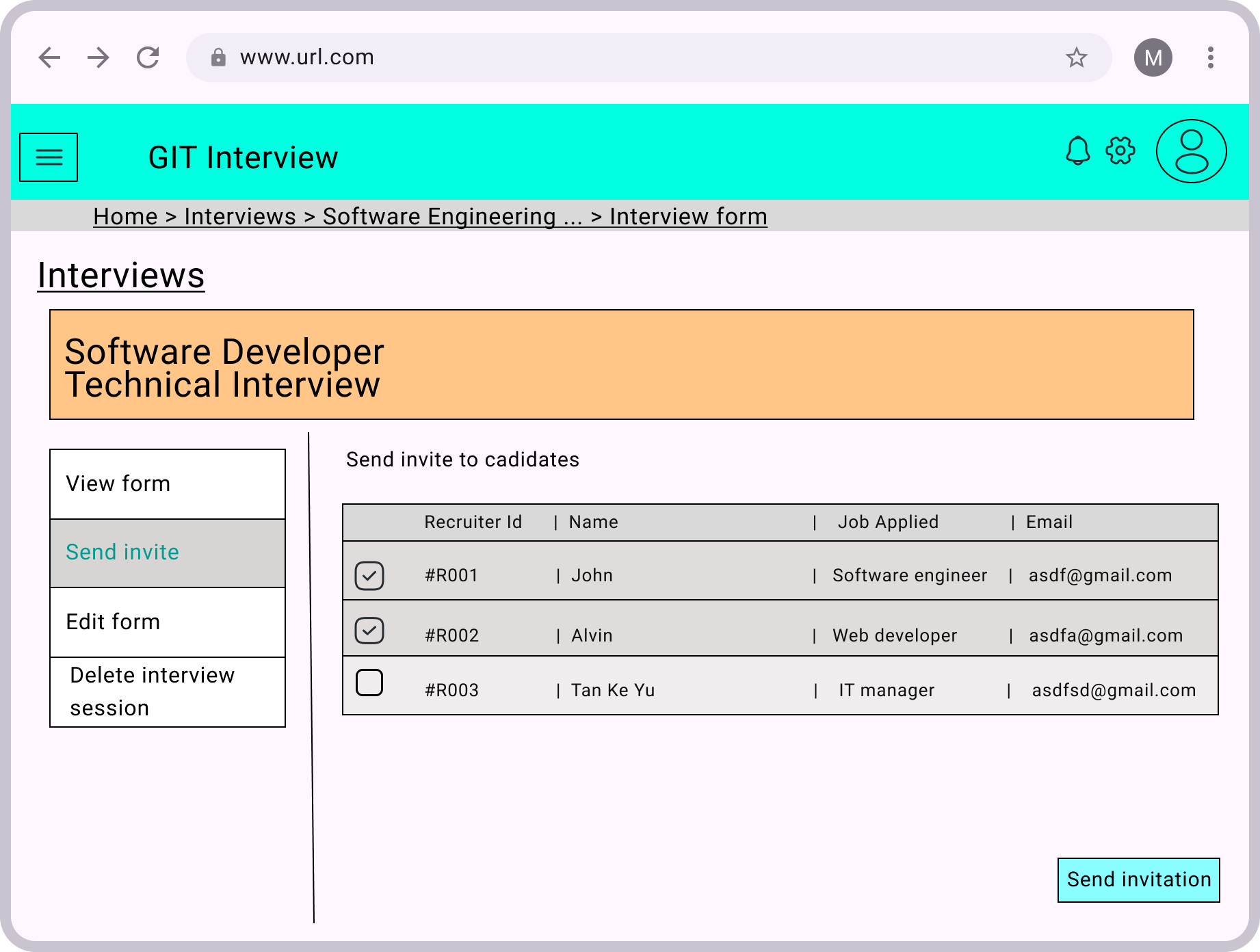
****

Figure 4.32: Invite candidate page

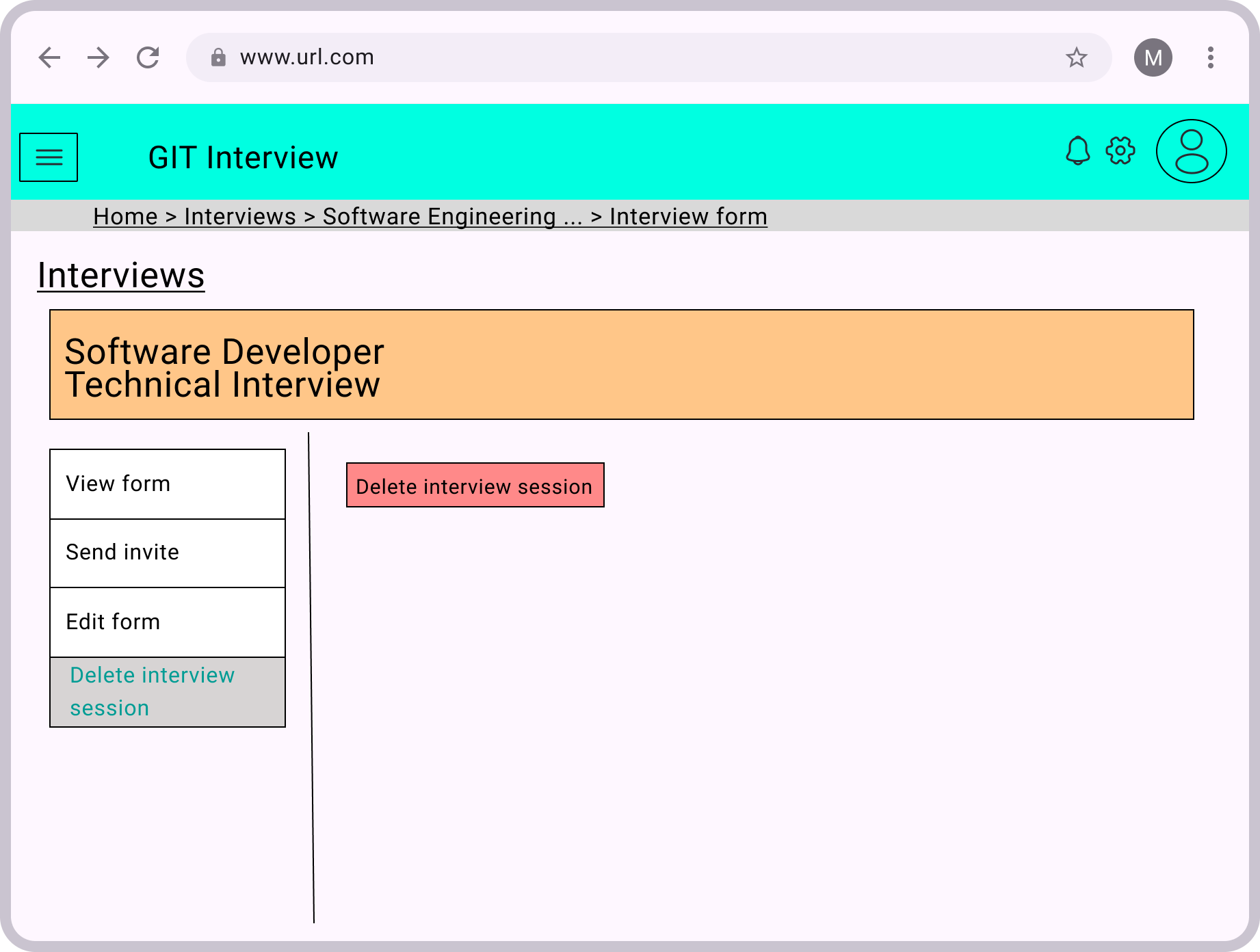
****

Figure 4.33: Delete interview session page

