

# **Q & Ace**

*"Redefining Interviews with Intelligence."*

Project Team

Abdullah Jamil 22i-2435  
Aziq Rauf 22i-2367  
Shamil Sajjad 22i-2451

Session 2022-2026

Supervised by

Dr. Behjat Zuhaira



**Department of Software Engineering**

**National University of Computer and Emerging Sciences  
Islamabad, Pakistan**

**June, 2026**

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Problem Statement . . . . .	1
1.2	Scope . . . . .	2
1.3	Modules . . . . .	2
1.3.1	Client Web App . . . . .	2
1.3.1.1	Module 1: Facial Emotion Detection . . . . .	2
1.3.1.2	Module 2: NLP-Based Answer Evaluation . . . . .	3
1.3.2	Client Mobile App . . . . .	3
1.3.2.1	Module 3: Speech Emotion and Tone Analysis . . . . .	3
1.3.2.2	Module 4: Preparation and Practice Module . . . . .	3
1.3.3	Admin Web App . . . . .	3
1.3.3.1	Module 5: Feedback and Progress Tracking . . . . .	3
1.3.3.2	Module 6: System Management and Updates . . . . .	4
1.4	User Classes and Characteristics . . . . .	4
<b>2</b>	<b>Project Requirements</b>	<b>5</b>
2.1	Use-case . . . . .	5
2.1.0.1	Job Seeker . . . . .	6
2.1.0.2	System Administrator . . . . .	7
2.1.0.3	Functional Requirements . . . . .	7
2.1.1	Non-Functional Requirements . . . . .	8
2.1.1.1	Reliability . . . . .	8
2.1.1.2	Usability . . . . .	8
2.1.1.3	Performance . . . . .	8
2.1.1.4	Security . . . . .	9
2.1.2	Domain Model . . . . .	35
<b>3</b>	<b>System Overview</b>	<b>37</b>
3.1	Architectural Design . . . . .	37
3.2	Design Models . . . . .	39
3.3	Data Design . . . . .	53
	<b>References</b>	<b>59</b>

# List of Figures

2.1	Use Case Diagram of the Proposed System . . . . .	6
2.2	Domain Model of the Interview Coaching Platform. . . . .	36
3.1	Architectural Diagram of the Q&Ace System. . . . .	39
3.2	Activity Diagram: Create Profile . . . . .	40
3.3	Activity Diagram: Select Domain . . . . .	41
3.4	Activity Diagram: Practice Interview . . . . .	42
3.5	Activity Diagram: View Recommended Jobs / Preparation Module . . . .	43
3.6	Activity Diagram: View Report / Progress Tracking . . . . .	44
3.7	Activity Diagram: Generate Feedback Report . . . . .	45
3.8	Activity Diagram: Manage Interview Question Bank . . . . .	45
3.9	Activity Diagram: Monitor System Performance & Logs . . . . .	46
3.10	DFD Level 0 (Context Diagram) for the Q&Ace System. . . . .	47
3.11	DFD Level 1 showing the main processes of the system. . . . .	49
3.12	DFD Level 2 for the Conduct Interview Session Process. . . . .	50
3.13	DFD Level 2 for the Analyze User Performance Process. . . . .	51
3.14	DFD Level 2 for the Generate Feedback and Recommendations Process. .	52

# List of Tables

1.1	User Classes and Their Characteristics . . . . .	4
2.1	UC-01-01: Practise Interview . . . . .	9
2.2	UC-02-01: Analyze Facial Expressions . . . . .	11
2.3	UC-03-01: Analyze Speech Emotion and Tone . . . . .	13
2.4	UC-04-01: Evaluate Answer Content . . . . .	15
2.5	UC-05-01: Generate Report . . . . .	17
2.6	UC-06-01: Create Profile . . . . .	18
2.7	UC-07-01: Select Domain . . . . .	20
2.8	UC-08-01: View Report . . . . .	21
2.9	UC-09-01: Manage Interview Questions . . . . .	23
2.10	UC-10-01: Recommend Improvements . . . . .	25
2.11	UC-11-01: Take Quiz . . . . .	27
2.12	UC-12-01: Provide Resources . . . . .	29
2.13	UC-13-01: View Recommended Jobs . . . . .	30
2.14	UC-14-01: Record Response . . . . .	32
2.15	UC-15-01: Monitor System Performance . . . . .	34

# Chapter 1

## Introduction

In today's competitive job market, strong interview performance is crucial for career success. Many candidates struggle due to nervousness, insufficient practice, or a lack of feedback. While traditional mock interviews are effective, they can be resource-intensive and inaccessible. This project proposes "Q&Ace", an AI-Powered Career Coach and Personalized Interview Preparation Platform. This web-based system uses artificial intelligence to help users practice for job interviews. The platform will provide personalized and objective feedback by analyzing a candidate's facial expressions, tone of voice, and the content of their answers to improve their interview skills. The solution leverages recent advancements in computer vision, speech processing, and natural language processing to create a virtual interview coach that is available anytime, anywhere.

This document is intended for project managers, developers, testers, and documentation writers who will be involved in the creation and maintenance of the platform, as well as faculty advisors overseeing the project's development.

### 1.1 Problem Statement

Job seekers often lack awareness of how they appear or sound to an interviewer and may not recognize weaknesses in their answers. It is difficult to self-evaluate key performance aspects such as confident body language, appropriate tone of voice, and well-structured, relevant answers. There is a lack of affordable tools for comprehensive interview practice. Candidates often receive generic advice or limited feedback from peers, which is seldom personalized or multi-faceted. The core problem is the absence of an accessible platform that can simulate a real interview environment and provide structured feedback across facial expressions, vocal delivery, and answer quality. Without such targeted feedback, candidates may repeat mistakes, leading to missed job opportunities. Therefore, a solution is needed to help individuals identify and improve their interview weaknesses through

personalized coaching.

## 1.2 Scope

The project's scope is to design and implement an AI-powered mock interview coaching platform that offers a comprehensive practice environment. The system will analyze three dimensions of interview performance: facial expressions, speech tone, and response content. It will be a web-based platform where users can log in, receive interview questions, record their video and audio responses, and get detailed feedback. Key functionalities include detecting emotions and confidence via facial analysis, identifying nervousness or clarity through vocal analysis, and evaluating answer structure and relevance using NLP. The system will also feature adaptive questioning and stress simulation, such as timed responses and difficult follow-ups, to mimic real-world scenarios. Users will have access to progress tracking and personalized feedback reports to help them improve continuously. The project is bounded by these core functionalities; mobile deployment and large-scale integrations are considered future scope. The goal is to create a scalable, accessible, and intelligent tool that addresses the need for multi-faceted interview preparation solutions.

## 1.3 Modules

Write down the modules of the proposed project. Each module should highlight features, using bulleted/numbered notation. When developing both a mobile app and a web app, group the modules according to the system types, such as, Client Web App, Client Mobile App, Admin Web App etc.

### 1.3.1 Client Web App

#### 1.3.1.1 Module 1: Facial Emotion Detection

This module processes the user's webcam feed to detect non-verbal cues and facial expressions that reflect emotions such as confidence, nervousness, or enthusiasm. [2]

1. Detect emotions like happiness, anxiety, or neutrality.
2. Track eye contact and facial expressions during responses.
3. Provide visual feedback on engagement and confidence.

### **1.3.1.2 Module 2: NLP-Based Answer Evaluation**

This module evaluates the relevance and structure of user answers using NLP models.

1. Analyze text for clarity, completeness, and coherence.
2. Identify missing keywords or vague statements.
3. Suggest improvements for logical flow and precision.

## **1.3.2 Client Mobile App**

### **1.3.2.1 Module 3: Speech Emotion and Tone Analysis**

This module analyzes the user's voice to evaluate tone, clarity, and pacing during interview practice.

1. Detect tone, pitch, and speaking speed.
2. Identify filler words, hesitations, and nervous pacing.
3. Provide feedback on vocal clarity and enthusiasm.

### **1.3.2.2 Module 4: Preparation and Practice Module**

This module offers guided interview preparation through lessons and mock sessions.

1. Provide study material and question banks.
2. Offer timed mock interviews with simulated stress.
3. Track progress and suggest personalized practice.

## **1.3.3 Admin Web App**

### **1.3.3.1 Module 5: Feedback and Progress Tracking**

This module compiles analytics from user sessions and generates detailed performance reports.

1. Summarize user progress and confidence metrics.
2. Highlight strengths and improvement areas.
3. Export reports for review and system training.

**1.3.3.2 Module 6: System Management and Updates**

This module enables administrators to manage platform data and maintain system performance.

- 1. Manage user profiles and interview datasets.
- 2. Update question banks and AI evaluation models.
- 3. Monitor system analytics and performance metrics.

**1.4 User Classes and Characteristics**

The primary users of this product are individuals preparing for job interviews. The table below identifies and describes the pertinent characteristics of these user classes.

Table 1.1: User Classes and Their Characteristics	
User class	Description
Students and Job Seekers	The main end-users who will use the platform to prepare for interviews, practice answering questions, and receive personalized feedback
Fresh Graduates/ Professionals	Individuals preparing for competitive job interviews who benefit from realistic practice and performance evaluation



# Chapter 2

## Project Requirements

This chapter specifies the functional and non-functional requirements for the Q&Ace system. These requirements were determined using a Use Case modeling approach to clearly outline all interactions between the system and its users. They serve as the foundational blueprint guiding the project's design, development, and testing.

The chapter first presents the Use Case model, followed by a detailed breakdown of:

- **Functional Requirements:** Define *what* the system must do, detailing its specific features and behaviors.
- **Non-Functional Requirements:** Define *how well* the system must perform, covering critical quality attributes like performance, usability, and security.

### 2.1 Use-case

Given that the Q&Ace platform is an interactive end-user application, a **Use Case** modeling approach has been selected as the most effective technique for requirement gathering. This method allows for a clear visualization of how different actors interact with the system to achieve their goals. The model is composed of a detailed use case diagram, which provides a high-level overview, and descriptive text for each use case, which clarifies the specific steps and conditions for each interaction.

#### Overview of the Use Case Diagram

The use case diagram in Figure 2.1 illustrates the primary functionalities of the AI-Powered Interview Coaching platform. It defines the boundaries of the system and shows the interactions for the two main actors: the **Job Seeker**, who is the primary user, and the

**System Administrator**, who manages the platform's backend. The diagram outlines the key services the system provides and the relationships between different functionalities.

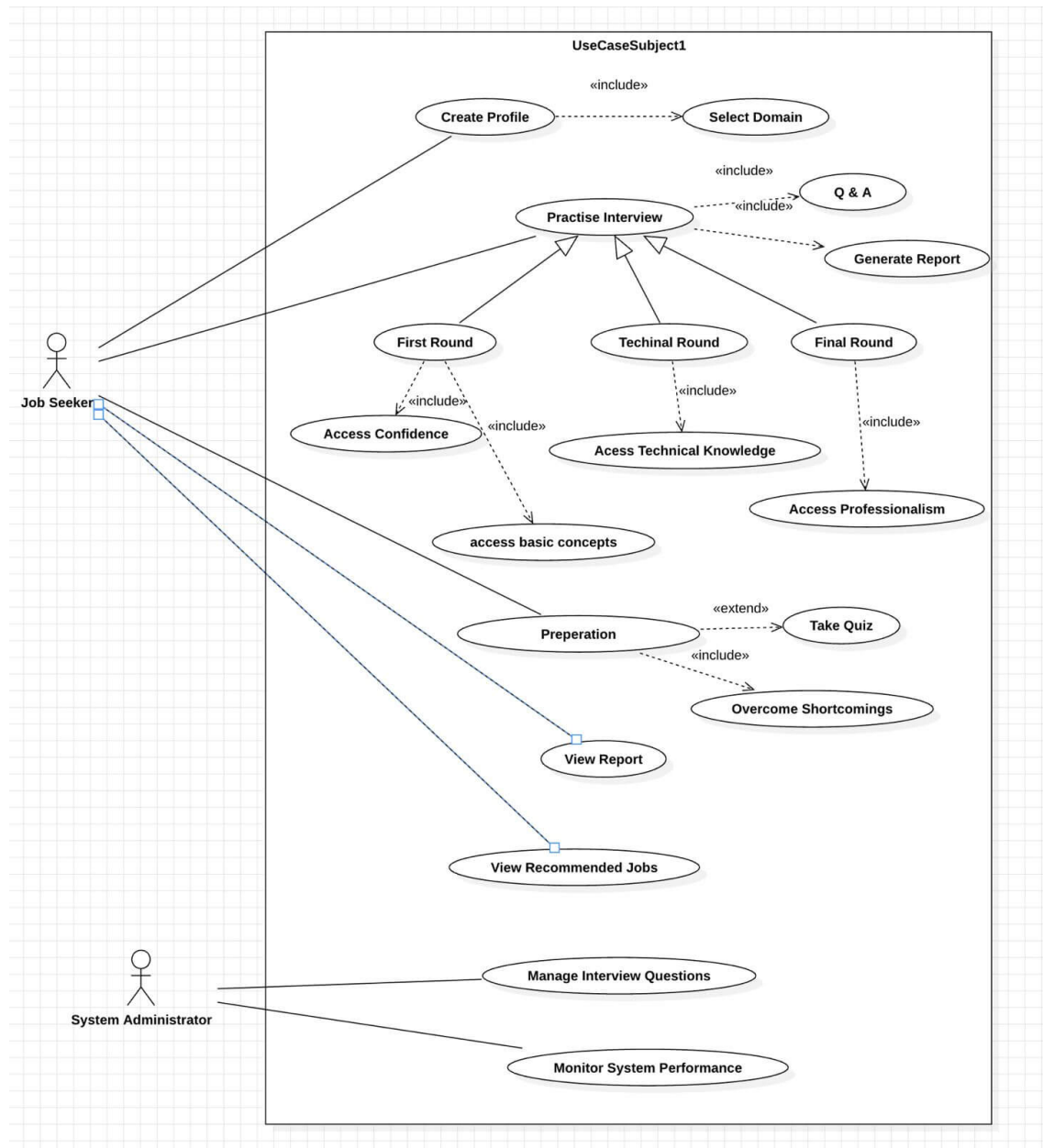


Figure 2.1: Use Case Diagram of the Proposed System

### 2.1.0.1 Job Seeker

The **Job Seeker** is the primary user, focused on interview preparation and performance review. Key interactions include:

- **Setup:** The user must first **Create Profile**, a process that requires them to **Select a Domain** (e.g., Software Engineering) to tailor the experience.
- **Practice:** The central **Practice Interview** use case has three specialized types: a **First Round** (assessing confidence), a **Technical Round** (assessing knowledge), and a **Final Round** (assessing professionalism). Every session involves a **Q&A** and concludes with a mandatory **Generated Report**.
- **Improvement:** The **Preparation** module helps users **Overcome Shortcomings** with targeted resources and access **Basic Concepts**. Optionally, users can extend this by choosing to **Take a Quiz**.
- **Review:** After a session, the user can **View their Report** for detailed analysis and **View Recommended Jobs** tailored to their profile.

