Analysis and Requirements Report

RecruitAssistant - Team T2502

Deniz Öztürk - 22102126
Emir Ensar Sevil - 22201926
Yüksel Barkın Baydar - 22201939
Mehmet Anıl Yeşil - 22102614
Cankutay Dündar - 22103284

Supervisor: Prof. Özgür Ulusoy

1. Introduction

This Analysis and Requirements Report presents the detailed analytical foundation for the design and development of **RecruitAssistant**, an AI-powered, end-to-end career guidance platform that supports users throughout the hiring lifecycle. The report formalizes the requirements of the system, identifies functional and non-functional specifications, develops object-oriented models to represent system behavior, and documents all constraints, risks, engineering considerations, teamwork strategies, and learning plans.

The analysis stage aims to achieve a specification that is **correct, complete, consistent, and verifiable**, following principles described in both the CS491 guideline and software engineering standards such as IEEE 830 and UML 2.5.1. [1] This document therefore functions as a **contract** between the development team (Team T2502) and the customer (the CS491 committee and supervisor), defining what will be built, why it is needed, and how it will behave under normal and exceptional circumstances.

RecruitAssistant's mission is to become a comprehensive and intelligent platform for job-seeking students and early-career professionals. It integrates advanced NLP, transformer-based question generation, automated CV creation, speech-to-text transcription, interview evaluation models, quizzes, and progress analytics into a unified solution. The system aims to assist users from the earliest stages of job application preparation (CV and cover letter creation) through mock interviews and post-interview performance feedback. The MVP will target **4th-year CS students from METU, Bilkent, and Hacettepe**, with Turkish-language interviews and English-language website interfaces.

The remainder of this report presents a detailed, structured analysis of the system, the requirements identified, the constraints and engineering factors considered, and the project plan that will guide the development effort throughout CS491 and CS492.

2. Current System

There is currently **no integrated solution** that provides an end-to-end AI-based recruitment preparation workflow for university-level job seekers. Existing tools typically address isolated components, such as:

- Simple CV builders
- Limited AI-based resume enhancers

- Standalone mock interview platforms
- Generic quiz or assessment tools
- Interview question banks without personalization
- Commercial coaching services with high costs

None of these platforms offer:

- Dynamic **job-specific CV generation** using NLP
- **Turkish-language mock interviews** evaluated automatically
- **Real-time speech-to-text scoring** using Whisper
- **Personalized learning paths** based on user weaknesses
- A **unified progress dashboard** tracking performance over time
- A **cross-platform, AI-driven, university-focused MVP**

Thus, the current landscape lacks a cohesive, structured, data-driven platform that supports the full recruitment preparation pipeline. RecruitAssistant fills this gap by providing a seamless, unified experience with strong personalization and state-of-the-art ML models.

3. Proposed System

3.1 Overview

RecruitAssistant is an AI-powered web application designed to guide users from CV creation through mock interviews and post-interview analysis. It consists of:

- **React frontend** (web, responsive)
- **FastAPI backend** implementing business logic
- **AI/ML engine** integrating Whisper, transformer-based NLP models, sentiment analysis, and evaluation modules
- **PostgreSQL database** for structured storage (users, CVs, quizzes, transcripts, analytics)

- **AWS infrastructure** for scalability and model hosting (S3, EC2/Lambda, RDS as needed)

- **GitHub repository + project management tooling**

The system emphasizes modularity, efficiency, ethical AI usage, and compliance with security and privacy regulations.

3.2 Functional Requirements

3.2.1 User Account & Authentication

- The system shall allow users to create accounts using an email address and a password.

- The system shall securely authenticate users using session-based or token-based authentication.

- The system shall allow password reset via email.

- The system shall ensure that only authenticated users access personalized data.

3.2.2 CV & Cover Letter Generation

- The system shall allow users to upload or enter job descriptions.

- The system shall generate job-specific CVs using NLP models.

- The system shall allow users to modify generated CVs via an editor.

- The system shall allow exporting CVs in PDF format.

- The system shall store multiple CV versions per user.

3.2.3 Mock Interview Module

- The system shall generate interview questions based on job descriptions and domains.

- The system shall support **Turkish-language** interview sessions for the MVP.

- The system shall perform **real-time transcription** using Whisper.

- The system shall evaluate responses using NLP scoring metrics.
- The system shall identify weaknesses and produce targeted feedback.
- The system shall store interview transcripts and evaluation results.
- The system shall calculate performance metrics (fluency, relevance, structure). [9]

3.2.4 Quiz & Learning Module

- The system shall generate quizzes based on user weaknesses.
- The system shall store quiz results and track improvement.
- The system shall provide detailed explanations for correct/incorrect answers.
- The system shall adapt future quizzes based on previous results.

3.2.5 Progress Tracking & Analytics

- The system shall provide dashboards displaying performance trends.
- The system shall visualize strengths and weaknesses.
- The system shall produce weekly summaries and suggestions.

3.2.6 Administrative Functions

- The system shall allow admin/mentor access to anonymized analytics (optional).
- The system shall allow content updates (question sets, CV templates).

3.3 Non-Functional Requirements

3.3.1 Performance

- Interview evaluation shall complete within 5 seconds after each response.
- CV generation shall complete within 2–5 seconds. [2]

3.3.2 Reliability

- The system shall maintain backend availability during operational hours.
- The system shall store interview transcripts even if sessions disconnect.
- Backups of user data shall occur weekly.

3.3.3 Usability

- UI shall be intuitive for non-technical users.
- Users shall complete registration within 2 minutes.
- Mock interview flow shall require no more than three clicks from the dashboard.

3.3.4 Scalability

- The system shall support horizontal scaling through AWS.
- The PostgreSQL database shall support potential sharding or read replicas. [10]

3.3.5 Security

- Passwords shall be hashed using industry standards.
- Sensitive data shall be encrypted at rest.
- All APIs shall require HTTPS. [5] [6]

3.3.6 Supportability

- Code shall follow modular architecture with service separation.
- Documentation shall cover API endpoints, environment setup, and ML pipelines.

3.4 Pseudo Requirements

- The MVP shall target METU, Bilkent, and Hacettepe CS students.
- Mock interview language: **Turkish**
- System interface language: **English/Turkish**

- The product shall support browser-based access only for MVP.
- The system shall use only free/open-source models or AWS resources under free tier when possible.

3.5 System Models

3.5.1 Scenarios

Use Case 1: User Registration

Entry Condition: User is not logged in and has not created an account.

Actor(s): User

Flow of Events:

1. The user opens the platform and selects Sign Up.
2. The user enters required information (e.g., email, password, name).
3. The system validates the input format and checks if the email already exists.
4. The system creates the user account and securely stores credentials (hashed).
5. The system sends verification (optional) and redirects users to login or onboarding.

   Exit Condition: Account is created (and optionally verified); user is ready to log in.

Use Case 2: User Login

Entry Condition: User has an existing account and is not logged in.

Actor(s): User

Flow of Events:

1. User selects Login.
2. The user enters an email and password.
3. System validates credentials.
4. The system creates an authenticated session/token.

5. The system redirects the user to the dashboard.

   Exit Condition: User is authenticated and can access protected features.

Use Case 3: Profile Creation / Update

Entry Condition: User is logged in.

Actor(s): User

Flow of Events:

1. User navigates to Profile.
2. User enters/updates data (education, experience, skills, projects, target roles).
3. The system validates required fields and formats.
4. The system saves profile data to the database.
5. System updates “readiness” summary and suggestions (optional).