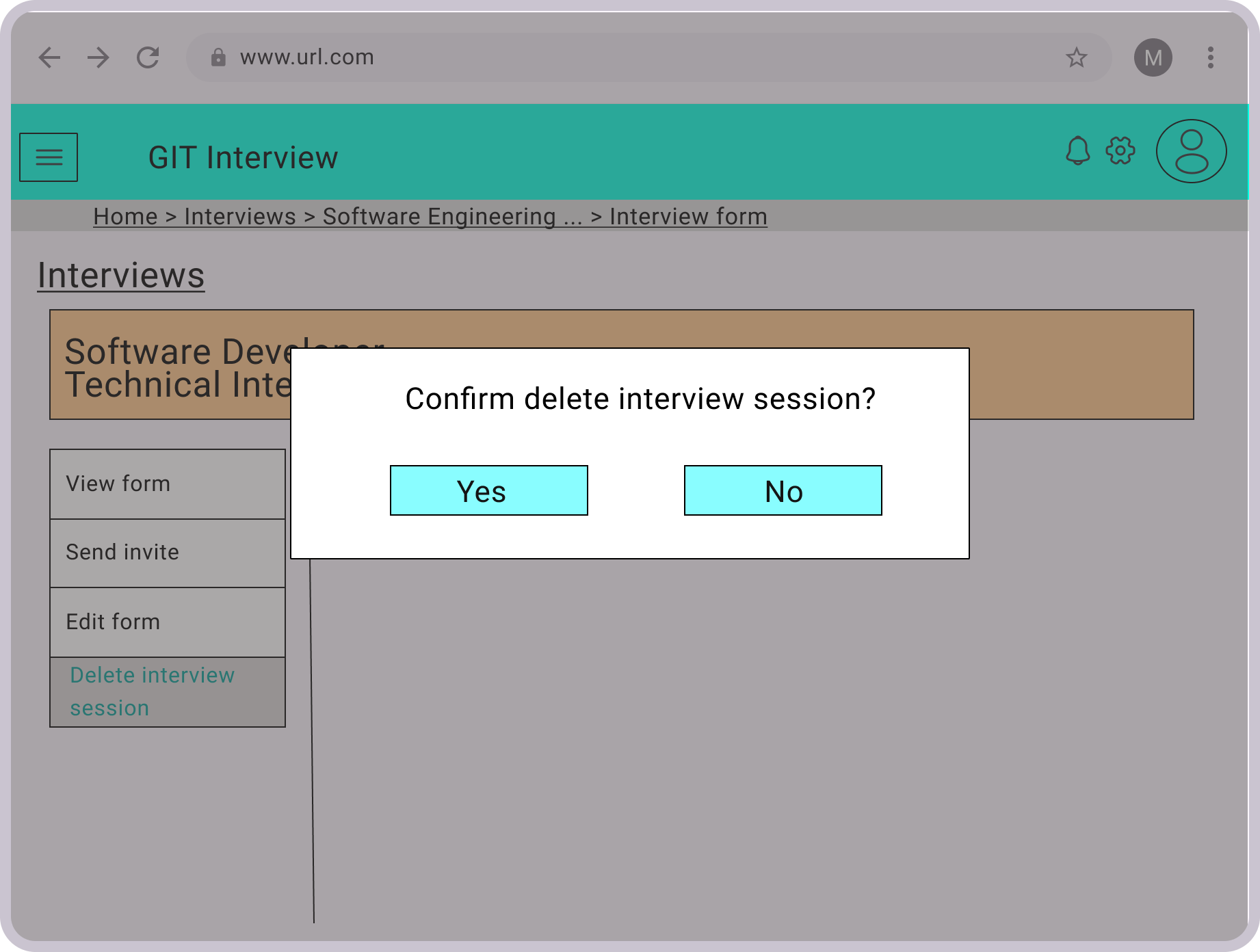
****

Figure 4.34: Confirmation of delete interview session page

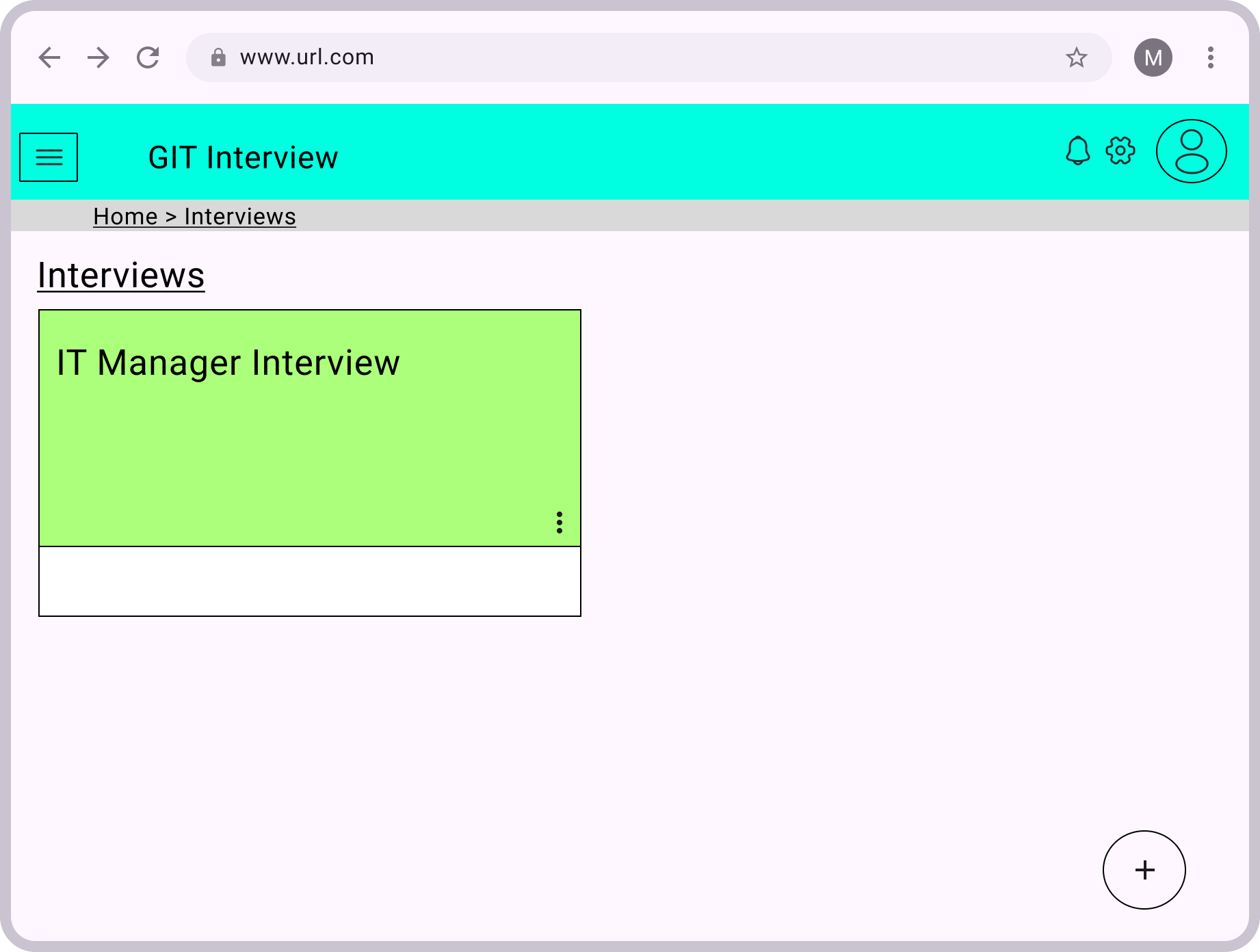