### 2.1.0.2 System Administrator

The **System Administrator** is a backend user who maintains the platform's content and health. Their responsibilities include **Managing Interview Questions** (adding, updating, or deleting them) and **Monitoring System Performance** to ensure all services are running optimally.

### 2.1.0.3 Functional Requirements

The functional requirements define the specific behaviors of the Q&Ace system, organized by module.

- **Facial Emotion Detection:** The system shall analyze the user's video feed to detect emotions, track eye contact and head posture to infer confidence, and log micro-expressions like smiling or frowning.
- **Speech Emotion and Tone Analysis:** The system shall analyze audio to measure vocal tone, pitch, and pace. It will also quantify filler words (e.g., "um," "ah") and generate scores for vocal clarity and enthusiasm.
- **NLP-Based Answer Evaluation:** The system shall transcribe spoken answers to text, evaluating them for logical flow, grammar, and relevance by identifying domain-specific keywords. It will also suggest improvements for vague or incomplete answers.
- **Adaptive Questioning & Stress Simulation:** The system shall generate dynamic follow-up questions based on the user's previous answers. It can also simulate stress through time limits or interruptions to train users to perform under pressure.

- **Feedback and Progress Tracking:** The system shall generate a comprehensive visual feedback report after each session. It will also track performance metrics over time with charts and provide targeted recommendations for improvement.
- **Preparation Module:** The system shall provide a resource library with articles, video tutorials, and a question bank. It will also offer interactive exercises for specific skills and allow users to create personalized preparation plans.

### 2.1.1 Non-Functional Requirements

This section specifies the non-functional requirements that define the system's quality attributes, ensuring it is reliable, usable, and secure.

#### 2.1.1.1 Reliability

The system must be consistently available and robust to ensure that users can complete their practice sessions without interruption or loss of data.

- The system shall maintain an uptime of at least 99.5% during operational hours.
- In the event of a server-side failure during the analysis phase, the system shall safely store the user's recorded data, allowing the user to re-initiate the analysis at a later time without needing to re-record their response.

#### 2.1.1.2 Usability

The platform must be intuitive, user-friendly, and accessible to individuals with varying levels of technical proficiency.

- **USE-1:** A new user shall be able to complete the registration process, log in, and start their first mock interview session in less than three minutes, with minimal on-screen instructions.
- **USE-2:** The feedback report shall use clear, intuitive visualizations, such as gauges, charts, and color-coded indicators, to present complex performance data in an easy-to-understand format.

#### 2.1.1.3 Performance

The system must process user responses and generate feedback in a timely manner to maintain an interactive and engaging user experience.

- **PER-1:** For a recorded user response of up to two minutes in length, the complete multi-modal analysis (facial, speech, and NLP) and the generation of the feedback report shall be completed and displayed to the user within 60 seconds.

#### 2.1.1.4 Security

As the system handles sensitive personal data, including video and audio recordings, robust security measures must be in place to protect user privacy.

- All user data, especially video and audio files, shall be encrypted both in transit (using TLS/SSL) and at rest.
- Access to a user's recorded sessions and feedback reports shall be strictly restricted to that user, enforced through a secure authentication and authorization mechanism.

## Extended Use Cases

This section details the primary use cases for the Q&Ace system. The following tables outline the actors, triggers, scenarios, and quality attributes for each core functionality.

Table 2.1: UC-01-01: Practise Interview

Section	Content
<b>Designation</b>	UC-01-01
<b>Name</b>	Practise Interview
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: high
<b>Criticality</b>	High
<b>Source</b>	Job seeker requirements, Career coaching best practices
<b>Responsible</b>	Muhammad Aziz Rauf
<b>Description</b>	The job seeker initiates a mock interview session. The system records video and audio of their responses, analyzes facial expressions, voice tone, and answer content using AI models, and provides personalized feedback to improve interview performance.
<b>Trigger event</b>	The job seeker selects "Start Interview" after choosing their domain and interview round type.
<b>Actors</b>	Job Seeker, Facial Emotion Detection Model, Speech Emotion Analysis Model, NLP Answer Evaluation Model

**Table 2.1 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Pre-condition</b>	<ol style="list-style-type: none"><li>1. Job seeker has created a profile and selected a domain.</li><li>2. Camera and microphone permissions are granted.</li><li>3. AI models are loaded and ready.</li></ol>
<b>Post-condition</b>	<ol style="list-style-type: none"><li>1. Interview session is recorded and stored.</li><li>2. Multi-modal analysis is completed.</li><li>3. Feedback report is generated and available to view.</li></ol>
<b>Result</b>	Comprehensive feedback on facial expressions, voice tone, and answer quality.
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system presents an interview question based on selected domain and round type.</li><li>2. The job seeker reads the question.</li><li>3. The system begins recording video (webcam) and audio (microphone).</li><li>4. The job seeker answers the question verbally.</li><li>5. The system stops recording when the job seeker indicates completion.</li><li>6. The facial emotion detection model analyzes recorded video frames.</li><li>7. The speech emotion analysis model processes audio features (MFCCs).</li><li>8. The system transcribes the audio to text.</li><li>9. The NLP model evaluates answer content for relevance and structure.</li><li>10. The system compiles feedback from all three analyses.</li><li>11. The system displays a feedback report with scores and improvement suggestions.</li><li>12. The job seeker can proceed to the next question or end the session.</li></ol>

**Table 2.1 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Alternative scenarios</b>	<p>3a. Recording fails due to hardware issues.</p> <p>3a1. The system displays an error message: "Unable to access camera/microphone."</p> <p>3a2. The system prompts the job seeker to check permissions and retry.</p> <p>8a. Automatic speech recognition fails or is unavailable.</p> <p>8a1. The system prompts the job seeker to manually type their answer.</p> <p>8a2. The system proceeds with NLP analysis on typed text.</p> <p>9a. Answer is too short (less than 10 words).</p> <p>9a1. The system flags the answer as incomplete.</p> <p>9a2. The system suggests providing more detailed responses.</p>
<b>Exception scenarios</b>	<p>Trigger event: No face detected in video frames.</p> <p>→ The system notifies: "Please position your face within the camera frame" and pauses analysis until face is detected.</p> <p>Trigger event: Audio quality is too poor (excessive noise).</p> <p>→ The system warns about background noise and suggests retrying in a quieter environment.</p> <p>Trigger event: AI model inference fails (technical error).</p> <p>→ The system logs the error, saves the recording, and notifies the user that analysis will be completed later.</p>
<b>Qualities</b>	<p>→ QR.01 (system response time: feedback within 5 seconds)</p> <p>→ QR.02 (model accuracy: minimum 60% for emotion detection)</p> <p>→ QR.03 (usability: clear, actionable feedback)</p>

**Table 2.2: UC-02-01: Analyze Facial Expressions**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-02-01
<b>Name</b>	Analyze Facial Expressions
<b>Authors</b>	Muhammad Aزيق Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: high
<b>Criticality</b>	High
<b>Source</b>	Computer vision best practices, FER-2013 dataset research

**Table 2.2 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The system uses a custom-trained CNN model to detect emotions from the job seeker's facial expressions during interview responses. It identifies emotions such as happiness, sadness, fear, anger, surprise, disgust, and neutrality to assess confidence levels.
<b>Trigger event</b>	Video recording of interview response is completed.
<b>Actors</b>	Facial Emotion Detection Model (CNN), OpenCV Face Detector
<b>Pre-condition</b>	<ol style="list-style-type: none"><li>1. Video frames are captured and stored.</li><li>2. CNN model is loaded (trained on FER-2013 dataset).</li><li>3. Face detection module is active.</li></ol>
<b>Post-condition</b>	<ol style="list-style-type: none"><li>1. Dominant emotions are identified with confidence scores.</li><li>2. Emotion analysis is stored for report generation.</li><li>3. Confidence interpretation is generated (e.g., nervous, calm, confident).</li></ol>
<b>Result</b>	Emotion classification and confidence level assessment.
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system extracts frames from recorded video (1 frame per second).</li><li>2. OpenCV detects face regions in each frame using Haar cascades.</li><li>3. Detected face regions are cropped and resized to 48x48 pixels.</li><li>4. Images are converted to grayscale and normalized.</li><li>5. The CNN model processes each face image.</li><li>6. The model outputs probability scores for 7 emotions.</li><li>7. The system identifies the dominant emotion per frame.</li><li>8. The system calculates average emotion distribution across all frames.</li><li>9. The system interprets emotions in interview context (High fear/sadness = nervous; High happiness/neutral = confident).</li><li>10. The system stores emotion analysis results.</li></ol>



**Table 2.2 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Alternative scenarios</b>	<p>2a. Multiple faces detected in the frame.</p> <p>2a1. The system selects the largest face (assumed to be the interviewee).</p> <p>2a2. The system proceeds with analysis on that face.</p> <p>2b. Face detection confidence is low.</p> <p>2b1. The system skips that frame and moves to the next.</p> <p>2b2. If 80% of frames have no detectable face, the system flags insufficient data.</p>
<b>Exception scenarios</b>	<p>Trigger event: No face detected in any frame.</p> <p>→ The system reports "Facial analysis unavailable - please ensure your face is visible in the camera" and skips facial emotion components.</p> <p>Trigger event: CNN model inference error.</p> <p>→ The system logs the error, marks facial analysis as failed, and continues with other analyses.</p>
<b>Qualities</b>	<p>→ QR.04 (inference speed: less than 2 seconds for 30 frames)</p> <p>→ QR.05 (accuracy: greater than or equal to 60% on validation set)</p> <p>→ QR.06 (robustness: handle varied lighting conditions)</p>

**Table 2.3: UC-03-01: Analyze Speech Emotion and Tone**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-03-01
<b>Name</b>	Analyze Speech Emotion and Tone
<b>Authors</b>	Muhammad Aزيق Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: high
<b>Criticality</b>	High
<b>Source</b>	Audio processing research, RAVDESS dataset, Speech emotion recognition studies
<b>Responsible</b>	Abdullah Jamil
<b>Description</b>	The system analyzes the audio recording of the job seeker's response to detect vocal emotions, assess tone quality, and identify speech patterns such as pauses, filler words, and speaking rate. This provides insights into the candidate's confidence and communication clarity.

**Table 2.3 – continued from previous page**

Section	Content
<b>Trigger event</b>	Audio recording of interview response is completed.
<b>Actors</b>	Speech Emotion Analysis Model (RNN/CNN), LibROSA Audio Processor
<b>Pre-condition</b>	<ol style="list-style-type: none"><li>1. The audio file is captured and stored.</li><li>2. Speech emotion model is loaded (trained on RAVDESS dataset).</li><li>3. Audio processing libraries are initialized.</li></ol>
<b>Post-condition</b>	<ol style="list-style-type: none"><li>1. Vocal emotion is classified with confidence scores.</li><li>2. Speech quality metrics are calculated (pace, pauses, filler words).</li><li>3. Vocal analysis results are stored for report generation.</li></ol>
<b>Result</b>	Vocal emotion classification and speech quality assessment.
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system loads the audio file using LibROSA.</li><li>2. The system normalizes audio volume levels.</li><li>3. The system extracts MFCC features from the audio signal.</li><li>4. The MFCC features are fed into the trained RNN/CNN model.</li><li>5. The model outputs probability scores for vocal emotions.</li><li>6. The system identifies the dominant vocal emotion.</li><li>7. The system detects silence periods to count long pauses.</li><li>8. The system calculates speaking rate (words per minute).</li><li>9. The system uses speech-to-text to identify filler words ("um", "uh", "like").</li><li>10. The system interprets results.</li><li>11. The system stores vocal analysis results.</li></ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"><li>2a. Audio quality is poor (high background noise).<ol style="list-style-type: none"><li>2a1. The system applies noise reduction filters.</li><li>2a2. The system flags low confidence in analysis results.</li></ol></li><li>9a. Speech-to-text service is unavailable.<ol style="list-style-type: none"><li>9a1. The system skips filler word detection.</li><li>9a2. The system proceeds with emotion and pace analysis only.</li></ol></li></ol>

**Table 2.3 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Exception scenarios</b>	<p>Trigger event: Audio file is corrupted or unreadable.  → The system reports "Audio analysis failed - please retry the interview" and marks vocal analysis as unavailable.</p> <p>Trigger event: Audio is completely silent (no speech detected).  → The system notifies "No speech detected in audio" and suggests checking microphone settings.</p>
<b>Qualities</b>	<p>→ QR.07 (processing time: less than 3 seconds for 60-second audio)</p> <p>→ QR.08 (emotion accuracy: greater than or equal to 65% on validation set)</p> <p>→ QR.09 (noise tolerance: handle moderate background noise)</p>

**Table 2.4: UC-04-01: Evaluate Answer Content**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-04-01
<b>Name</b>	Evaluate Answer Content
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: medium
<b>Criticality</b>	High
<b>Source</b>	Interview best practices, NLP research, Custom Q&A dataset
<b>Responsible</b>	Muhammad Aziq Rauf
<b>Description</b>	The system evaluates the textual content of the job seeker's answer using a custom NLP model. It assesses relevance to the question, completeness of response, structural quality, and presence of key points to provide feedback on answer effectiveness.
<b>Trigger event</b>	Transcribed text of the interview answer is available.
<b>Actors</b>	NLP Answer Evaluation Model, Text Preprocessing Module
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Answer text is obtained.</li> <li>2. NLP model is loaded.</li> <li>3. Question context is available for comparison.</li> </ol>