   Exit Condition: Profile data is stored and available to CV/interview/quiz modules.

Use Case 4: Job Posting Import (Job Description Parsing)

Entry Condition: User is logged in and wants role-specific preparation.

Actor(s): User

Flow of Events:

1. User selects Add Target Job.
2. The user pastes a job description or provides a job link/text.
3. The system extracts keywords (skills, responsibilities, requirements).
4. The system stores the job posting and maps it to user profile fields.
5. The system generates a role-specific preparation plan (optional).

   Exit Condition: Target job is saved and parsed; role keywords are available for tailoring.

Use Case 5: Job-Specific CV Generation

Entry Condition: User is logged in; user profile exists; at least one target job exists or user provides role/keywords.

Actor(s): RecruitAssistant System

Flow of Events:

1. The system detects a CV generation request initiated by the user.
2. The system retrieves the user's profile information, including skills, education, experience, and projects.
3. The system retrieves and analyzes the selected job description or provided role keywords.
4. The system generates a job-specific CV draft using an ATS-friendly structure and formatting rules.
5. The system presents the generated CV draft to the user for preview and optional manual editing.
6. Upon user confirmation, the system saves the generated CV version to persistent storage.

   Exit Condition: A new job-specific CV version is successfully generated and stored in the system.

Use Case 6: CV Export / Download

Entry Condition: User is logged in; at least one CV version exists.

Actor(s): User

Flow of Events:

1. The user opens the CV list and selects a CV version.
2. The user selects export format (e.g., PDF/DOCX) and template style.
3. System renders the CV into the selected format.
4. The system provides the file for download.

5. System logs the export action for history (optional).

   Exit Condition: User successfully downloads/exported the CV file.

Use Case 7: Quiz Generation (Pre-Interview Quiz)

Entry Condition: User is logged in; target job/role exists; question bank or generation model is available.

Actor(s): RecruitAssistant System

Flow of Events:

1. The system detects a quiz generation request associated with a selected job role or skill focus.
2. The system analyzes job requirements and previously recorded user performance data (if available).
3. The system selects or generates quiz questions tailored to the job role and identifies knowledge gaps.
4. The system initializes a new quiz session and configures quiz parameters (number of questions, time limits).
5. The system displays quiz instructions and loads the first question for the user.

   Exit Condition: A quiz session is created and ready to be answered.

Use Case 8: Quiz Taking and Automatic Grading

Entry Condition: An active quiz session exists for the user.

Actor(s): User

Flow of Events:

1. The system shows questions one-by-one or as a full list.
2. The user submits answers (MCQ, short answer, or mixed).
3. The system validates answer submission and time limits.
4. System grades answers using an answer key/rubric/model.

5. The system shows score breakdown and recommended topics to review.
6. The system stores quiz results and analytics in the database.

Exit Condition: Quiz is completed, graded, results stored, feedback displayed.

Use Case 9: Mock Interview Session (Audio + Transcription)

Entry Condition: User is logged in; target job/role exists; microphone permission granted (if audio).

Actor(s): User

Flow of Events:

1. The user selects the Start Mock Interview and chooses the job/role and difficulty.
2. The system generates interview questions (behavioral/technical) based on role.
3. User answers via audio (or text fallback).
4. The system records audio (if enabled) and transcribes speech to text (STT).
5. The system repeats for all questions until the interview ends.

Exit Condition: Interview session ends; transcripts (and optionally audio) are saved.

Use Case 10: Interview Evaluation and Personalized Feedback Report

Entry Condition: A completed mock interview session exists with transcript (and optionally quiz history).

Actor(s): User

Flow of Events:

1. The system retrieves the selected completed mock interview session and its transcript.
2. The system evaluates interview responses using predefined rubrics and machine learning models, considering clarity, relevance, structure, and technical correctness.
3. The system identifies weak areas and maps them to relevant skills or competency topics.

4. The system generates a personalized feedback report, including per-question scores, overall performance metrics, and improvement suggestions.

5. The system optionally generates personalized recommendations such as follow-up quizzes or targeted study plans.

6. The system stores the feedback report and associated evaluation results in persistent storage.

   Exit Condition: A personalized interview feedback report is generated, stored, and made available to the user.

3.5.2 Use Case Model

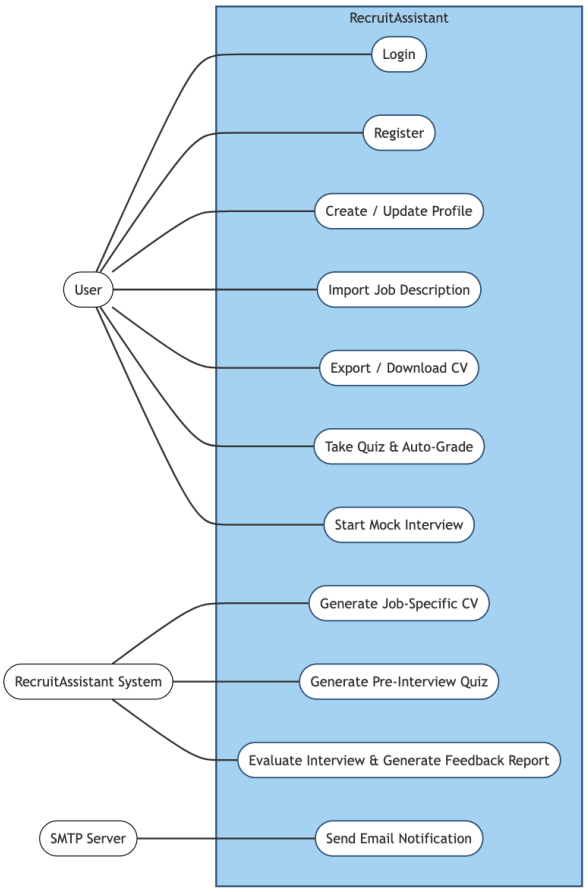

Fig 1: Use Case Model

3.5.3 Object & Class Model

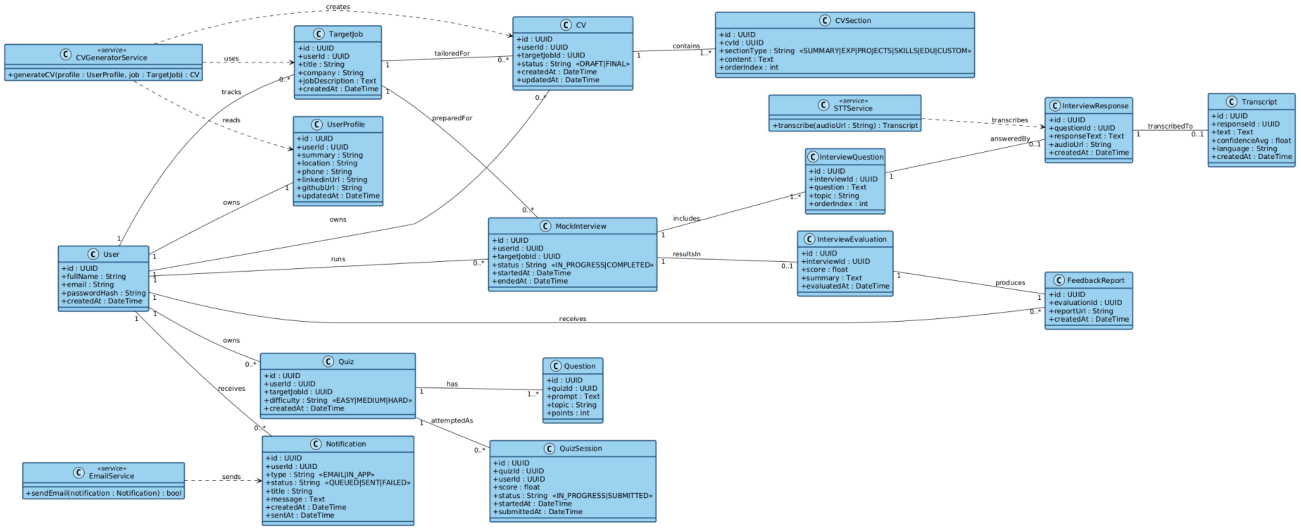

Fig 2: Object & Class Model

Class Descriptions:

User:

This object class represents the main user account in the RecruitAssistant system. It stores essential identity and authentication information such as full name, email address, hashed password, and account creation timestamp. A User can own a profile, track multiple target job applications, generate job-specific CVs, participate in quizzes and mock interviews, receive feedback reports, and receive system notifications.