****

Figure 4.34: Interview session deleted page

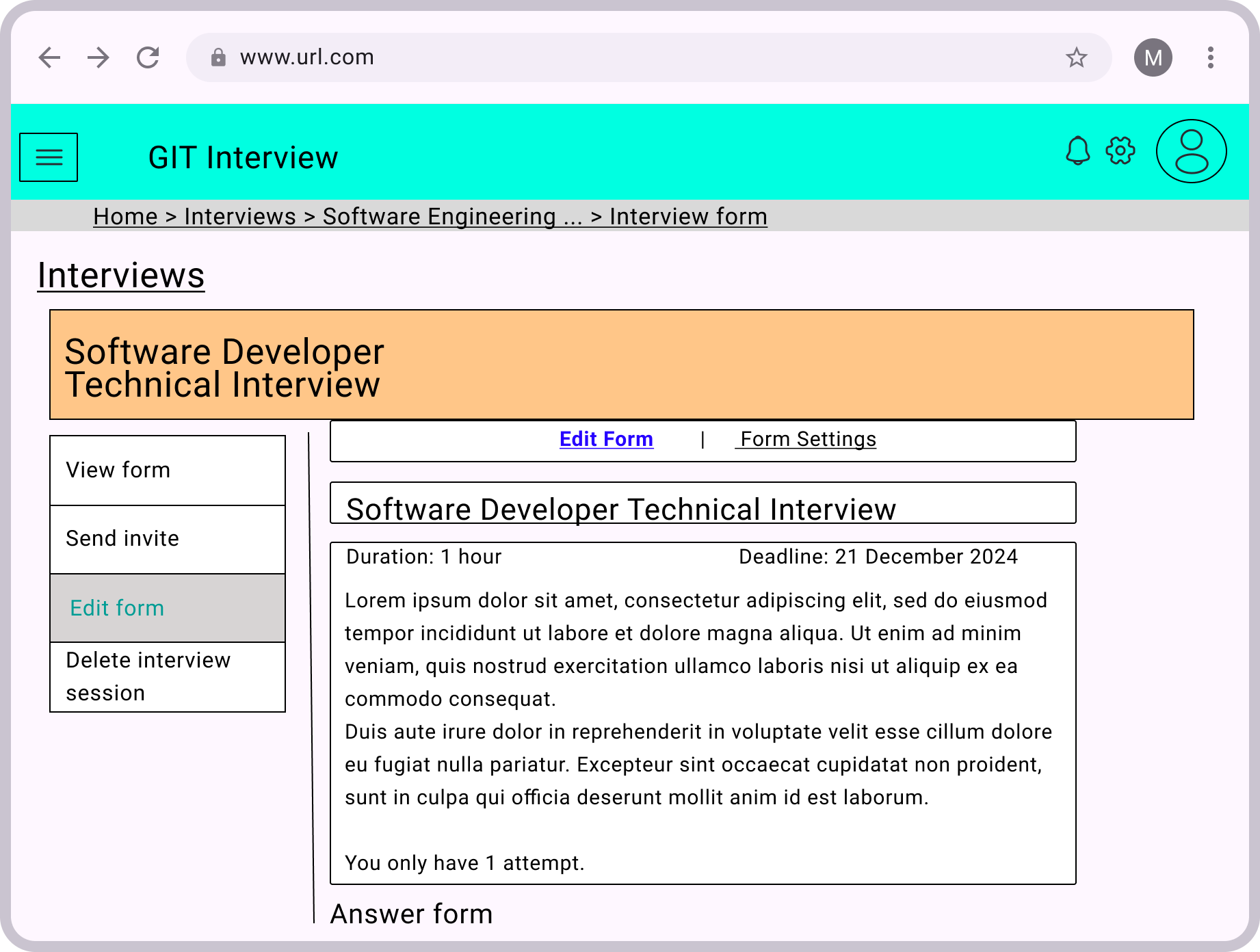
****

Figure 4.35: Edit interview form page

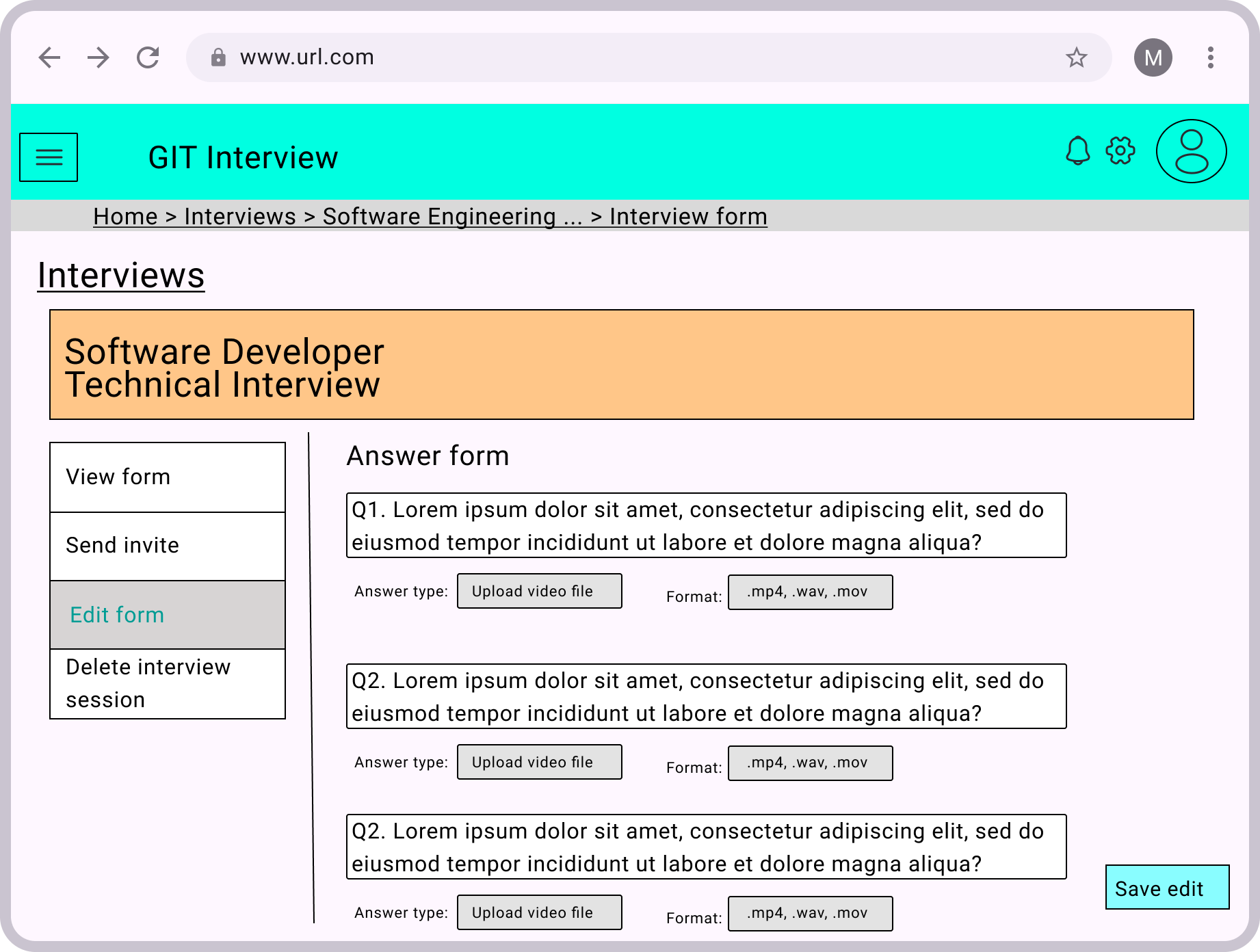