**Table 2.4 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. Answer quality is classified.</li> <li>2. Specific feedback on content weaknesses is generated.</li> <li>3. Content evaluation results are stored.</li> </ol>
<b>Result</b>	Answer quality score and improvement recommendations.
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system receives the transcribed answer text and original question.</li> <li>2. The system tokenizes and preprocesses the text.</li> <li>3. The system extracts features.</li> <li>4. The system concatenates question and answer representations.</li> <li>5. The NLP model (classifier) processes the combined input.</li> <li>6. The model outputs a quality label (Good/Average/Poor).</li> <li>7. The system checks answer length.</li> <li>8. The system identifies if key expected points are mentioned.</li> <li>9. The system checks for STAR method structure in behavioral questions.</li> <li>10. The system generates specific feedback.</li> <li>11. The system stores content evaluation results.</li> </ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"> <li>6a. Model confidence is low (below 50%). <ol style="list-style-type: none"> <li>6a1. The system labels the answer as "Average" by default.</li> <li>6a2. The system provides generic improvement suggestions.</li> </ol> </li> <li>7a. Answer length is less than 20 words. <ol style="list-style-type: none"> <li>7a1. The system automatically flags as "Poor - Too brief".</li> <li>7a2. The system suggests providing more detail and examples.</li> </ol> </li> </ol>
<b>Exception scenarios</b>	<p>Trigger event: Answer text is empty.</p> <p>→ The system reports "No valid answer detected" and marks content evaluation as failed.</p> <p>Trigger event: NLP model fails to load or process input.</p> <p>→ The system uses rule-based evaluation (length, keywords only) and logs the error.</p>
<b>Qualities</b>	<p>→ QR.10 (processing time: less than 1 second per answer)</p> <p>→ QR.11 (classification accuracy: greater than or equal to 70% on test set)</p> <p>→ QR.12 (feedback clarity: actionable and specific)</p>

Table 2.5: UC-05-01: Generate Report

Section	Content
<b>Designation</b>	UC-05-01
<b>Name</b>	Generate Report
<b>Authors</b>	Muhammad Aزيق Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: low
<b>Criticality</b>	High
<b>Source</b>	User feedback requirements, Reporting best practices
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The system compiles results from facial emotion analysis, speech emotion analysis, and answer content evaluation into a comprehensive feedback report. The report highlights strengths, identifies areas for improvement, and provides actionable recommendations.
<b>Trigger event</b>	All three AI analyses (facial, vocal, content) are completed.
<b>Actors</b>	Report Generation Module, Job Seeker
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Facial emotion analysis is complete.</li> <li>2. Speech emotion analysis is complete.</li> <li>3. Answer content evaluation is complete.</li> <li>4. Interview session data is stored.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. A comprehensive feedback report is generated.</li> <li>2. The report is stored in the database.</li> <li>3. The report is displayed to the job seeker.</li> </ol>
<b>Result</b>	Detailed interview performance report with multi-modal feedback.
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system retrieves facial emotion analysis results.</li> <li>2. The system retrieves speech emotion analysis results.</li> <li>3. The system retrieves answer content evaluation results.</li> <li>4. The system calculates an overall confidence score.</li> <li>5. The system formats feedback for each dimension.</li> <li>6. The system identifies the weakest area.</li> <li>7. The system generates prioritized improvement recommendations.</li> <li>8. The system creates a visual summary.</li> <li>9. The system displays the report on the interface.</li> <li>10. The system saves the report to the user's history.</li> </ol>

**Table 2.5 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Alternative scenarios</b>	<p>1a. Facial analysis was unavailable.</p> <p>1a1. The system omits facial feedback section.</p> <p>1a2. The system calculates overall score using only vocal and content analyses.</p> <p>2a. Speech analysis was unavailable.</p> <p>2a1. The system omits vocal feedback section.</p> <p>2a2. The system adjusts overall score calculation accordingly.</p>
<b>Exception scenarios</b>	<p>Trigger event: All three analyses failed.</p> <p>→ The system displays "Report generation failed - please retry the interview" and logs the error.</p> <p>Trigger event: Database storage fails.</p> <p>→ The system still displays the report to the user but notifies "Report may not be saved - please screenshot for reference."</p>
<b>Qualities</b>	<p>→ QR.13 (generation time: less than 2 seconds)</p> <p>→ QR.14 (clarity: feedback is understandable)</p> <p>→ QR.15 (actionability: specific improvement suggestions provided)</p>

**Table 2.6: UC-06-01: Create Profile**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-06-01
<b>Name</b>	Create Profile
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: low
<b>Criticality</b>	Medium
<b>Source</b>	User onboarding requirements, Personalization needs
<b>Responsible</b>	Abdullah Jamil
<b>Description</b>	The job seeker creates a profile by providing personal information and selecting their professional domain. This profile is used to tailor interview questions and track progress over time.
<b>Trigger event</b>	The job seeker accesses the system for the first time or selects "Create Profile".
<b>Actors</b>	Job Seeker, Profile Management Module

**Table 2.6 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. The job seeker has access to the system.</li> <li>2. Database is accessible for storing profile data.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. Profile is created and stored in the database.</li> <li>2. Job seeker is redirected to domain selection.</li> <li>3. Job seeker can now access interview practice features.</li> </ol>
<b>Result</b>	User profile with selected domain.
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system displays the profile creation form.</li> <li>2. The job seeker enters their name.</li> <li>3. The job seeker enters their email address.</li> <li>4. The job seeker enters their phone number (optional).</li> <li>5. The job seeker provides their current experience level.</li> <li>6. The system validates all required fields are filled.</li> <li>7. The system checks if email is in valid format.</li> <li>8. The system saves the profile to the database.</li> <li>9. The system displays "Profile created successfully".</li> <li>10. The system redirects to the "Select Domain" page.</li> </ol>
<b>Alternative scenarios</b>	<p>7a. Email format is invalid.</p> <p>7a1. The system displays error: "Please enter a valid email address".</p> <p>7a2. The job seeker corrects the email and resubmits.</p> <p>8a. Email already exists in the database.</p> <p>8a1. The system displays: "An account with this email already exists".</p> <p>8a2. The system offers options: "Login" or "Use different email".</p>
<b>Exception scenarios</b>	<p>Trigger event: Database connection fails during save.</p> <p>→ The system displays "Unable to create profile - please try again later" and logs the error.</p> <p>Trigger event: Required field is left empty.</p> <p>→ The system highlights the empty field and displays "This field is required" without submitting.</p>
<b>Qualities</b>	<p>→ QR.16 (response time: profile creation within 1 second)</p> <p>→ QR.17 (data validation: email format, required fields)</p> <p>→ QR.18 (security: passwords hashed if authentication added)</p>

Table 2.7: UC-07-01: Select Domain

Section	Content
<b>Designation</b>	UC-07-01
<b>Name</b>	Select Domain
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: low
<b>Criticality</b>	Medium
<b>Source</b>	Personalization requirements, Question database organization
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The job seeker selects their professional domain (e.g., Software Engineering, Marketing, Finance, HR) to receive relevant interview questions tailored to their field.
<b>Trigger event</b>	Profile creation is completed or user selects "Change Domain" from settings.
<b>Actors</b>	Job Seeker, Domain Management Module
<b>Pre-condition</b>	<ol style="list-style-type: none"><li>1. Job seeker profile exists.</li><li>2. Domain options are loaded from the database.</li></ol>
<b>Post-condition</b>	<ol style="list-style-type: none"><li>1. Selected domain is associated with the user profile.</li><li>2. Interview questions are filtered based on the selected domain.</li><li>3. User can proceed to interview practice.</li></ol>
<b>Result</b>	Domain preference saved to user profile.
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system displays a list of available domains.</li><li>2. The job seeker reviews the domain options.</li><li>3. The job seeker selects their primary domain of interest.</li><li>4. The system confirms the selection with a visual indicator.</li><li>5. The system saves the domain preference to the user's profile.</li><li>6. The system displays "Domain saved successfully".</li><li>7. The system loads interview questions relevant to the selected domain.</li><li>8. The system redirects to the interview practice dashboard.</li></ol>



**Table 2.7 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Alternative scenarios</b>	<p>3a. Job seeker wants to select multiple domains.</p> <p>3a1. The system allows selection of up to 3 domains.</p> <p>3a2. Interview questions will be drawn from all selected domains.</p> <p>7a. No questions are available for the selected domain.</p> <p>7a1. The system notifies: "Questions for this domain are being added soon".</p> <p>7a2. The system suggests selecting an alternative domain.</p>
<b>Exception scenarios</b>	<p>Trigger event: Domain list fails to load from database.</p> <p>→ The system displays "Unable to load domains - please refresh the page" and provides a manual retry button.</p> <p>Trigger event: Save operation fails.</p> <p>→ The system displays "Domain selection not saved - please try again" and allows the user to resubmit.</p>
<b>Qualities</b>	<p>→ QR.19 (load time: domain list appears within 1 second)</p> <p>→ QR.20 (flexibility: allow domain changes anytime)</p> <p>→ QR.21 (coverage: minimum 8 professional domains available)</p>

**Table 2.8: UC-08-01: View Report**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-08-01
<b>Name</b>	View Report
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: medium; Technological risk: low
<b>Criticality</b>	Medium
<b>Source</b>	User feedback requirements, Progress tracking needs
<b>Responsible</b>	Abdullah Jamil
<b>Description</b>	The job seeker views previously generated feedback reports from past interview practice sessions. This allows them to track progress over time and review improvement suggestions.
<b>Trigger event</b>	The job seeker selects "View Reports" or "Report History" from the dashboard.
<b>Actors</b>	Job Seeker, Report Retrieval Module

**Table 2.8 – continued from previous page**

Section	Content
<b>Pre-condition</b>	1. Job seeker has completed at least one interview practice session. 2. Reports are stored in the database.
<b>Post-condition</b>	1. List of past reports is displayed. 2. Job seeker can select and view detailed feedback from any session. 3. Job seeker can compare performance across sessions.
<b>Result</b>	Historical interview performance data displayed.
<b>Main scenario</b>	1. The system retrieves all reports associated with the job seeker's profile from the database. 2. The system displays a list of reports sorted by date. 3. Each report entry shows: Date, Interview Round Type, Overall Score, Status. 4. The job seeker selects a specific report to view details. 5. The system loads and displays the full feedback report. 6. The system highlights improvements or declines compared to previous sessions. 7. The job seeker can navigate back to the report list or start a new interview.
<b>Alternative scenarios</b>	1a. No reports are available. 1a1. The system displays "No reports yet - complete an interview to get feedback". 1a2. The system provides a button: "Start Your First Interview". 4a. Report data is corrupted or incomplete. 4a1. The system displays "This report cannot be loaded - data may be incomplete". 4a2. The system offers to retry or contact support.
<b>Exception scenarios</b>	Trigger event: Database query fails. → The system displays "Unable to load reports - please try again later" and logs the error. Trigger event: Report file is missing from storage. → The system displays "Report not found" and offers to re-generate if session data exists.

**Table 2.8 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Qualities</b>	→ QR.22 (load time: report list appears within 2 seconds) → QR.23 (historical data: store reports for minimum 6 months) → QR.24 (comparison: show trend indicators)

**Table 2.9: UC-09-01: Manage Interview Questions**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-09-01
<b>Name</b>	Manage Interview Questions
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: medium; Technological risk: low
<b>Criticality</b>	Low
<b>Source</b>	System administration requirements, Content management needs
<b>Responsible</b>	Muhammad Aziz Rauf
<b>Description</b>	The system administrator adds, edits, or deletes interview questions in the database. Questions are categorized by domain, difficulty level, and interview round type to ensure relevant content for users.
<b>Trigger event</b>	The system administrator selects "Manage Questions" from the admin dashboard.
<b>Actors</b>	System Administrator, Question Management Module
<b>Pre-condition</b>	1. Administrator is logged in. 2. Admin dashboard is accessible. 3. Question database is connected.
<b>Post-condition</b>	1. Question database is updated. 2. Changes are immediately reflected. 3. Audit log records all changes made.
<b>Result</b>	Updated interview question database.

**Table 2.9 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system displays the question management interface.</li> <li>2. The system shows a list of all existing questions with filters.</li> <li>3. The administrator selects an action: Add New, Edit, or Delete.</li> </ol> <p>For Add New:</p> <ol style="list-style-type: none"> <li>4a. The system displays a form with fields.</li> <li>5a. The administrator fills in all required fields.</li> <li>6a. The system validates that all required fields are complete.</li> <li>7a. The system saves the new question to the database.</li> <li>8a. The system displays "Question added successfully".</li> </ol> <p>For Edit:</p> <ol style="list-style-type: none"> <li>4b. The administrator selects a question.</li> <li>5b. The system displays the question details in an editable form.</li> <li>6b. The administrator modifies the desired fields.</li> <li>7b. The system validates and saves changes.</li> <li>8b. The system displays "Question updated successfully".</li> </ol> <p>For Delete:</p> <ol style="list-style-type: none"> <li>4c. The administrator selects a question and clicks "Delete".</li> <li>5c. The system displays a confirmation dialog.</li> <li>6c. The administrator confirms deletion.</li> <li>7c. The system removes the question from the database.</li> <li>8c. The system displays "Question deleted successfully".</li> </ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"> <li>6a. Required field is left empty during add/edit. <ol style="list-style-type: none"> <li>6a1. The system highlights the empty field: "This field is required".</li> <li>6a2. The administrator completes the field and resubmits.</li> </ol> </li> <li>7a. Duplicate question detected. <ol style="list-style-type: none"> <li>7a1. The system warns: "A similar question already exists".</li> <li>7a2. The administrator can choose to add anyway or cancel.</li> </ol> </li> </ol>
<b>Exception scenarios</b>	<p>Trigger event: Database save operation fails.</p> <p>→ The system displays "Unable to save changes - please try again" and logs the error.</p> <p>Trigger event: Administrator attempts to delete a question currently in use.</p> <p>→ The system prevents deletion and notifies: "This question is currently active in user sessions - please try again later."</p>

**Table 2.9 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Qualities</b>	→ QR.25 (response time: operations complete within 1 second) → QR.26 (validation: prevent duplicate or empty questions) → QR.27 (audit trail: log all administrative changes)

**Table 2.10: UC-10-01: Recommend Improvements**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-10-01
<b>Name</b>	Recommend Improvements
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: medium; Technological risk: medium
<b>Criticality</b>	Medium
<b>Source</b>	User learning needs, Personalized coaching requirements
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The system analyzes the job seeker's performance across multiple interview sessions and generates personalized improvement recommendations. These suggestions target the weakest areas and provide actionable steps for enhancement.
<b>Trigger event</b>	Job seeker completes an interview session and views the report, or explicitly requests recommendations from the Preparation module.
<b>Actors</b>	Recommendation Engine, Job Seeker
<b>Pre-condition</b>	1. At least one interview session is completed. 2. Performance data is available. 3. Recommendation logic is implemented.
<b>Post-condition</b>	1. Personalized improvement recommendations are generated. 2. Recommendations are displayed to the user. 3. Recommended resources or practice areas are highlighted.
<b>Result</b>	Personalized action plan for interview improvement.