UserProfile:

This object class represents the professional profile of a user. It stores reusable career-related information such as professional summary, location, phone number, and external links (e.g., LinkedIn and GitHub). The UserProfile is used by multiple system components, including CV generation, interview question generation, and evaluation services. Each UserProfile belongs to exactly one User.

TargetJob:

This object class represents a target job or role selected by the user. It stores job-specific information such as job title, company name, and the raw job description text. TargetJob instances are used as contextual inputs for tailoring CVs, generating quizzes, preparing mock interviews, and producing role-specific feedback.

CV:

This object class represents a job-specific CV generated for a user. It contains metadata such as status (draft or final), creation time, and last update time. Each CV belongs to a User and may be linked to a TargetJob to indicate which role the CV was tailored for. A CV is composed of multiple CVSection objects.

CVSection:

This object class represents a structured section of a CV, such as Summary, Experience, Projects, Skills, Education, or Custom sections. It stores the section content and

an order index that determines its placement in the CV. CVSection enables modular CV editing and ATS-friendly formatting.

CVGeneratorService:

This service class is responsible for generating job-specific CVs. It retrieves user profile data and job requirements, applies keyword alignment and formatting rules, and creates a tailored CV instance. This class encapsulates all CV generation and customization logic in the system.

Quiz:

This object class represents a pre-interview quiz generated for a user. It may be associated with a specific TargetJob and stores metadata such as difficulty level and creation time. A Quiz serves as a container for multiple quiz questions and is used to assess user readiness before interviews.

Question:

This object class represents an individual quiz question. It stores the question prompt, topic tag, and point value. Each Question belongs to exactly one Quiz and is used during quiz sessions to evaluate the user's knowledge in job-relevant areas.

QuizSession:

This object class represents a single attempt by a user to complete a quiz. It stores session-level information such as score, status (in-progress or submitted), and timestamps. QuizSession allows the system to track performance history and identify recurring knowledge gaps.

MockInterview:

This object class represents a mock interview session conducted by a user. It is associated with a TargetJob and stores status information (in-progress or completed) along with start and end timestamps. A MockInterview contains multiple interview questions and may result in an interview evaluation.

InterviewQuestion:

This object class represents an interview question presented during a mock interview. It stores the question text, topic, and ordering index. Each InterviewQuestion belongs to exactly one MockInterview and may have one associated InterviewResponse.

InterviewResponse:

This object class represents the user's response to an interview question. It stores the textual response and may optionally reference an audio recording URL for spoken answers. InterviewResponse serves as the input for transcription and interview evaluation processes.

Transcript:

This object class represents the speech-to-text transcription of an interview response. It stores the transcribed text, detected language, and an average confidence score. Transcript objects are generated by the speech-to-text service and are used in the interview evaluation pipeline.

STTService:

This service class is responsible for handling speech-to-text operations in the RecruitAssistant system. It takes audio input from interview responses and produces Transcript objects. This class abstracts the underlying transcription technology and integrates it into the interview workflow.

InterviewEvaluation:

This object class represents the evaluation results of a completed mock interview. It stores an overall score, a summary of the evaluation, and the evaluation timestamp. Each InterviewEvaluation is associated with one MockInterview and acts as the basis for generating user feedback.

FeedbackReport:

This object class represents the final feedback report generated for a user after an interview evaluation. It may store a reference to a generated report file or URL and includes creation metadata. FeedbackReport objects allow users to review performance, scores, and improvement suggestions.

Notification:

This object class represents system notifications delivered to users. It stores notification type (email or in-app), delivery status (queued, sent, or failed), title, message content, and timestamps. Notifications are triggered by important system events such as registration completion or feedback availability.

EmailService:

This service class is responsible for sending email notifications in the RecruitAssistant system. It processes Notification objects, handles communication with the email delivery infrastructure (e.g., SMTP), and updates notification delivery status. This class ensures reliable and consistent outbound communication.