****

Figure 4.36: Edit interview form page

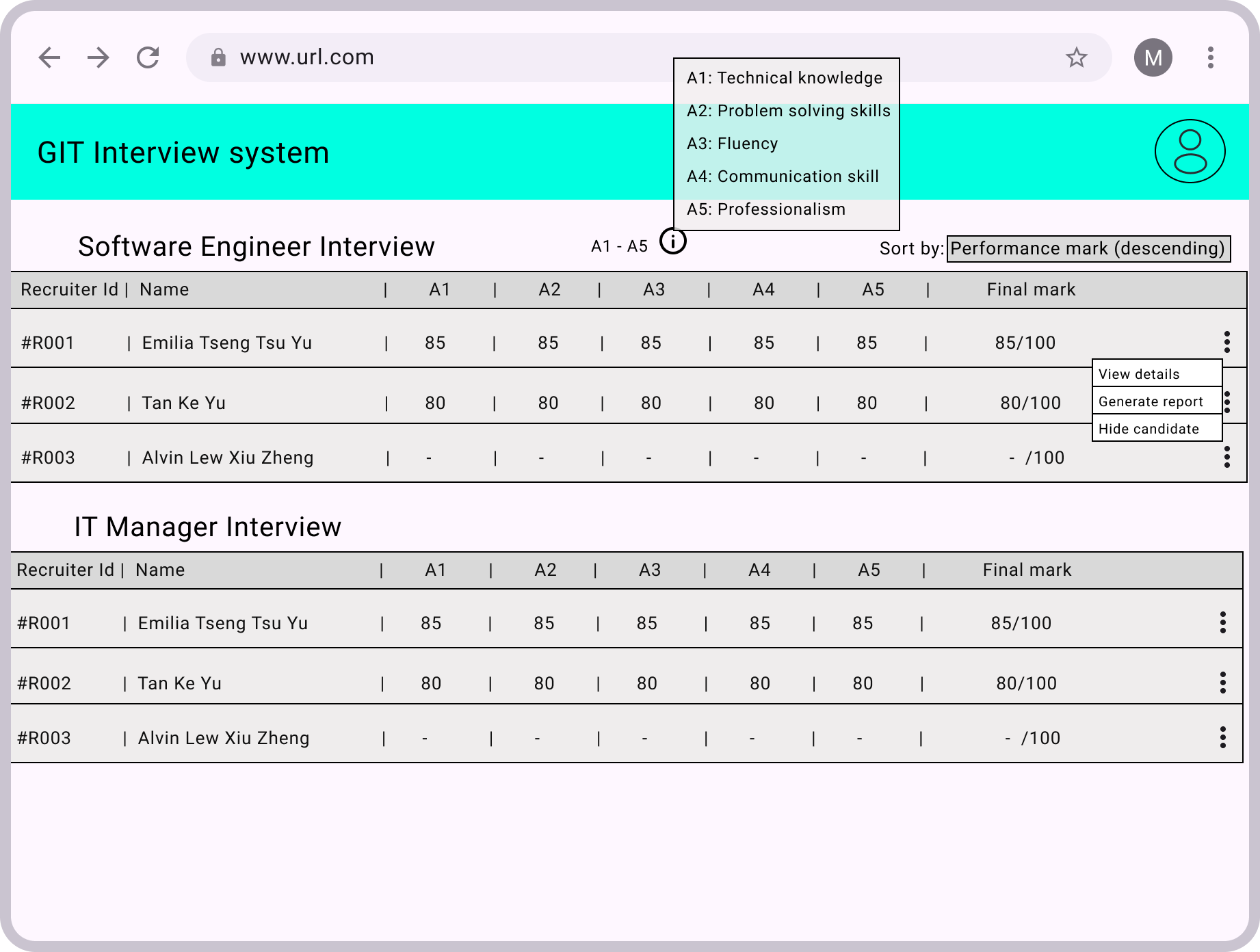
****

Figure 4.37: Edit interview form page

The information icon shows what does A1~A5 means. By clicking three dots, recruiter can view details and generate report or hide user if the candidate didn’t accept the offer for them to make better decision.