**Table 2.10 – continued from previous page**

Section	Content
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system retrieves performance data.</li><li>2. The system calculates scores for each dimension.</li><li>3. The system identifies the lowest-scoring dimension.</li><li>4. The system analyzes patterns across multiple sessions.</li><li>5. The system generates targeted recommendations based on identified weaknesses.</li><li>6. The system prioritizes recommendations by impact.</li><li>7. The system links each recommendation to relevant resources.</li><li>8. The system displays recommendations in an easy-to-read format.</li><li>9. The job seeker can mark recommendations as "Completed" or "In Progress".</li><li>10. The system tracks which recommendations have been acted upon.</li></ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"><li>4a. All dimensions have similar scores.<ol style="list-style-type: none"><li>4a1. The system provides general best practices.</li><li>4a2. The system suggests advanced techniques.</li></ol></li><li>1a. This is the first interview session.<ol style="list-style-type: none"><li>1a1. The system bases recommendations solely on current session results.</li><li>1a2. The system notes: "Complete more sessions for better personalized insights".</li></ol></li><li>3a. All scores are already high (above 80%).<ol style="list-style-type: none"><li>3a1. The system congratulates the user: "Excellent performance!".</li><li>3a2. The system suggests maintaining consistency.</li></ol></li></ol>
<b>Exception scenarios</b>	<p>Trigger event: Insufficient data to generate meaningful recommendations.</p> <p>→ The system displays "Complete a full interview session to receive personalized recommendations".</p> <p>Trigger event: Recommendation engine logic fails.</p> <p>→ The system displays generic improvement tips and logs the error.</p>

**Table 2.10 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Qualities</b>	<p>→ QR.28 (generation time: recommendations appear within 3 seconds)</p> <p>→ QR.29 (relevance: recommendations match actual weaknesses)</p> <p>→ QR.30 (actionability: each recommendation includes specific next steps)</p>

**Table 2.11: UC-11-01: Take Quiz**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-11-01
<b>Name</b>	Take Quiz
<b>Authors</b>	Muhammad Aziz Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: low; Technological risk: low
<b>Criticality</b>	Low
<b>Source</b>	Preparation module requirements, Self-assessment needs
<b>Responsible</b>	Abdullah Jamil
<b>Description</b>	The job seeker takes a knowledge quiz related to their domain or interview preparation topics. The quiz helps assess their understanding of key concepts and identifies knowledge gaps without the pressure of a full mock interview.
<b>Trigger event</b>	Job seeker selects "Take Quiz" from the Preparation module or from a recommended improvement action.
<b>Actors</b>	Job Seeker, Quiz Management Module
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Job seeker has selected a domain.</li> <li>2. Quiz questions are available.</li> <li>3. Quiz module is loaded.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. Quiz is completed and scored.</li> <li>2. Results are displayed with correct/incorrect answers.</li> <li>3. Knowledge gaps are identified and linked to learning resources.</li> <li>4. Quiz score is stored in user history.</li> </ol>
<b>Result</b>	Quiz score and knowledge gap analysis.

**Table 2.11 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Main scenario</b>	<ol style="list-style-type: none"><li>1. The system displays available quiz topics.</li><li>2. The job seeker selects a quiz topic.</li><li>3. The system loads 10-15 multiple-choice questions.</li><li>4. The system displays the first question with options.</li><li>5. The job seeker selects an answer.</li><li>6. The system records the selection and moves to the next question.</li><li>7. Steps 4-6 repeat until all questions are answered.</li><li>8. The system calculates the total score.</li><li>9. The system displays results.</li><li>10. The system shows which questions were answered correctly/incorrectly.</li><li>11. For incorrect answers, the system displays the correct answer with an explanation.</li><li>12. The system identifies knowledge gaps.</li><li>13. The system recommends resources to address identified gaps.</li><li>14. The job seeker can retake the quiz or return to the preparation module.</li></ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"><li>5a. Job seeker is unsure and wants to skip a question.<ol style="list-style-type: none"><li>5a1. The system allows marking the question for later review.</li><li>5a2. At the end, the system prompts to answer skipped questions.</li></ol></li><li>8a. Job seeker wants to exit quiz midway.<ol style="list-style-type: none"><li>8a1. The system asks to save progress.</li><li>8a2. If yes, the system saves answered questions.</li><li>8a3. If no, the system discards progress.</li></ol></li></ol>
<b>Exception scenarios</b>	<p>Trigger event: Quiz questions fail to load. → The system displays "Unable to load quiz - please try again later".</p> <p>Trigger event: Timer expires (if timed quiz). → The system auto-submits all answered questions.</p>



**Table 2.11 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Qualities</b>	→ QR.31 (question variety: minimum 50 questions per domain) → QR.32 (immediate feedback: results shown within 1 second) → QR.33 (educational value: explanations provided)

**Table 2.12: UC-12-01: Provide Resources**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-12-01
<b>Name</b>	Provide Resources
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: low; Technological risk: low
<b>Criticality</b>	Low
<b>Source</b>	Learning support requirements, Content library management
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The system provides curated learning resources (articles, videos, tips, example answers) to help job seekers prepare for interviews. Resources are categorized by topic and tailored to the user's domain and identified weaknesses.
<b>Trigger event</b>	Job seeker selects "Resources" from the Preparation module or clicks on a resource link from recommendations.
<b>Actors</b>	Job Seeker, Resource Management Module
<b>Pre-condition</b>	1. Resource database is populated. 2. Resources are categorized. 3. User profile and performance data are available.
<b>Post-condition</b>	1. Relevant resources are displayed. 2. Job seeker can access, bookmark, or mark resources as completed. 3. Resource usage is tracked.
<b>Result</b>	Curated learning materials accessible to user.

**Table 2.12 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system displays resource categories.</li> <li>2. The job seeker selects a category.</li> <li>3. The system retrieves resources from the database.</li> <li>4. The system displays a list of resources.</li> <li>5. The system highlights recommended resources based on performance.</li> <li>6. The job seeker selects a resource.</li> <li>7. The system displays the full content.</li> <li>8. The job seeker can bookmark the resource.</li> <li>9. The job seeker can mark the resource as "Completed".</li> <li>10. The system tracks which resources have been accessed.</li> </ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"> <li>3a. No resources are available for the selected category.               <ol style="list-style-type: none"> <li>3a1. The system displays "Resources for this category are being added soon".</li> <li>3a2. The system suggests exploring other categories.</li> </ol> </li> <li>6a. External video link is broken.               <ol style="list-style-type: none"> <li>6a1. The system displays "This resource is temporarily unavailable".</li> <li>6a2. The system offers alternative similar resources.</li> </ol> </li> </ol>
<b>Exception scenarios</b>	<p>Trigger event: Resource database query fails.          → The system displays "Unable to load resources - please try again".</p> <p>Trigger event: PDF file is corrupted.          → The system notifies "This file cannot be opened" and offers to report the issue.</p>
<b>Qualities</b>	<p>→ QR.34 (variety: minimum 30 resources)</p> <p>→ QR.35 (load time: resource list appears within 2 seconds)</p> <p>→ QR.36 (personalization: prioritize resources matching weaknesses)</p>

**Table 2.13: UC-13-01: View Recommended Jobs**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-13-01
<b>Name</b>	View Recommended Jobs
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil

Table 2.13 – continued from previous page

Section	Content
<b>Priority</b>	Importance for system success: low; Technological risk: medium
<b>Criticality</b>	Low
<b>Source</b>	Career guidance requirements, Job market integration
<b>Responsible</b>	Muhammad Aziq Rauf
<b>Description</b>	The system recommends job listings based on the job seeker's profile, selected domain, experience level, and interview performance.
<b>Trigger event</b>	Job seeker selects "View Recommended Jobs" from the dashboard.
<b>Actors</b>	Job Seeker, Job Recommendation Engine, External Job API
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Job seeker profile is complete.</li> <li>2. Job API integration is configured.</li> <li>3. User has completed at least one interview session.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. List of relevant job openings is displayed.</li> <li>2. Job seeker can view details, save jobs, or apply externally.</li> <li>3. Job recommendations are logged.</li> </ol>
<b>Result</b>	Curated list of job opportunities matching user profile.
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system retrieves the user's profile data.</li> <li>2. The system analyzes the user's interview performance.</li> <li>3. The system queries external job APIs.</li> <li>4. The system receives job listings from APIs.</li> <li>5. The system filters jobs based on relevance and skill level.</li> <li>6. The system ranks jobs by match score.</li> <li>7. The system displays top 10-20 job recommendations.</li> <li>8. The job seeker clicks on a job to view full details.</li> <li>9. The system displays the complete job posting.</li> <li>10. The job seeker can save, apply, or share the job.</li> <li>11. The system logs which jobs were viewed/saved.</li> </ol>

**Table 2.13 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Alternative scenarios</b>	<p>4a. API request fails or returns no results.</p> <p>4a1. The system displays "No job listings available at the moment".</p> <p>4a2. The system suggests expanding search criteria.</p> <p>3a. User has not specified location preferences.</p> <p>3a1. The system defaults to "Remote" jobs.</p> <p>3a2. The system prompts user to update profile with location.</p> <p>5a. User's interview scores are very low.</p> <p>5a1. The system filters for entry-level or training positions.</p> <p>5a2. The system suggests improving interview skills.</p>
<b>Exception scenarios</b>	<p>Trigger event: Job API authentication fails.</p> <p>→ The system displays "Job recommendations temporarily unavailable" and uses cached results.</p> <p>Trigger event: Network connection is lost.</p> <p>→ The system displays "Unable to fetch jobs - please check your internet connection".</p>
<b>Qualities</b>	<p>→ QR.37 (freshness: job listings updated daily)</p> <p>→ QR.38 (relevance: minimum 70% match score)</p> <p>→ QR.39 (variety: show jobs from multiple sources)</p>

**Table 2.14: UC-14-01: Record Response**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-14-01
<b>Name</b>	Record Response
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: high; Technological risk: high
<b>Criticality</b>	High
<b>Source</b>	Multi-modal analysis requirements, Media capture technology
<b>Responsible</b>	Abdullah Jamil
<b>Description</b>	The system captures video (via webcam) and audio (via microphone) of the job seeker's interview response. The recorded media is then processed by the three AI analysis modules.
<b>Trigger event</b>	Job seeker clicks "Start Recording" after being presented with an interview question.

**Table 2.14 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Actors</b>	Job Seeker, Media Capture Module, Browser Media API
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Camera and microphone permissions are granted.</li> <li>2. Browser supports MediaRecorder API.</li> <li>3. Interview question is displayed.</li> <li>4. Storage space is available.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. Video and audio files are captured and stored.</li> <li>2. Files are ready for AI model processing.</li> <li>3. Recording metadata is saved.</li> </ol>
<b>Result</b>	Video and audio files of interview response.
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system requests browser permissions.</li> <li>2. The user grants permissions.</li> <li>3. The system activates the camera and displays a live preview.</li> <li>4. The system displays a countdown.</li> <li>5. The system begins recording video and audio.</li> <li>6. The system displays a recording indicator.</li> <li>7. The job seeker delivers their answer.</li> <li>8. The job seeker clicks "Stop Recording".</li> <li>9. The system stops recording and processes the media.</li> <li>10. The system saves the video file.</li> <li>11. The system saves the audio file separately.</li> <li>12. The system generates a unique identifier.</li> <li>13. The system stores file paths and metadata.</li> <li>14. The system displays "Recording saved successfully".</li> </ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"> <li>2a. User denies permissions. <ol style="list-style-type: none"> <li>2a1. The system displays "Camera and microphone access required".</li> <li>2a2. The system provides instructions on how to enable permissions.</li> </ol> </li> <li>5a. Camera feed fails but audio works. <ol style="list-style-type: none"> <li>5a1. The system notifies "Camera unavailable - recording audio only".</li> <li>5a2. The system proceeds with audio recording only.</li> </ol> </li> </ol>

**Table 2.14 – continued from previous page**

<b>Section</b>	<b>Content</b>
<b>Exception scenarios</b>	<p>Trigger event: Browser does not support MediaRecorder API.  → The system displays "Your browser is not supported".</p> <p>Trigger event: Recording is interrupted.  → The system attempts to recover partial recording.</p> <p>Trigger event: Storage write fails.  → The system displays "Unable to save recording".</p>
<b>Qualities</b>	<p>→ QR.40 (video quality: minimum 480p)</p> <p>→ QR.41 (audio quality: minimum 16kHz)</p> <p>→ QR.42 (latency: recording starts within 1 second)</p> <p>→ QR.43 (reliability: 95% successful capture rate)</p>

**Table 2.15: UC-15-01: Monitor System Performance**

<b>Section</b>	<b>Content</b>
<b>Designation</b>	UC-15-01
<b>Name</b>	Monitor System Performance
<b>Authors</b>	Muhammad Aziq Rauf, Shamil Sajjad, Abdullah Jamil
<b>Priority</b>	Importance for system success: medium; Technological risk: low
<b>Criticality</b>	Medium
<b>Source</b>	System administration requirements, Quality assurance needs
<b>Responsible</b>	Shamil Sajjad
<b>Description</b>	The system administrator monitors system health, AI model performance, user activity metrics, and error logs to ensure the platform operates smoothly and identify areas for improvement.
<b>Trigger event</b>	System administrator selects "Monitor System Performance" from the admin dashboard.
<b>Actors</b>	System Administrator, Performance Monitoring Module
<b>Pre-condition</b>	<ol style="list-style-type: none"> <li>1. Administrator is logged in.</li> <li>2. System metrics are being collected.</li> <li>3. Dashboard is accessible.</li> </ol>
<b>Post-condition</b>	<ol style="list-style-type: none"> <li>1. System health status is reviewed.</li> <li>2. Issues are identified and flagged.</li> <li>3. Performance trends are analyzed.</li> </ol>
<b>Result</b>	System health report and performance metrics.

Table 2.15 – continued from previous page

Section	Content
<b>Main scenario</b>	<ol style="list-style-type: none"> <li>1. The system displays the performance monitoring dashboard.</li> <li>2. The dashboard shows key metrics.</li> <li>3. The administrator reviews the metrics.</li> <li>4. The system displays trend graphs.</li> <li>5. The administrator can drill down into specific metrics.</li> <li>6. The system highlights anomalies or warnings.</li> <li>7. The administrator investigates flagged issues.</li> <li>8. The system allows exporting reports.</li> <li>9. The administrator can set alert thresholds.</li> <li>10. The system automatically sends alerts when thresholds are breached.</li> </ol>
<b>Alternative scenarios</b>	<ol style="list-style-type: none"> <li>6a. All metrics are within normal ranges. <ol style="list-style-type: none"> <li>6a1. The system displays "All systems operating normally".</li> <li>6a2. The administrator reviews trends for optimization.</li> </ol> </li> <li>5a. Administrator wants to analyze a specific time period. <ol style="list-style-type: none"> <li>5a1. The system provides date range filters.</li> <li>5a2. The system updates metrics for the selected period.</li> </ol> </li> </ol>
<b>Exception scenarios</b>	<p>Trigger event: Metrics database is unavailable.  → The system displays "Unable to load performance data".</p> <p>Trigger event: No data collected.  → The system displays "Insufficient data for analysis".</p>
<b>Qualities</b>	<p>→ QR.44 (real-time updates: dashboard refreshes every 30 seconds)</p> <p>→ QR.45 (data retention: store metrics for minimum 90 days)</p> <p>→ QR.46 (alerting: critical issues flagged within 5 minutes)</p>

### 2.1.2 Domain Model

The domain model provides a conceptual representation of the core entities within the Q&Ace system and the relationships between them. It serves as a blueprint for the system's structure, defining the key objects, their attributes, and how they interact.