3.5.4 Dynamic Models

3.5.4.1 Sequence Diagrams

3.5.4.1.1 Sequence Diagrams

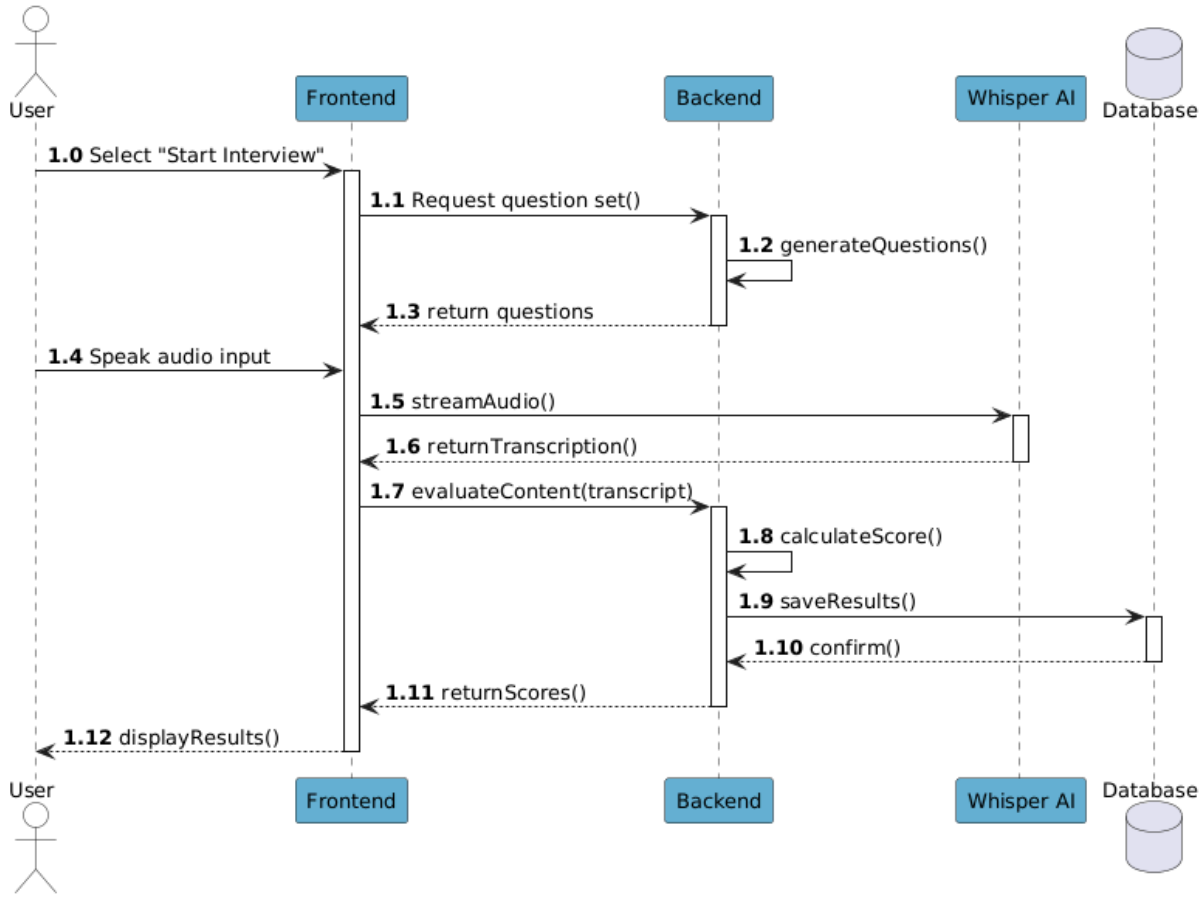

Fig 3: Sequence for Mock Interview

The user starts the interview, and the system loads a set of questions. When the user speaks their answer, an AI tool immediately listens and converts their voice into text. The system then grades that text, saves the results, and shows the user their score and feedback.

3.5.4.2 Sequence Diagrams

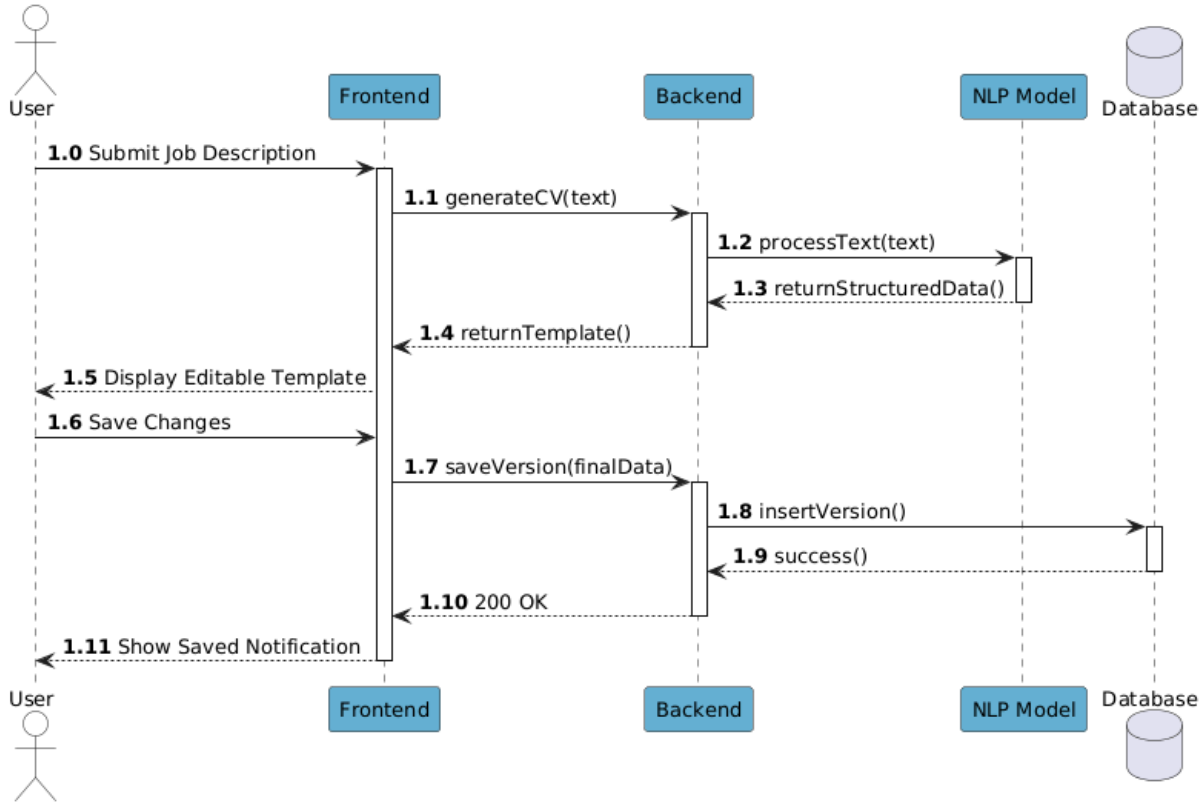

Fig 4: Sequence for CV Generation

The user pastes a job description, and an AI reads it to automatically build a custom CV. The user gets an editable draft on their screen to make any changes. Once they are happy, they click save, and the system stores that specific version of the CV.

3.5.4.2 Activity Diagrams

3.5.4.2.1 Mock Interview

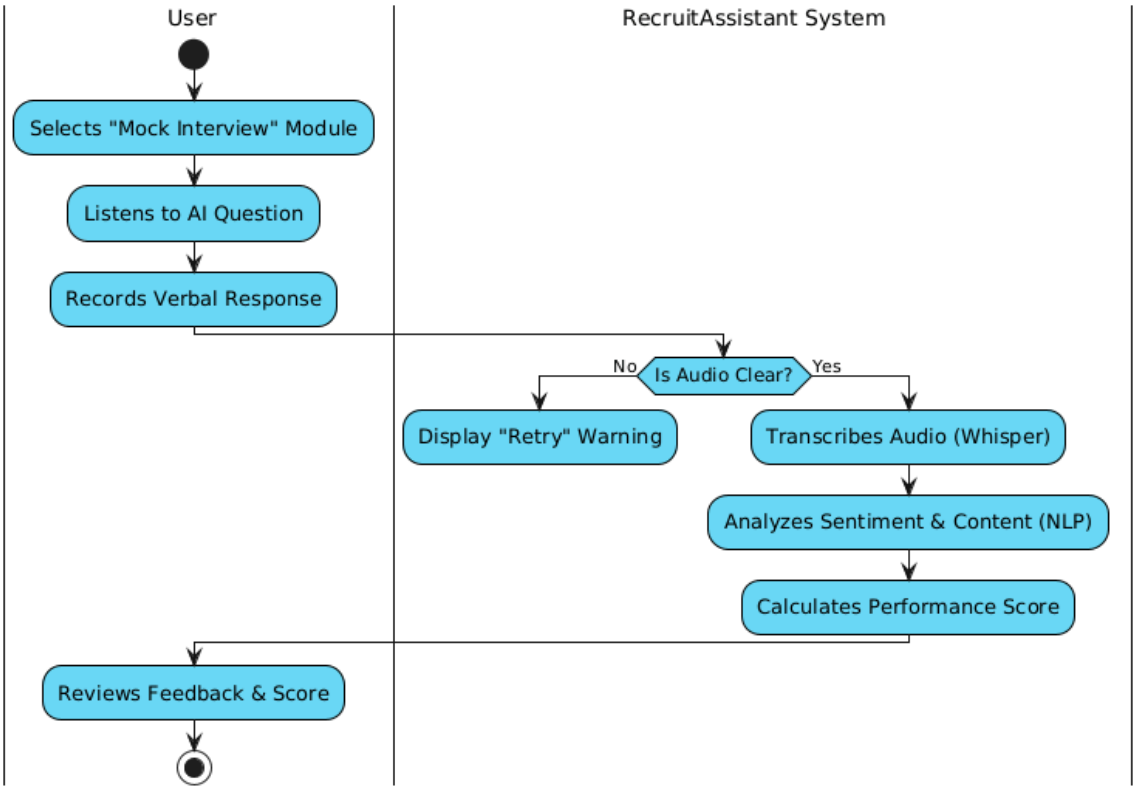

Fig 5: Mock Interview Activity Diagram

The flow of a mock interview session is illustrated in the diagram above. The user initiates the module, prompting the system to generate a context-aware interview question. After the user records and submits a verbal response, the system first validates the audio quality. Upon validation, the Whisper model transcribes the speech into text, and NLP models analyze the content for sentiment and relevance. Finally, the system calculates a performance score and presents a detailed feedback report to the user.