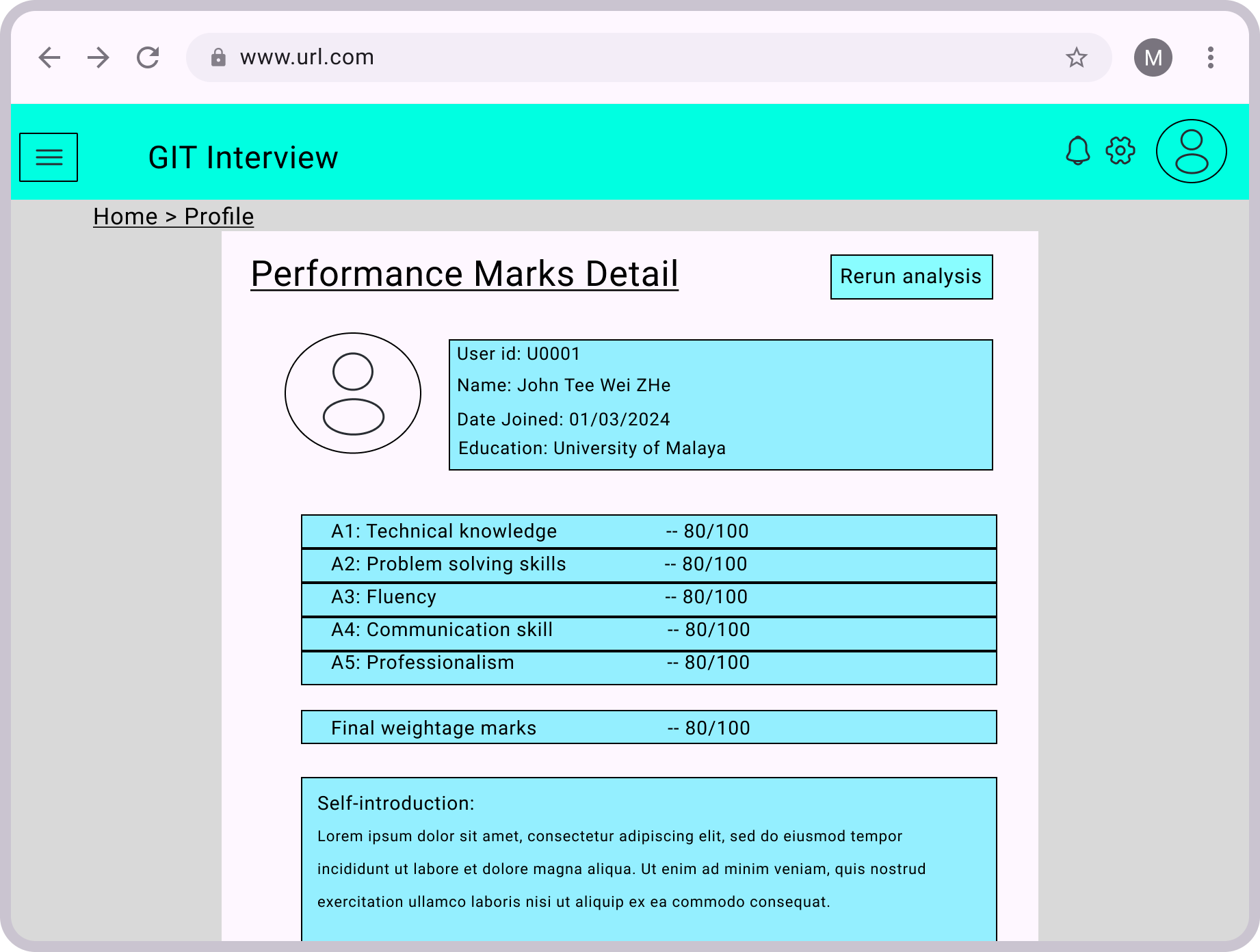
****

Figure 4.38: Edit interview form page

By clicking view details, recruiter is able to view the detail of candidate’s performance with their details shown. Recruiter can also rerun the analysis for this candidate if they think that there is problem in calculating candidate’s performance score.

## Chapter Summary and Evaluation

This chapter summarize how exactly the system will be implemented, including how the system is designed from OOP class diagram, how the process is done, how the database is designed and how UI is design using diagrams and tables. This chapter give developer a clear instruction on exactly how the system is implemented.

## References and Citations

Blacksmith, N., Willford, J., & Behrend, T. (2016). Technology in the Employment Interview: A Meta-Analysis and Future Research Agenda. Personnel Assessment and Decisions, 2(1). <https://doi.org/10.25035/pad.2016.002>

University of Florida. (n.d.). Identifying and avoiding interview biases. Identifying and Avoiding Interview Biases. <https://training.hr.ufl.edu/resources/leadershiptoolkit/transcripts/identifying_and_avoiding_interview_biases.pdf>

Zhang, P. & AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT. (2024). Application of artificial intelligence (AI) in recruitment and selection: the case of Company A and Company B. In Journal of Business and Management Studies (pp. 224–225) [Journal-article]. Al-Kindi Centre for Research and Development. https://doi.org/10.32996/jbms

Poonam. S. (2023, December 23). How can you use sentiment analysis to evaluate a candidate’s attitude in selection? How Sentiment Analysis Can Evaluate Candidate Attitude. <https://www.linkedin.com/advice/3/how-can-you-use-sentiment-analysis-evaluate-candidates-gufqc>