The diagram illustrates how a **User** (which can be a **Student** or supervised by a **Supervisor**) initiates an **InterviewSession**. Each session presents multiple **Questions** and records a **Response** for each. Each response is then analyzed, generating an **AnalysisResult** that is composed of **FacialAnalysis**, **SpeechAnalysis**, and **NLPAnalysis**. These results are aggregated into a **FeedbackReport**, which the user receives.

2. Project Requirements

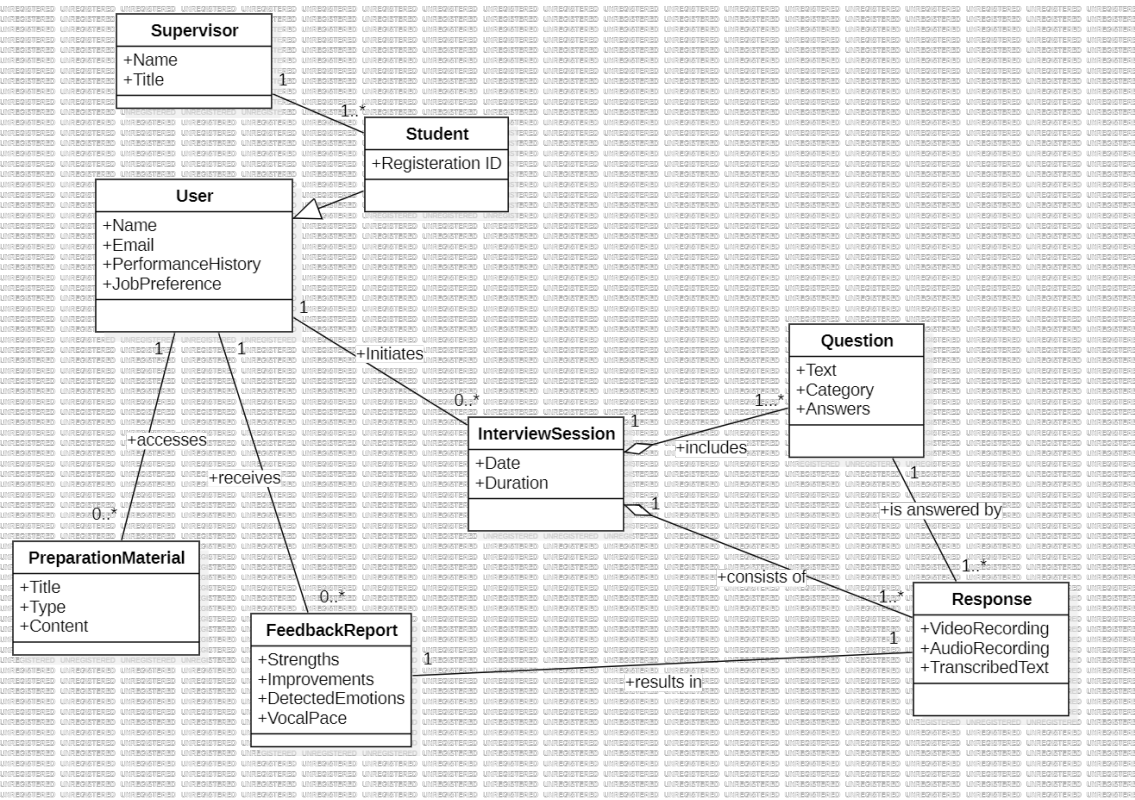


Figure 2.2: Domain Model of the Interview Coaching Platform.



# Chapter 3

## System Overview

The Q & Ace system is an AI-powered web platform designed to function as a personalized interview coach. It provides a simulated and interactive environment for job seekers to practice and refine their interview skills. The system's core functionality is its ability to analyze a user's performance across multiple dimensions by recording their video and audio responses. It leverages distinct AI modules to evaluate non-verbal cues through facial expression analysis, vocal delivery through speech and tone analysis, and the relevance and structure of answers through Natural Language Processing (NLP).

Following each practice session, the platform generates a comprehensive feedback report that includes quantitative scores and qualitative, actionable insights. This allows users to identify specific weaknesses, track their progress over time, and build confidence. The overarching goal is to offer an accessible, on-demand, and intelligent tool that makes high-quality interview coaching available to everyone.

### 3.1 Architectural Design

The architecture of the Q&Ace system is designed using a **Client-Server model with a Microservices-based backend**. This architectural style ensures a clear separation of concerns between the user interface and the complex data processing, promoting scalability, and allowing for the independent development and maintenance of each component. The system is structured into four primary layers.

#### 1. User (Client-Side Layer)

This is the presentation layer, which runs entirely in the user's web browser. It is the sole point of interaction for the job seeker.

- **Technology:** The front end is developed using **ReactJS**.

- **Functionality:** This layer is responsible for rendering the user interface, capturing video and audio data via the user's webcam and microphone, and transmitting the raw interview responses to the backend server for analysis.

## 2. Backend Server (Gateway Layer)

This layer serves as the central API gateway and session manager, acting as a bridge between the client and the analysis microservices.

- **Technology:** It is implemented using the **Flask** web framework.
- **Functionality:** Its primary roles are to handle user session management, receive raw data securely from the client, and forward this data to the appropriate downstream microservices for processing and analysis.

## 3. Microservice Layers (Processing Layer)

The core analytical functionality of the system is decomposed into a set of independent, modular microservices. Each service is responsible for a single, specific task, allowing for greater flexibility and fault tolerance.

- **Module 1 - Facial Emotion Detection:** Utilizes **OpenCV** and a **CNN** model to process video data and detect non-verbal cues.
- **Module 2 - Speech Emotion Tone Analysis:** Employs **LibROSA**, **RNNs**, and **LSTMs** to analyze audio data for vocal tone and emotional indicators.
- **Module 3 - NLP-Based Analysis:** Uses **NLTK**, **spaCy**, and **Transformer** models to evaluate the transcribed text of the user's answer.
- **Module 4 - Adaptive Questioning:** Leverages a **Dynamic DB** and **BERT** to implement stress logic and generate context-aware follow-up questions.
- **Module 5 - Feedback & Progress Tracking:** Aggregates the results from all other analysis modules, generates comprehensive reports, and updates the analytics database.
- **Module 6 - Preparation Module:** Manages and serves content from the **Question Banks** and other data storage.

## 4. Data Storage Layer

This is the persistence layer of the system, responsible for storing all relevant data.

- **Functionality:** It includes a **Centralized Database** for user information, session data, and feedback metrics, as well as storage for the pre-trained **AI/ML Models** used by the microservices.

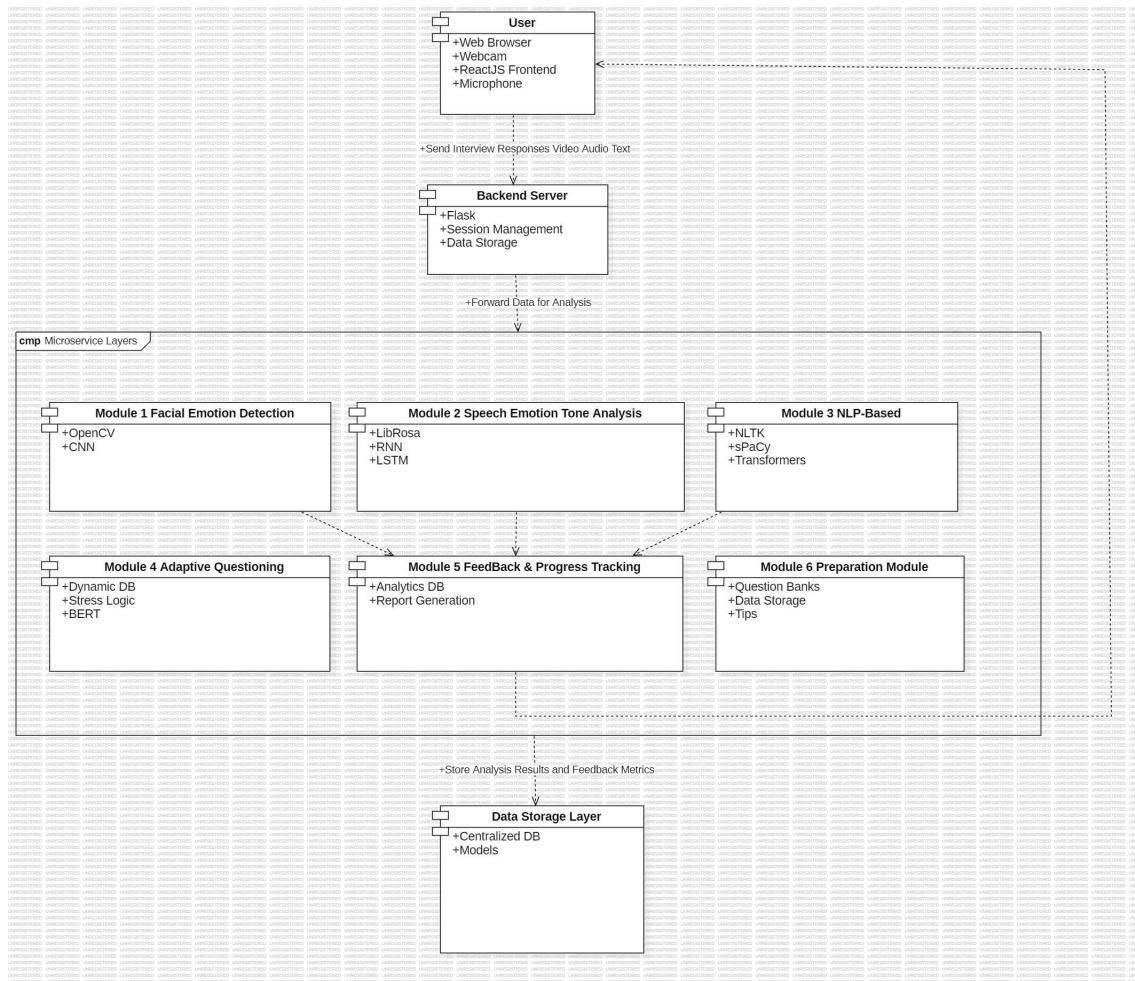


Figure 3.1: Architectural Diagram of the Q&amp;Ace System.

## 3.2 Design Models

As this project follows a Procedural Approach, the following design models are used to clearly represent the system's workflows, data flows, and component interactions.

### Design Models for Procedural Approach

The applicable models for the project using procedural approach may include:

### Activity Diagram

This document presents the activity diagrams for the Q&Ace system, illustrating the workflows for both the Job Seeker and the System Administrator.

## Create Profile Workflow

This diagram shows the sequence of actions a new user takes to create their profile. The process begins with the user opening the application, entering their personal and educational information, and submitting the profile. The system then validates and stores this information, confirming the profile's creation.

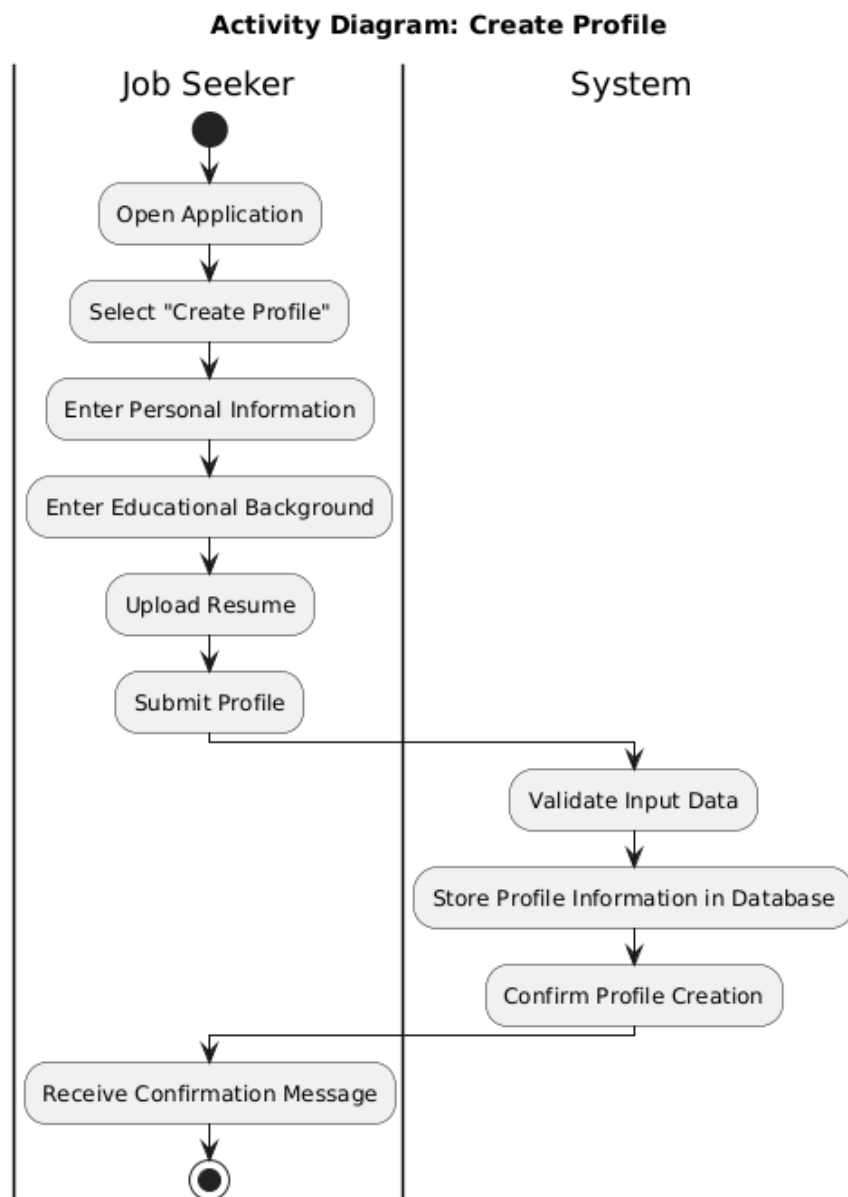


Figure 3.2: Activity Diagram: Create Profile

## Select Domain Workflow

This diagram illustrates how a user selects their professional domain. After logging in, the user navigates to the domain selection screen, chooses a preferred domain, and confirms it. The system validates and saves this preference, which is then used to load a relevant question bank for practice sessions.