3.5.4.2.2 CV Generation

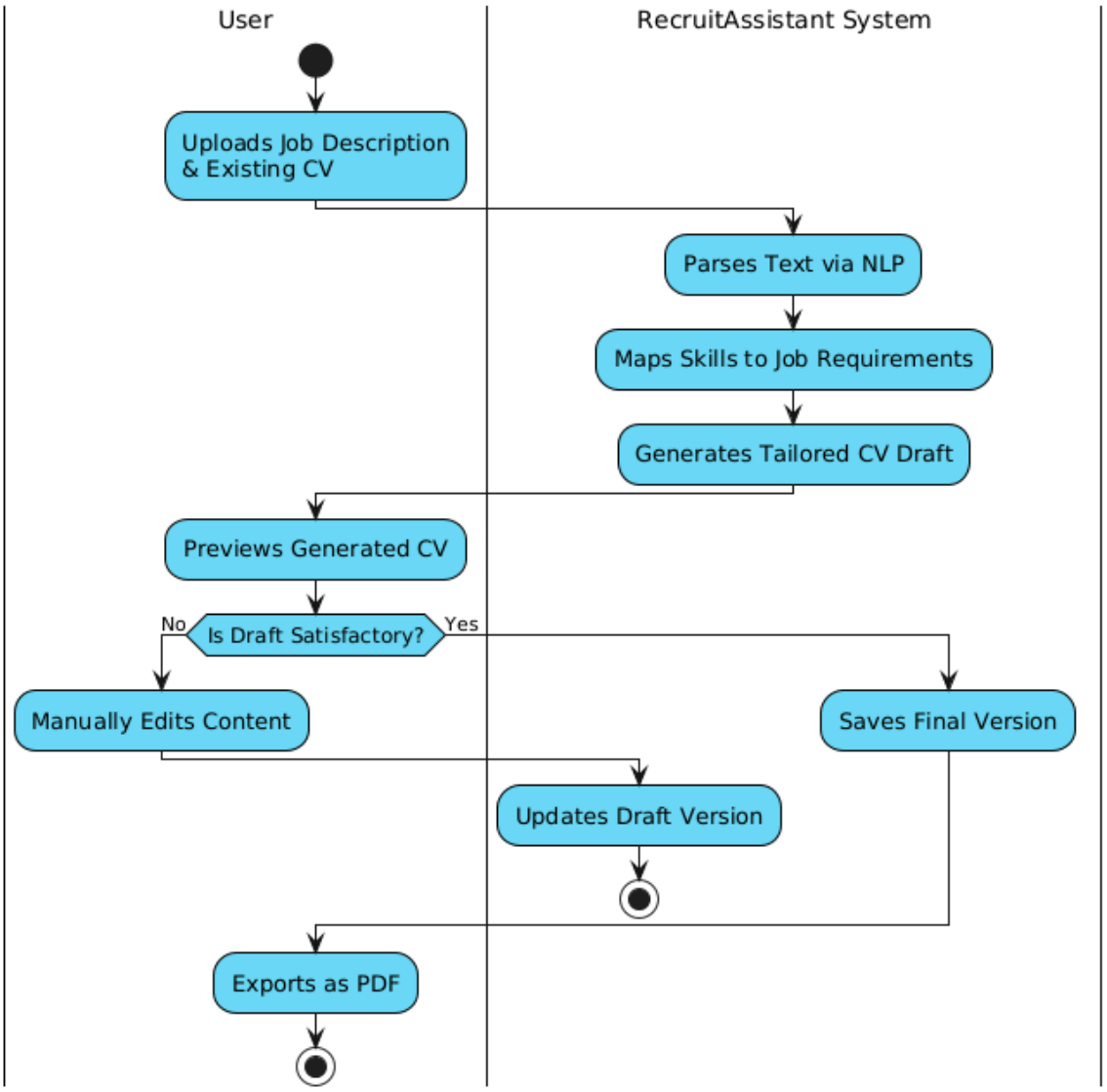

Fig 6: CV Generation Activity Diagram

The AI-powered CV generation workflow is summarized in the figure above. The process begins when the user uploads a target job description along with their existing CV or personal details. The system parses these inputs using NLP algorithms to map the user's skills to the specific job requirements and generates a tailored CV draft. The user reviews the draft; if revisions are needed, they can manually edit the content. Once the draft is approved, the final version is saved to the database, and the user exports the document as a PDF.