B C Lee, B. Y., & Kim. (2021). Development of an AI-Based Interview System for Remote Hiring. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, *12*(3), 654–663. https://doi.org/10.34218/IJARET.12.3.2021.060

Borman, W. C., & Motowidlo, S. J. (1997). Task Performance and Contextual Performance: The Meaning for Personnel Selection Research. Human Performance, 10(2), 99–109. <https://doi.org/10.1207/s15327043hup1002_3>

Apple, J. M., Guerci, J. C., Seligson, N. D., & Curtis, S. D. (2020). Adding the second T: Elevating STAR to START for behavioral interviewing. American Journal of Health-System Pharmacy, 78(1), 18–21. https://doi.org/10.1093/ajhp/zxaa356

Ibrahim, N. a. N., Rani, N. S. A., Jamri, M. H., Bakar, M. H., Wahab, S. A., Mahbob, M. H., & Kahar, N. (2022). The importance of Non-Verbal Communication in organizations. International Journal of Academic Research in Business and Social Sciences, 12(6). https://doi.org/10.6007/ijarbss/v12-i6/13901

R. N. Kulkarni, & Swetha Koduri. (2024, February 23). Novel approach for the extraction of keywords from text document. IEEE Conference Publication | IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/10607099?casa_token=Ii2wBO61LZ8AAAAA:1cr_m0skQJNTZdnjOOD5TVOxsCB6OW-LDUDUd_FFTieI0-R1h2IXk2b7KjfZqMeCJ4rhHwFZCG7GYSk>

G. Sri Harsh, Y. Sai Sri Vivek, Maneesha. P, Saroja Kumar Rout, S. Ranjith Reddy, & Bijaya Kumar Sethi. (2024, March 1). Automated Interview Evaluation System Using RoBERTa Technology. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/10530789?casa\_token=LxTNamQmbhIAAAAA:RvHNNkQQVzlNPM0p\_wRjNIX7xVEo068nHI5m-ELhoK1XJYj-JTJOf1pgTXtSFAndiqxUXaRQ5oKCh3A

Dhivya Chandrasekaran, & Vijay Mago. (2021). Comparative Analysis of Word Embeddings in Assessing Semantic Similarity of Complex Sentences. IEEE Journals & Magazine | IEEE Xplore. <https://ieeexplore.ieee.org/abstract/document/9652410>