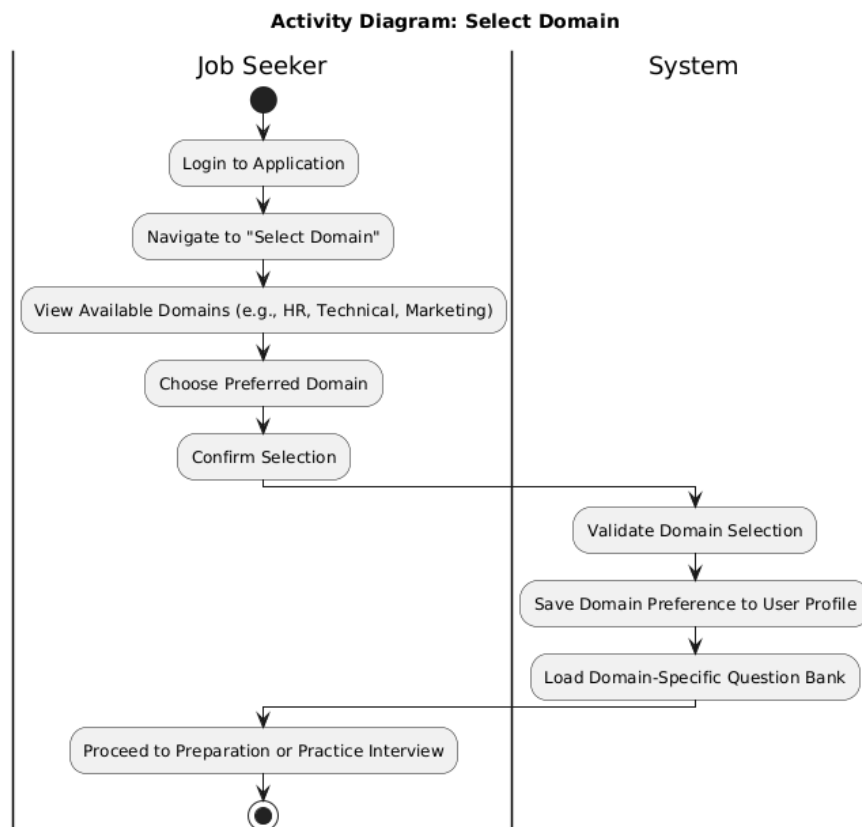


Figure 3.3: Activity Diagram: Select Domain

## Practice Interview Workflow

This is the core workflow for the Job Seeker. The diagram shows the user logging in, choosing an interview type, and answering questions. For each answer, the system captures video, audio, and text, analyzes the data using AI modules, and stores the combined feedback before the user decides to continue or end the session.

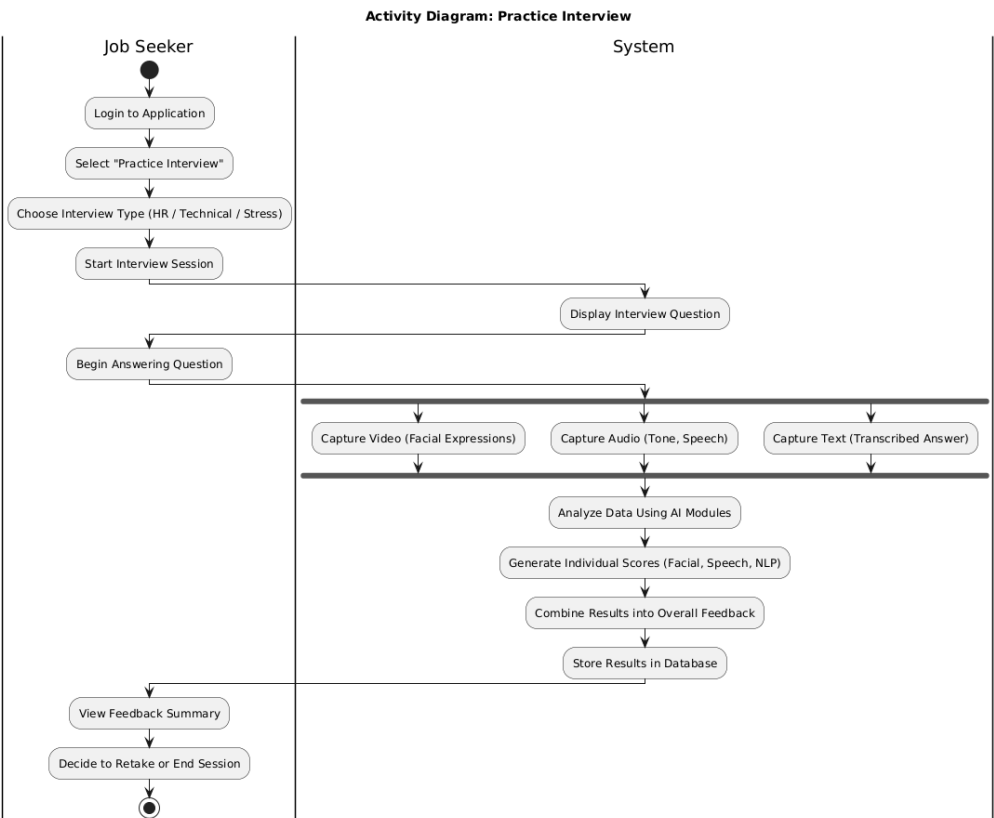


Figure 3.4: Activity Diagram: Practice Interview

### Preparation Module and Job Recommendations

This diagram details the user’s interaction with the preparation module. The user views available resources, selects a topic, and completes exercises. Based on their performance, the system may generate a list of recommended job opportunities.

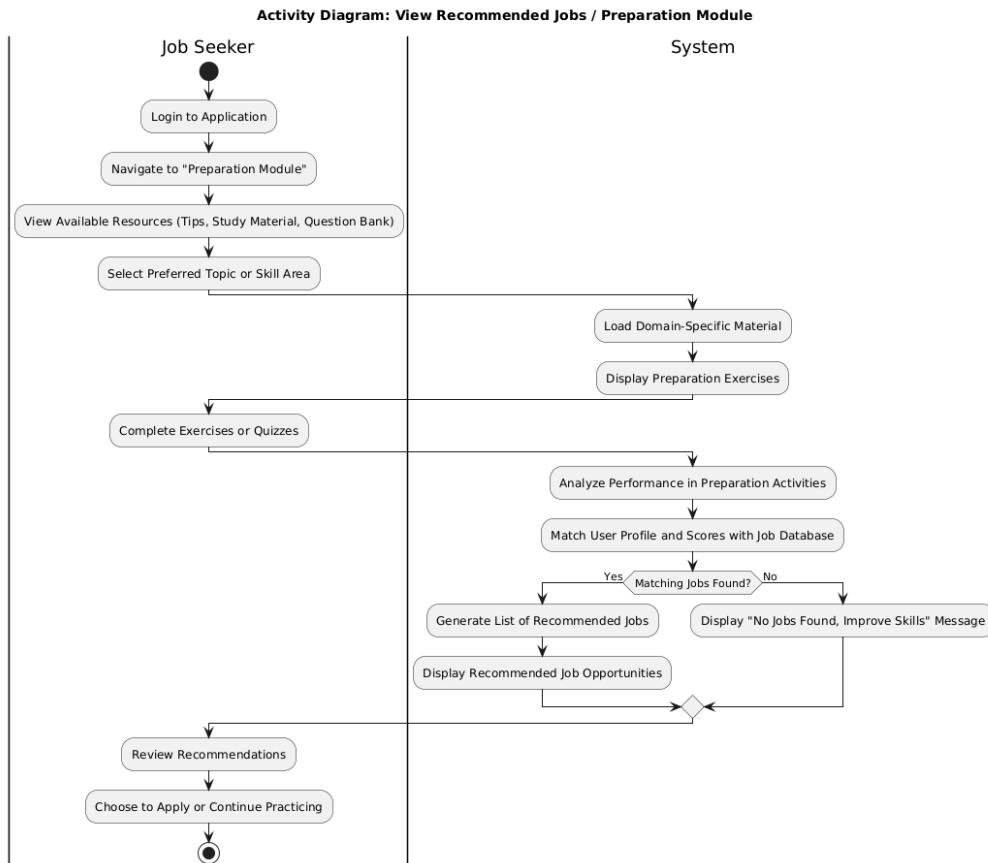


Figure 3.5: Activity Diagram: View Recommended Jobs / Preparation Module

## View Report and Progress Tracking

This workflow shows how a user can review their performance. After navigating to "My Reports," the system checks for existing reports. If found, it retrieves the user's history, generates performance graphs, and displays the detailed report for review.

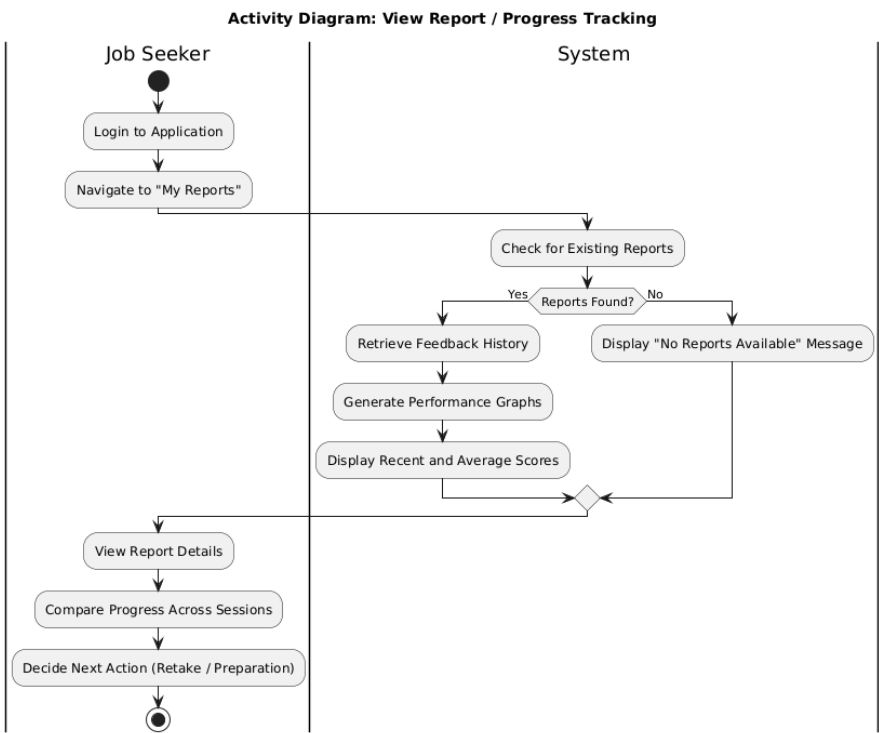


Figure 3.6: Activity Diagram: View Report / Progress Tracking

### Generate Feedback Report Workflow

This diagram details the system’s process for generating a feedback report after a user submits a recorded response. The system retrieves and preprocesses the data, runs the three parallel AI analyses (Facial, Speech, NLP), aggregates the results, and stores the final detailed report in the user’s profile.



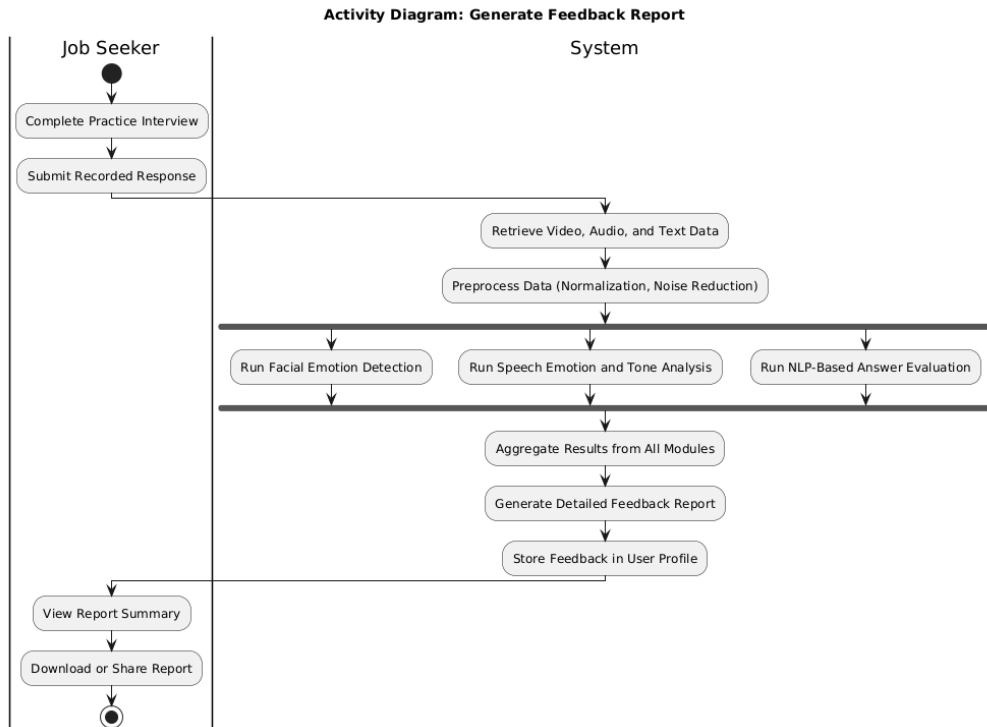


Figure 3.7: Activity Diagram: Generate Feedback Report

## Admin: Manage Interview Question Bank

This diagram illustrates the administrator's workflow for managing the question bank. The admin can add, edit, or delete questions. The system validates these inputs and updates the database accordingly, showing a success message to the admin.