3.5.4.2.3 Quiz Generation and Grading

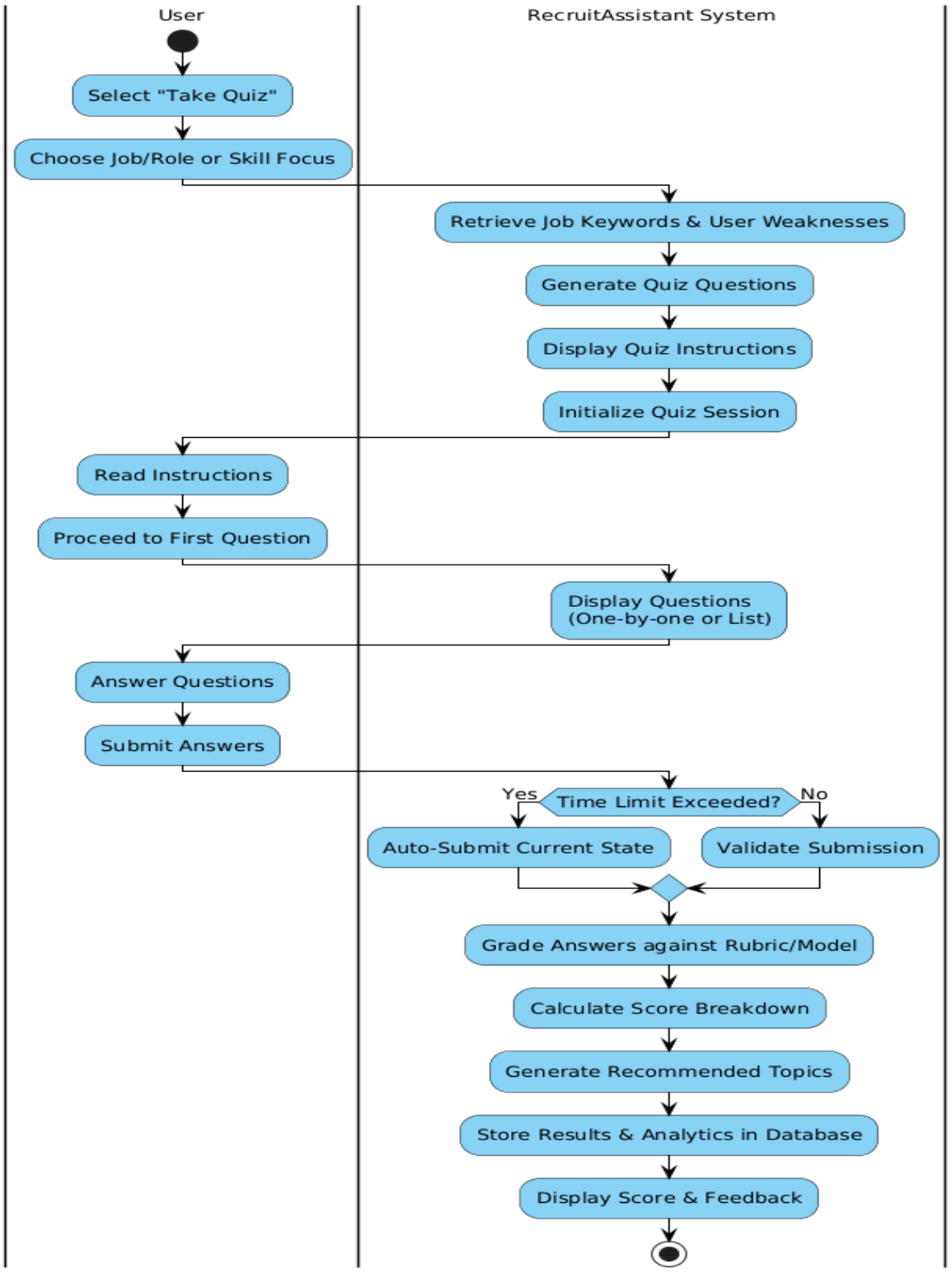

Fig 7: Quiz Generation and Grading Activity Diagram

This process begins when the user selects a specific job role or skill to practice. The system automatically creates a custom quiz based on that role's keywords and the user's previous weak spots. Once the user answers the questions and submits them (or runs out of time), the system instantly grades the quiz, saves the results to the database, and shows the user their score along with recommended topics for improvement.

3.5.4.2.4 User Login

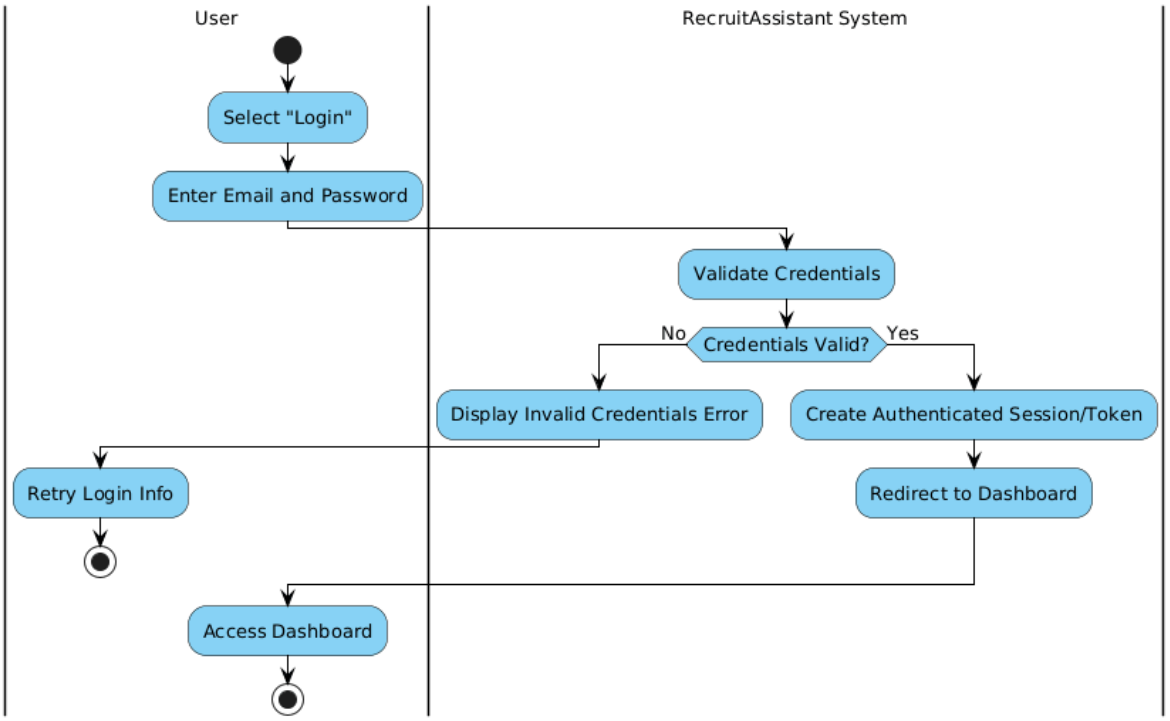

Fig 8: User Login Activity Diagram

The user starts by clicking login and entering their email and password. The system checks these credentials. If the information is incorrect, it shows an error message, asking the user to retry. If the information is correct, the system securely logs the user in and redirects them straight to their main dashboard.

3.5.4.3 State Diagrams

3.5.4.3.1 Quiz Generation

Fig 9: Quiz Generation State Diagram

The quiz lifecycle begins with an analysis phase to identify specific user weaknesses. Based on this data, the AI generates a set of targeted questions. After the user answers the

questions in the progress state, the system grades the submission. The process concludes with a feedback review, where the user is presented with scores and detailed explanations for their answers.

3.5.4.3.2 CV Generation

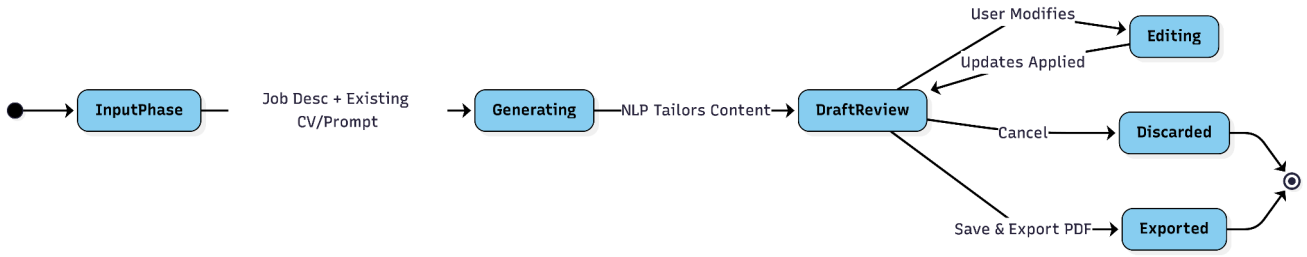

Fig 10: CV Generation State Diagram

The CV generation process initiates with the input phase, where the user submits a job description alongside an existing CV or a self-descriptive prompt. The system utilizes NLP models to generate tailored content based on these inputs. Users can review the draft, enter an editing loop to make modifications, and finally choose to either export the document as a PDF or discard the session.

3.5.4.3.3 Interview Analysis

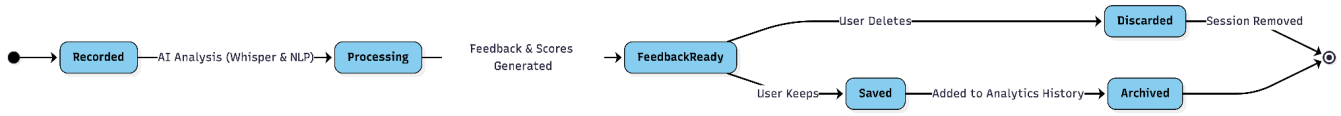

Fig 11: Interview Analysis State Diagram

Once an interview is recorded, the system processes the audio using Whisper and NLP models to generate performance scores and feedback. The user reviews these results in the feedback-ready state. From there, the user makes a decision to either discard the session, removing it permanently, or save it, which adds the data to the analytics history and archives it.