IANCU, B. (2019). Evaluating Google Speech-to-Text API’s Performance for Romanian e-Learning Resources. Informatica Economica, 23(1/2019), 17–25. https://doi.org/10.12948/issn14531305/23.1.2019.02

Noel, M. (2019, August 12). Watson Speech-To-Text: How to Train Your Own Speech “Dragon” — Part 1: Data Collection and Preparation. Medium; IBM Watson Speech Services. <https://medium.com/ibm-watson-speech-services/watson-speech-to-text-how-to-train-your-own-speech-dragon-part-1-data-collection-and-fdd8cea4f4b8>

Stoyan Stoitsev. (2023, August 14). OpenAI vs. Google vs. Azure: A Speech-to-Text Battle. Medium; Medium. https://sstoitsev.medium.com/google-vs-azure-a-speech-to-text-battle-f740aa481e8e

AltexSoft. (2020, May 29). Comparing API Architectural Styles: SOAP vs REST vs GraphQL vs RPC. AltexSoft. <https://www.altexsoft.com/blog/soap-vs-rest-vs-graphql-vs-rpc/>

Michael Goodwin. (2024, December 3). What is an API (application programming interface)? IBM. <https://www.ibm.com/think/topics/api>

Hall, D. (2024, October 30). How to Use an API: Just the Basics. TechnologyAdvice. https://technologyadvice.com/blog/information-technology/how-to-use-an-api/

Amazon Web Services, Inc. (2023). Tutorial - Deploy a machine learning model to a REST API - Amazon Web Services. https://aws.amazon.com/getting-started/hands-on/deploy-a-machine-learning-model-to-a-rest-api/

HireVue. (n.d.). HireVue Hiring Platform: Video Interviews, Assessment, Scheduling, AI, Chatbot | HireVue. hirevue.com. https://www.hirevue.com/

Sapia. (2024, November 5). Sapia.ai | The world’s first AI Smart Interviewer. Sapia.ai. https://sapia.ai/

Interviewer.AI. (2024, September 17). Interviewer.AI | End-to-End AI Video Interview Platform [Online forum post]. https://interviewer.ai/

React. (n.d.). React – A JavaScript library for building user interfaces. https://legacy.reactjs.org/

Vora, R. (2024, April 6). React: How React Works Under The Hood - Ruchi Vora - Medium. Medium. <https://medium.com/@ruchivora16/react-how-react-works-under-the-hood-9b621ee69fb5#:~:text=React%20creates%20a%20virtual%20DOM,DOM%20and%20the%20previous%20one>.

Angelina. (2024, November 5). 10 Advantages of React for Building Interactive User Interfaces. PixelCrayons. https://www.pixelcrayons.com/blog/dedicated-teams/advantages-of-react/

Adedimeji, I. L. (2023, September 17). Node.js Architecture: Understanding Node.js Architecture. Medium. <https://medium.com/@ibrahimlanre1890/node-js-architecture-understanding-node-js-architecture-5fb32879b994>

mdn. (2024, November 22). Express/Node introduction - Learn web development | MDN. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\_Nodejs/Introduction

Google. (n.d.-b). Quickstart: Transcribe speech to text by using the Google Cloud console. Google Cloud. <https://cloud.google.com/speech-to-text/docs/transcribe-console>

Karabiber, F. (n.d.). TF-IDF — Term Frequency-Inverse Document Frequency – LearnDataSci. https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/#:~:text=Term%20Frequency%20%2D%20Inverse%20Document%20Frequency%20(TF%2DIDF)%20is,%2C%20relative%20to%20a%20corpus).

Growth Acceleration Partners. (2024, November 28). Leveraging Off-The-Shelf AI Models Using Hugging Face’s Transformers Library. Medium. <https://medium.com/@wearegap/leveraging-off-the-shelf-ai-models-using-hugging-faces-transformers-library-2237a08a4085>

Pi. (2024, November 17). Sentence Similarity and Semantic Search using free Huggingface Embedding API. Medium. <https://medium.com/neural-engineer/sentence-similarity-and-semantic-search-d6995c5e368a>

Streamlit. (n.d.). Understanding Streamlit’s client-server architecture - Streamlit Docs. <https://docs.streamlit.io/develop/concepts/architecture/architecture>

Oracle. (n.d.). MySQL :: MySQL 8.4 Reference Manual :: 18.11 Overview of MySQL Storage Engine Architecture. <https://dev.mysql.com/doc/refman/8.4/en/pluggable-storage-overview.html>

Express. (n.d.). Using Express middleware. https://expressjs.com/en/guide/using-middleware.html

Campbell, D. (2024, January 11). Nodejs WebSocket Server & Programming Examples. PubNub. <https://www.pubnub.com/blog/nodejs-websocket-programming-examples/>

insightsoftware. (2024, September 3). 5 Tops Benefits of Using a 3-Tier Architecture. Insightsoftware. https://insightsoftware.com/blog/5-benefits-of-a-3-tier-architecture/

## Originality report