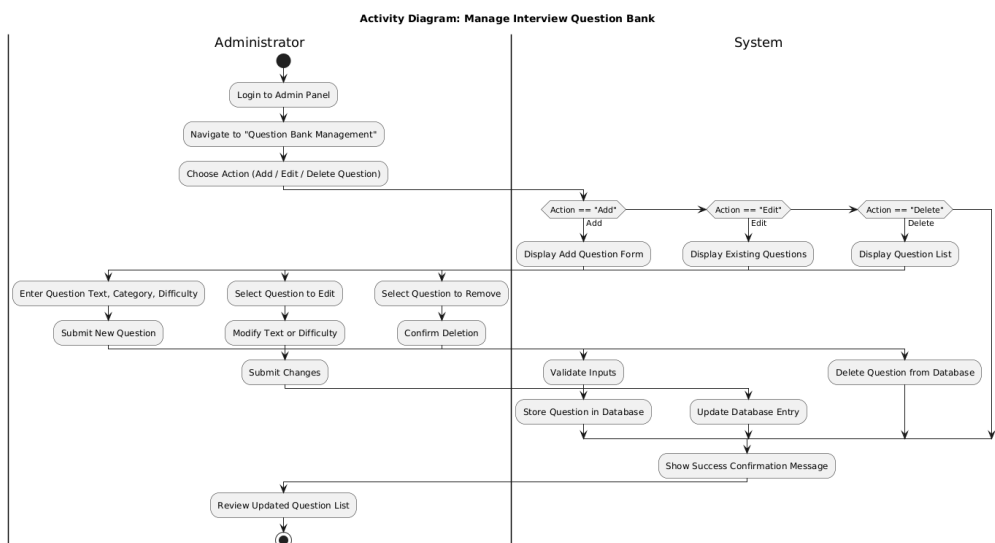


Figure 3.8: Activity Diagram: Manage Interview Question Bank

Admin: Monitor System Performance

This workflow shows how an administrator monitors the system’s health. The admin logs in and views a dashboard with server metrics and application logs. If an issue is detected, the admin can decide on an action, such as restarting a service or documenting the issue for developers.

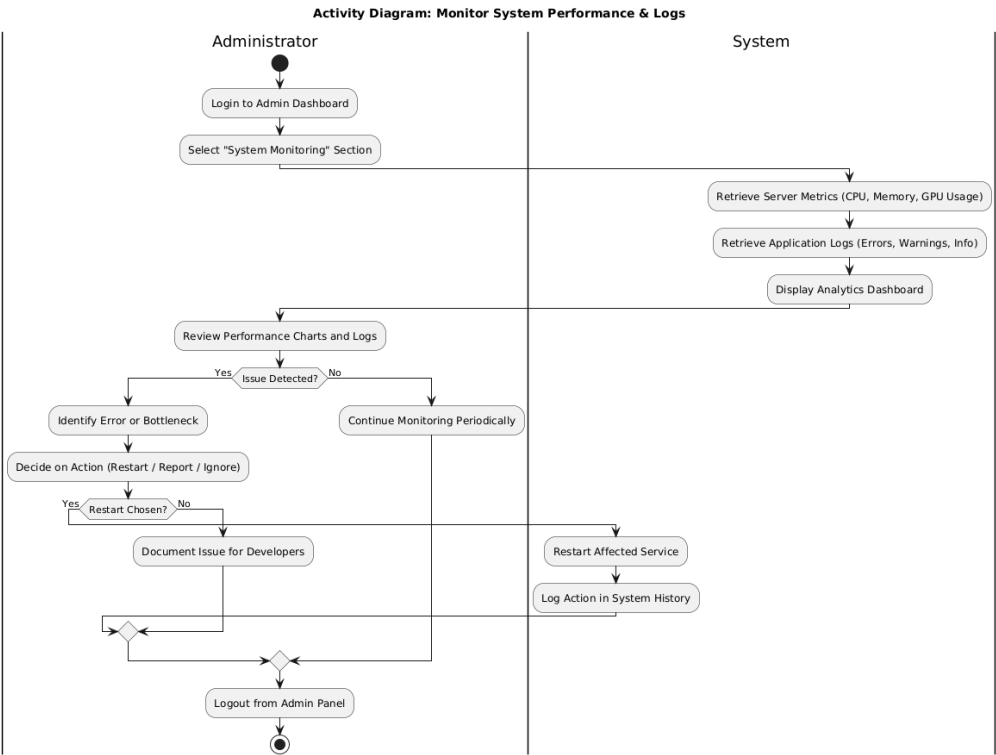


Figure 3.9: Activity Diagram: Monitor System Performance & Logs

Data Flow Diagrams (DFDs)

Data Flow Diagrams (DFDs) are used to graphically represent the flow of data through a system. The following diagrams illustrate the Q&Ace system at three levels of abstraction: the Context (Level 0) DFD provides a high-level overview, the Level 1 DFD breaks the system into its main functions, and the Level 2 DFDs provide a detailed, "exploded" view of the most critical processes.

Level 0 DFD: System Context Diagram

The Context Diagram shows the entire Q&Ace Interview Coaching System as a single process and illustrates its interactions with external entities. There are two primary entities:

- **Job Seeker:** The main user who provides interview responses and receives feedback, questions, and preparation materials.
- **System Administrator:** Responsible for maintaining the system's content by providing new or updated interview questions and monitoring system performance logs.

This diagram clearly defines the system's boundaries and its primary inputs and outputs at the highest level.

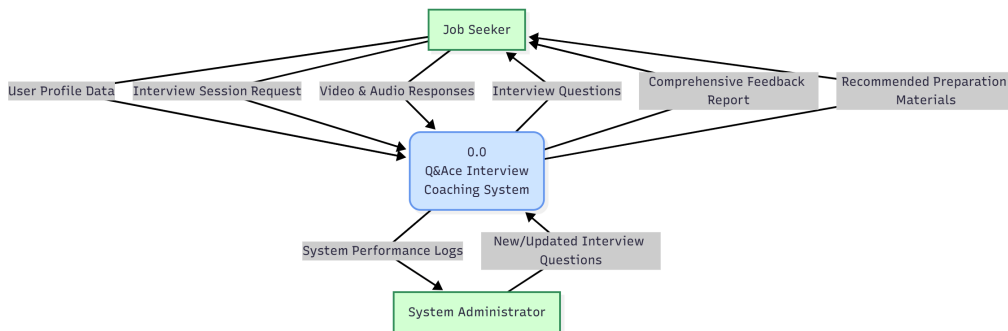


Figure 3.10: DFD Level 0 (Context Diagram) for the Q&Ace System.

### Level 1 DFD: Main System Processes

This diagram decomposes the single process from Level 0 into the five major functional components of the system. It shows how data flows between these processes and the primary data stores they interact with. The main processes are:

1. **Manage User Account (1.0):** Handles user profile creation and management, interacting with the User Profiles data store. This process manages secure registration, login authentication, and allows users to view their session history and track long-term progress. It serves as the central hub for all user-specific information.
2. **Conduct Interview Session (2.0):** Manages the interactive interview, pulling questions from the Question Bank and saving user responses to Raw User Responses. It orchestrates the entire mock interview by presenting questions and capturing the user's complete video and audio answers. This raw data is then securely stored, awaiting detailed analysis.
3. **Analyze User Performance (3.0):** Processes the raw responses to generate performance metrics, which are stored in Analysis Results. This core analytical engine uses AI models to evaluate multiple facets of the user's performance, including facial expressions, vocal tone, and the content of their answers. The resulting scores and metrics provide a holistic view of the user's skills.

4. **Generate Feedback & Recommendations (4.0):** Uses analysis results and Preparation Materials to create reports for the user, which are saved in Feedback Reports. This process transforms the raw scores into an easy-to-understand report that highlights strengths and weaknesses. It also suggests targeted exercises from the preparation library to help the user improve.

5. **Manage System Content (5.0):** Allows the System Administrator to update the Question Bank. This backend process is crucial for keeping the platform's content relevant and challenging. The administrator can add, modify, or remove questions to ensure they align with current industry standards and provide an effective coaching experience.

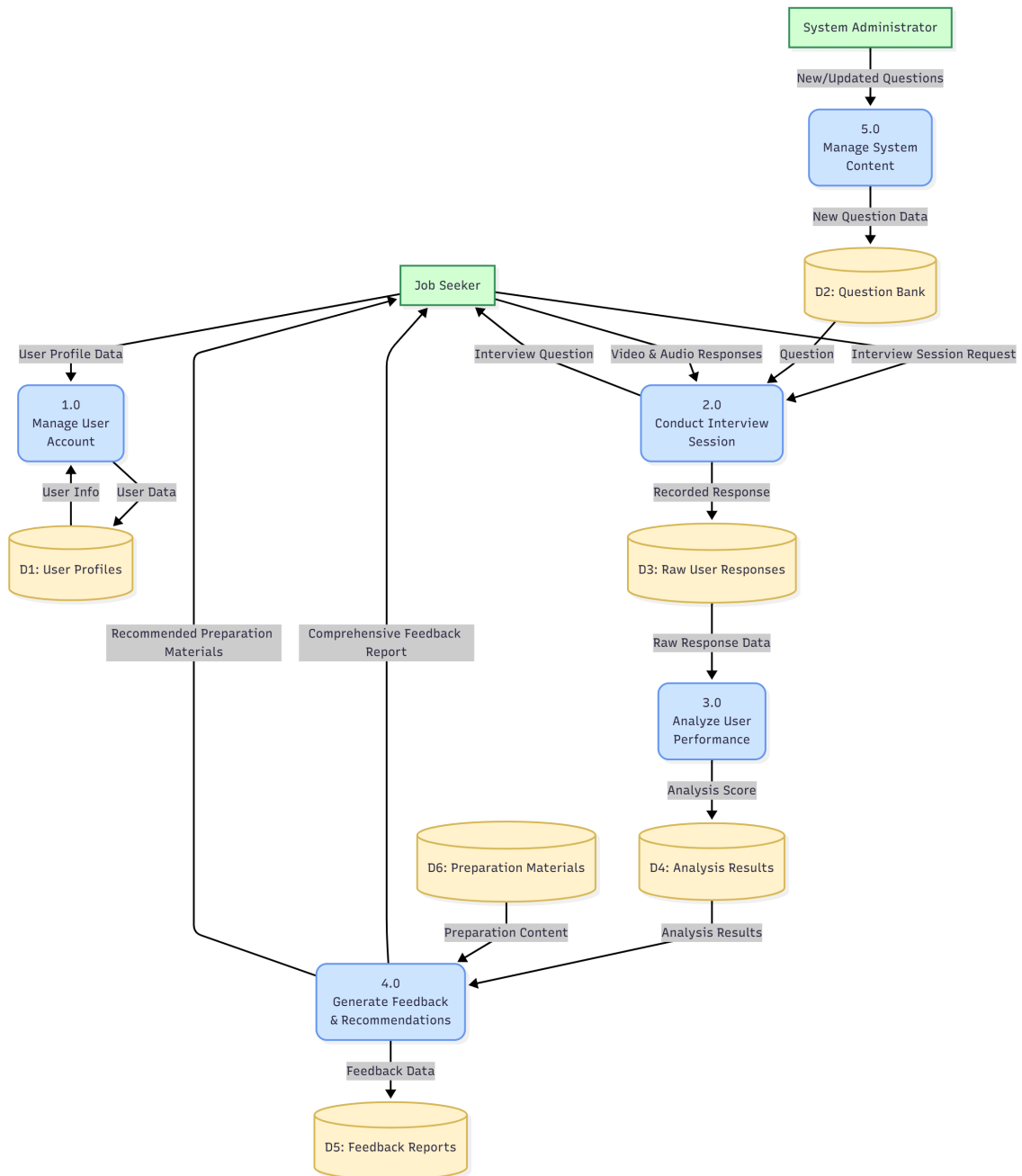


Figure 3.11: DFD Level 1 showing the main processes of the system.

### Level 2 DFDs: Detailed Process Breakdowns

The following diagrams provide a more detailed view of the system's core processes.

**Level 2 DFD for: 2.1 Conduct Interview Session** This diagram provides a detailed breakdown of the Conduct Interview Session process. The workflow begins when

the **Job Seeker** initiates a session by making an Interview Selection. This triggers the Select Interview Type sub-process, which then fetches a question from the **Question Bank**. The user's Spoken Answer is then captured by the Record User Response sub-process. A key part of this diagram is the Check Response Clarity logic; if a response is unclear, the system can re-prompt. Once clear, the response is saved to the Raw User Responses data store, and the Determine Next Question logic selects the subsequent question.

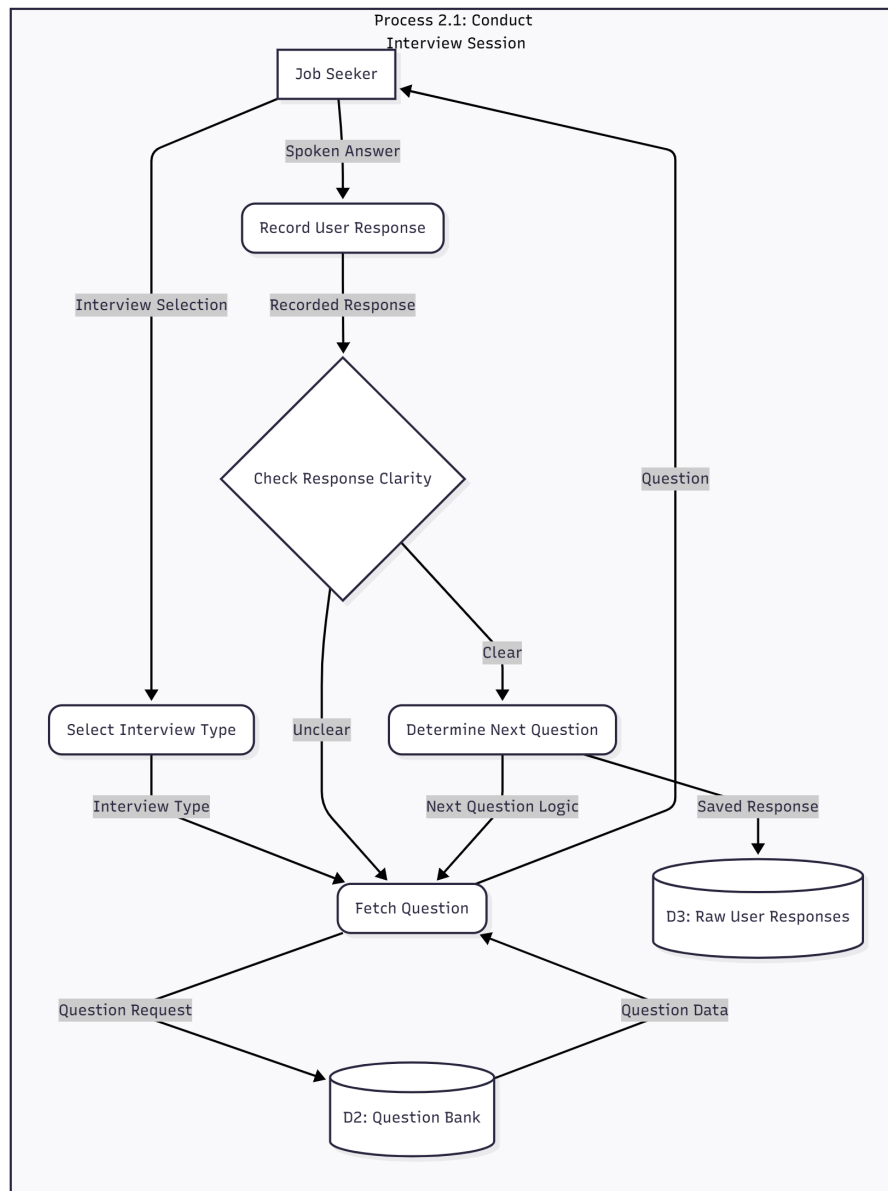


Figure 3.12: DFD Level 2 for the Conduct Interview Session Process.