3.5.5 UI Navigation Paths

3.5.5.1 Login Page

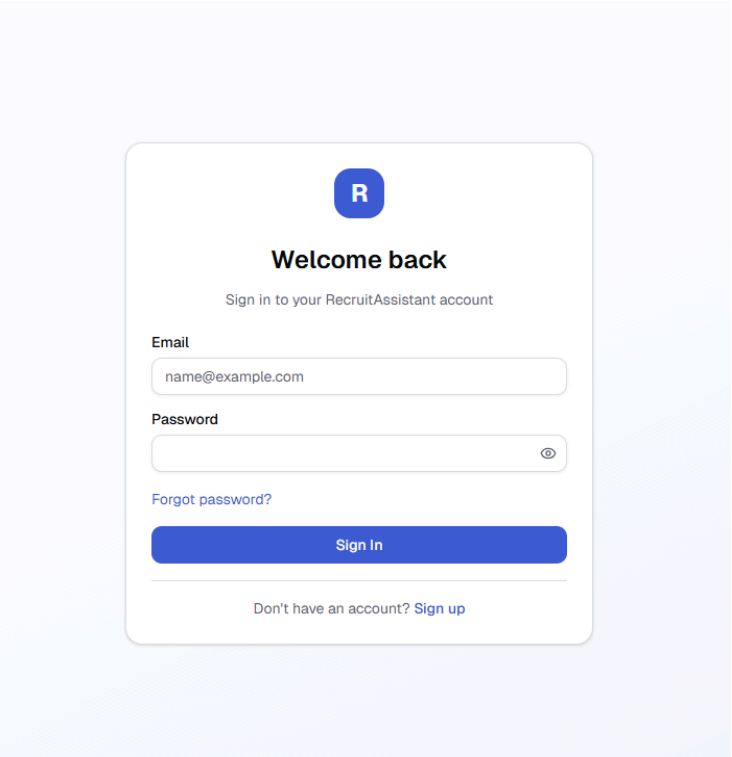

Fig 12: Login Page

The initial login page allows end users to log in to their account or redirect themselves to the sign-up page or the forgot password page.

3.5.5.2 Register Page

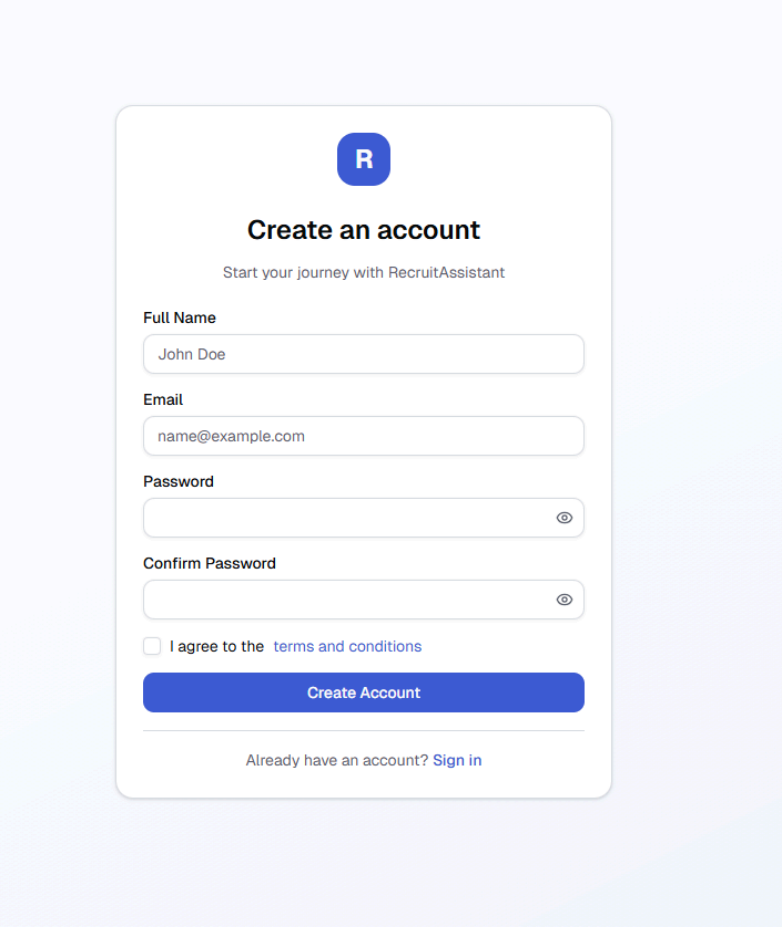

Fig 13: Register Page

The register page for users to create an account consists of name, email, and password fields.

3.5.5.3 Forgot Password Page

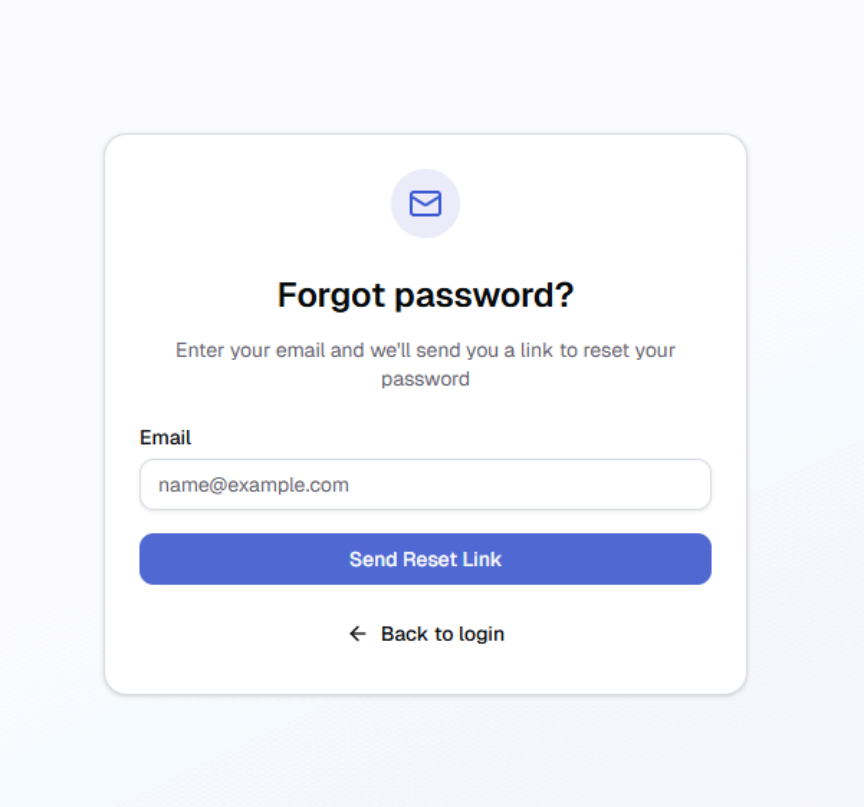

Fig 14: Forgot Password Page

The page for changing the password in case the user forgets their account password.