**Level 2 DFD for: 2.2 Analyze User Performance** This diagram illustrates the core analysis engine of the Q&Ace system. It "explodes" the Analyze User Performance

process, starting with raw Video & Audio Response data from the data store. This data is processed by three parallel sub-processes: Analyze Facial Data, Analyze Speech Data, and Transcribe Audio to Text. The output from the transcription is then fed into Analyze Text Content. The metrics from all analysis streams are aggregated by the Calculate Overall Scores sub-process, and the final Aggregated Scores are written to the Analysis Results data store.

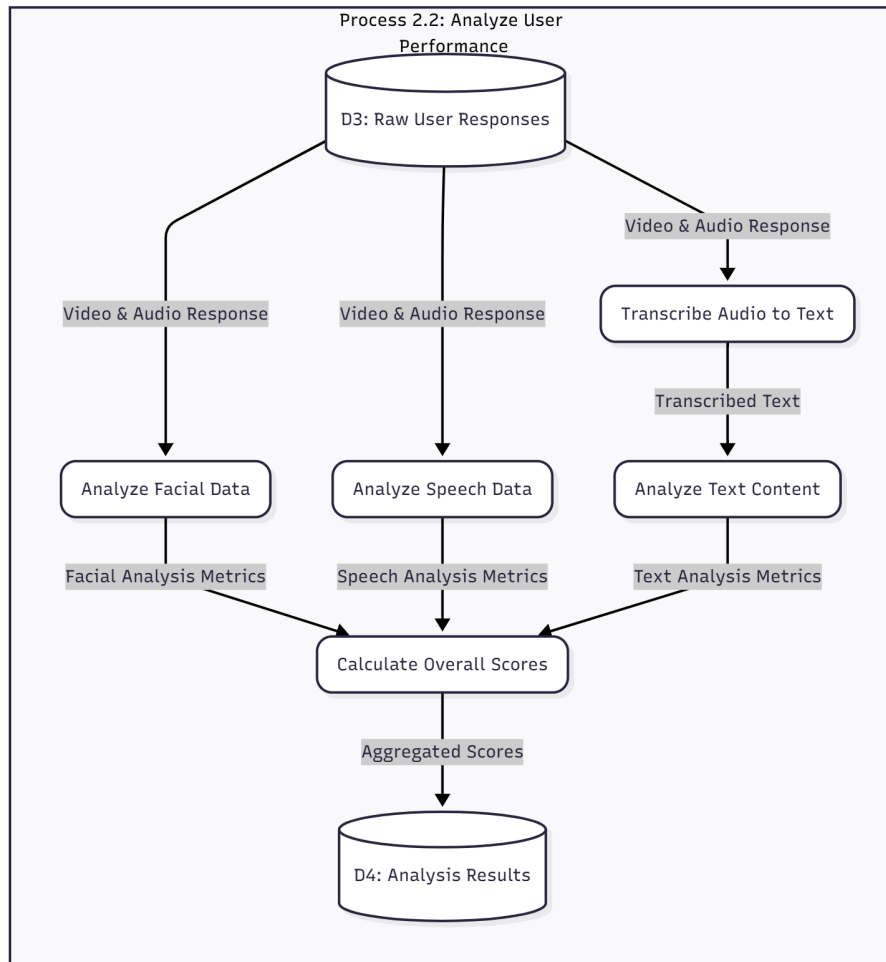


Figure 3.13: DFD Level 2 for the Analyze User Performance Process.

**Level 2 DFD for: 2.3 Generate Feedback and Recommendations** This diagram focuses on the post-analysis workflow. The process begins by reading Analysis Results to Compile Analysis Scores. This compiled data flows into two parallel sub-processes:

- **Generate Feedback Report:** Creates a user-friendly report for the **Job Seeker** and archives it in the Feedback Reports data store.
- **Identify Shortcomings:** Pinpoints weaknesses and triggers Recommend Preparation

Materials, which provides the user with Targeted Exercises from the corresponding data store.

Finally, the Track User Progress process logs the user's interaction with these materials, updating their profile in the User Profiles data store to create a continuous improvement loop.

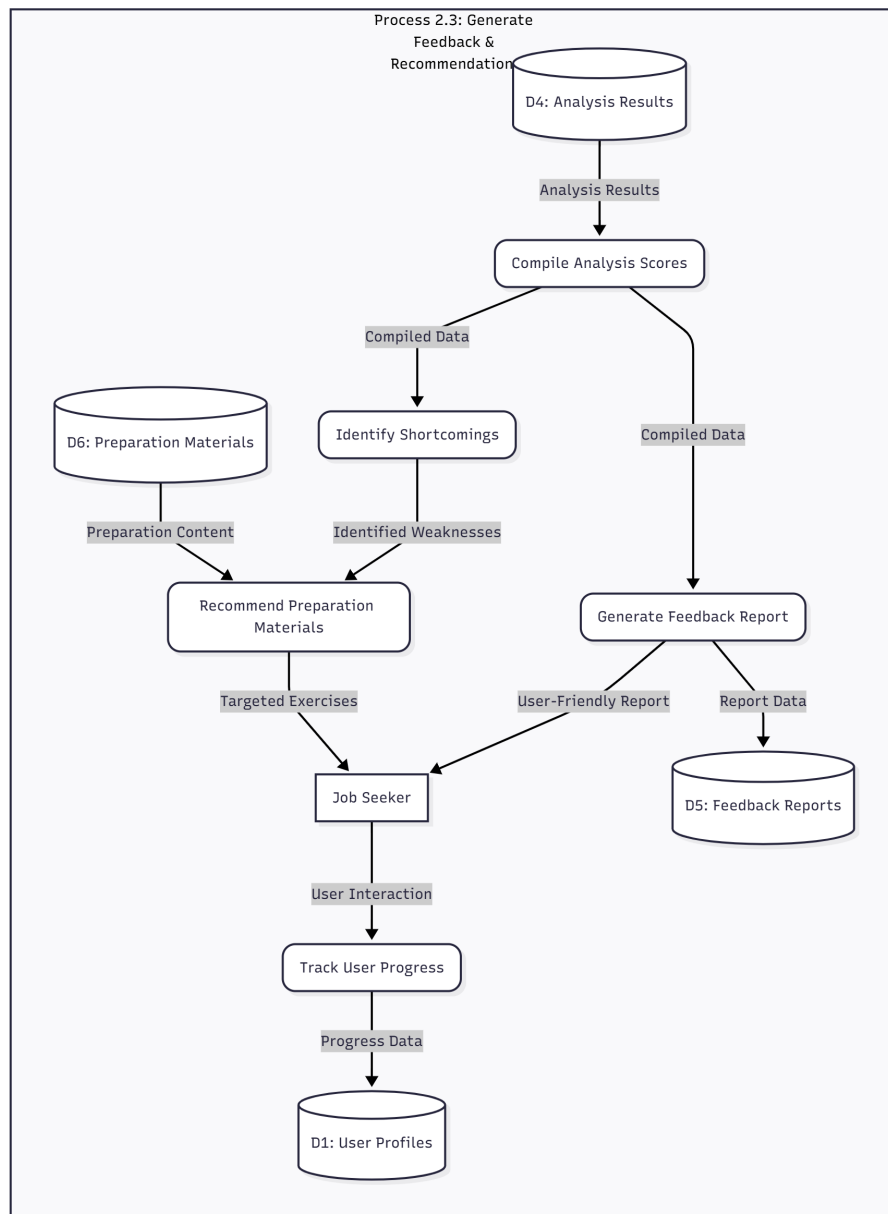


Figure 3.14: DFD Level 2 for the Generate Feedback and Recommendations Process.



## 3.3 Data Design

The data design for the Q&Ace system is structured to handle user information, raw interview data, and the structured results from the AI analysis. The information domain is transformed into distinct data structures that are processed and organized to support the entire interview coaching workflow.

### Data Entities

The primary data entities that represent the core components of the system include:

- **User:** Contains user-specific information like UserID, Name, Email, and historical performance metrics.
- **InterviewSession:** Stores data for each practice session, including SessionID, UserID, Date, and Duration.
- **Response:** Holds the data for each individual answer, including ResponseID, a path to the video and audio files, and the transcribed text.
- **AnalysisResult:** Stores the output from the AI models for each response, with fields like OverallScore, DetectedEmotions, FillerWordCount, and ClarityScore.
- **FeedbackReport:** Contains the compiled results for a session, including identified strengths and areas for improvement.

### Data Storage (MongoDB-centric)

**Overview** The system uses a hybrid storage architecture centered on **MongoDB** (NoSQL) for structured and semi-structured data, plus a separate file storage service for large media (video/audio). MongoDB is chosen for its flexible schema, horizontal scalability, and rich secondary-index and aggregation capabilities which are well-suited to storing user profiles, session metadata, analysis scores, and feedback.

#### High-level architecture

- **Primary datastore:** MongoDB cluster (replica set, optionally sharded for scale).
- **File storage:** Prefer object storage (S3 / S3-compatible cloud bucket) for raw session media; optionally GridFS for small deployments or when you require the media to live inside MongoDB.

- **Cache / ephemeral store:** Redis (optional) for fast session state, rate-limiting, and short-lived results.
- **Model training data:** Stored separately (object storage or data lake), with pointers/metadata in MongoDB.

#### Schema design guidance (embed vs reference)

- **Embed** when data is tightly coupled and small (e.g., small profile subdocument inside users).
- **Reference** when data grows independently or is reused (e.g., sessions reference users and question\_bank).
- Use **denormalization** for read performance where appropriate (store commonly queried summary fields like overallScore inside sessions).

#### Indexing and query patterns

- Index `users.email` and `users.username` (unique).
- Compound index on `sessions.userId`, `sessions.startedAt` for querying a user's session history.
- TTL index for any temporary or ephemeral collections (e.g., a collection that stores short-lived logs): `db.temp.createIndex({createdAt:1}, {expireAfterSeconds: 3600})`.
- Index on `analysis_results.sessionId` for fast join-like lookups.
- If you support searching questions by tag/difficulty, index `question_bank.tags` (multikey).

#### File storage: GridFS vs External Object Storage

**Recommended: External object storage (S3/S3-compatible)** for production:

- Pros: cheaper, scalable, CDN integration, direct upload from client (pre-signed URLs), independent lifecycle policies.
- Store only metadata/URLs in MongoDB (e.g., `media.audioUrl`).

**GridFS** can be used when:

- You need to keep all data inside MongoDB for transactional simplicity or operational constraints.
- But GridFS generally adds operational complexity and doesn't scale as cheaply as object storage.

### **Data lifecycle, retention and GDPR considerations**

- Define retention for raw media (e.g., keep raw video for 90 days) and implement lifecycle rules in object storage to delete or archive media automatically.
- Keep analysis results and anonymized aggregates longer if needed for model improvement, but purge PII on request.
- Implement deletion cascades: when a user requests account deletion, remove personal documents and revoke media via object storage API, then remove MongoDB metadata.

### **Transactions and consistency**

- Use MongoDB multi-document transactions for operations that must be atomic across collections (e.g., creating a session and writing its initial metadata + an audit document).
- For most read-heavy operations maintain eventual consistency and denormalized summary fields where necessary.

### **Real-time and change capture**

- Use **Change Streams** to react to new analysis results (e.g., trigger notifications, enqueue report generation, update dashboards).
- Change Streams require a replica set (even a single-node replica set in development).

### **Security and access control**

- Enable **authentication** and use least-privilege roles (separate read-only, app-write, and admin roles).
- Use **TLS/SSL** for all connections to MongoDB.

- Enable **Encryption at Rest** (MongoDB Encrypted Storage Engine or rely on cloud provider disk encryption).
- Use field-level encryption for highly sensitive fields (emails, personal identifiers) if required.
- Rotate credentials and use managed secret stores (Vault, cloud secret manager).

#### **Backup, replication and high availability**

- Deploy MongoDB as a **replica set** (primary + secondaries) for HA and failover.
- For horizontal scale, use **sharding** by a well-chosen shard key (e.g., tenantId or hashed userId) when dataset becomes very large.
- Implement regular backups (logical mongodump/mongoexport or cloud provider snapshots) and test restores periodically.
- Maintain a retention policy for backups and store them in separate object storage region.

#### **Performance and scaling**

- Monitor slow queries and use profilers to identify missing indexes.
- Use connection pooling in your application drivers and tune pool size to cluster capacity.
- Offload analytics work to an OLAP store or run aggregation pipelines on secondary nodes to avoid impacting primary write performance.
- For heavy search requirements consider integrating Elasticsearch / OpenSearch and store only authoritative data in MongoDB.

#### **Operational recommendations and tooling**

- Use managed MongoDB (Atlas) or a well-supported operator (e.g., MongoDB Kubernetes Operator) to simplify ops.
- Use monitoring (Prometheus + Grafana, or Atlas built-in metrics) and alerting for disk, CPU, replication lag, and memory.
- Enforce schema validation rules in MongoDB collections to catch malformed documents early.

- Use CI checks for migrations and data model changes; write migration scripts for any field renames or denormalization changes.

### **Model training data management**

- Store large training datasets (FER-2013, RAVDESS, custom QA corpora) in object storage with metadata in MongoDB (version, source, preprocessing steps). [1]
- Track dataset versions and training runs (store experiment metadata in a collection like `training_runs`) so analysis improvements are reproducible.

**Closing notes** This design favors a **single source of truth** for metadata in MongoDB and scalable, cost-effective object storage for large media. It supports transactional needs, real-time processing (via change streams), and production best practices (backup, monitoring, security). Adjust sharding, retention, and encryption choices to your expected scale and compliance requirements.



# Bibliography

- [1] Steven R. Livingstone and Frank A. Russo. The ryerson audio-visual database of emotional speech and song (ravdess). *PLOS ONE*, 13(5):e0196391, 2018.
- [2] OpenCV Team. Open source computer vision library. <https://opencv.org/>, 2024. [Online].