3.5.5.4 Main Page

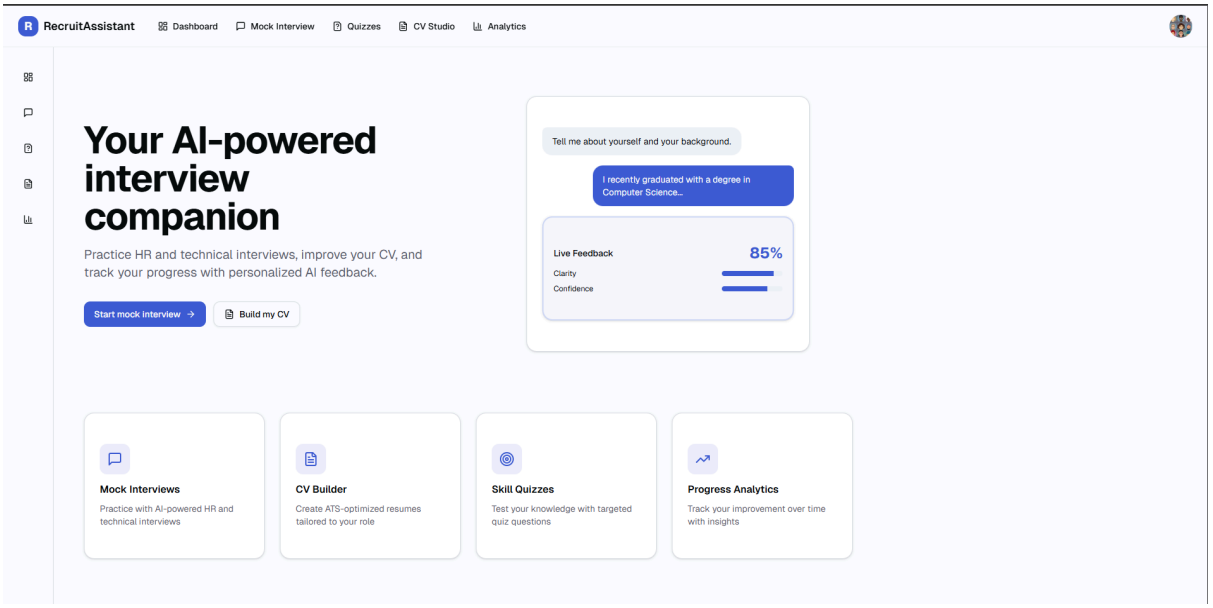

Fig 16: Main Page

The first screen the user sees after entering their credentials. The main page serves as the central access point to all project features, including mock-up interviews and quizzes, and is an introductory page for the resources the project provides.

3.5.5.5 Dashboard

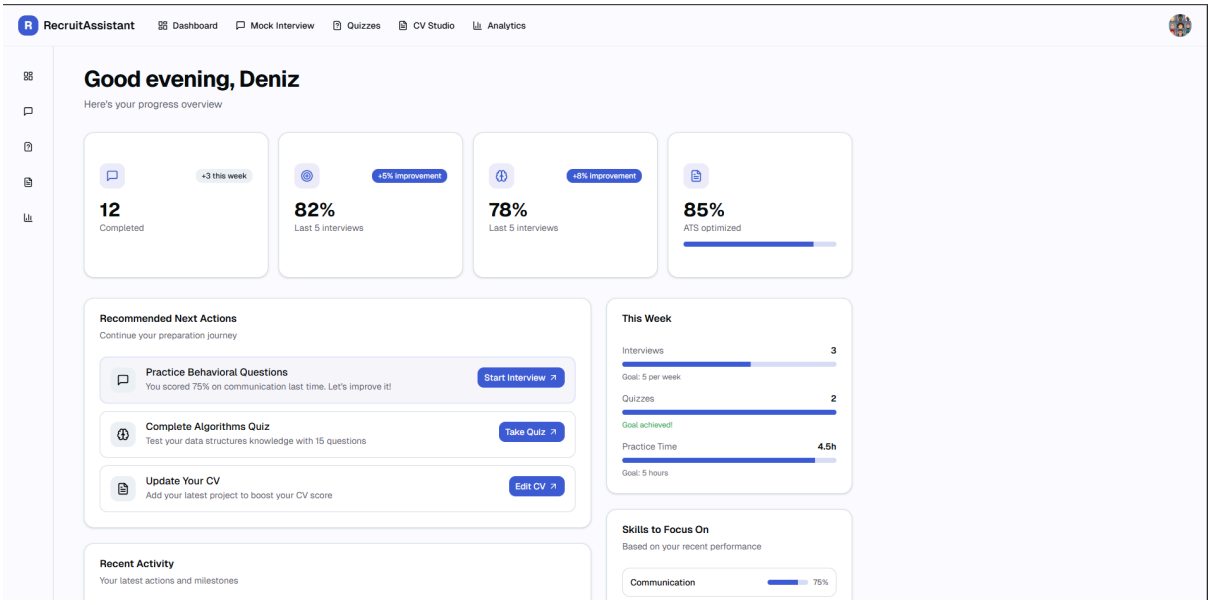

Fig 17: Dashboard

This interface is designed to help users track their performance and quickly access key learning tools.

The screen is organized into sections:

- **Progress Overview:** Showing the current improvements on interviews, quizzes, and CVs.
- **Recommended Next Actions:** Recommendations based on the previous activities and results. Another table showing the skills to improve.
- **Weekly Activity Tracking:** A timeline showing the user's previous activity. Upcoming events are listed as such.

3.5.5.6 Mock Interview

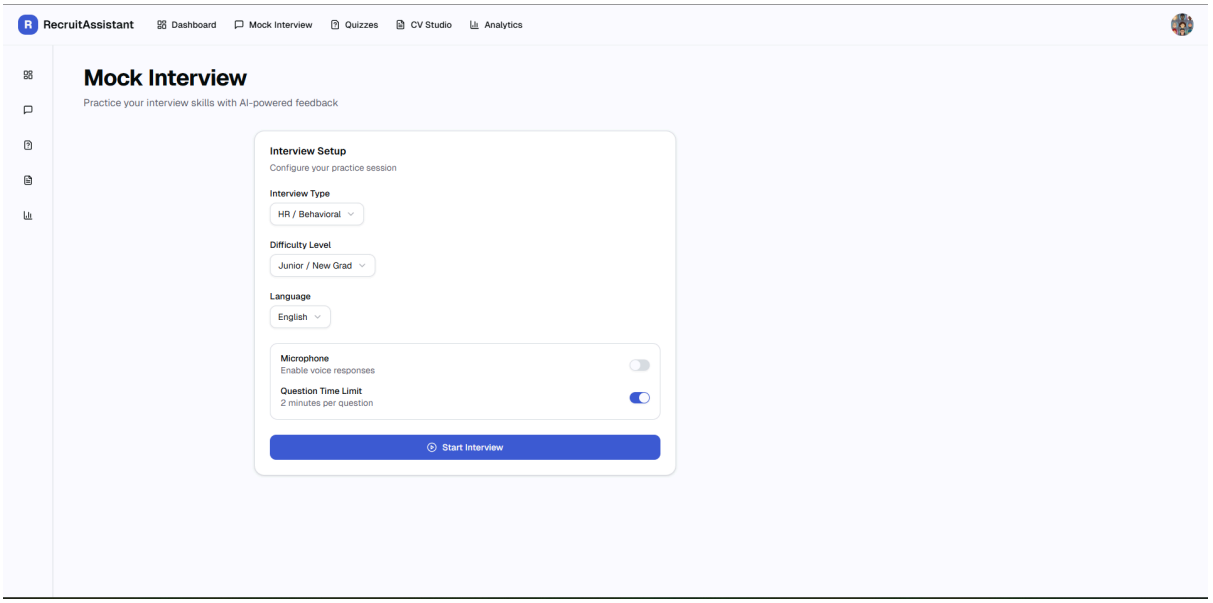

Fig 18: Mock Interview

The Mock Interview Setup screen serves as the configuration hub, where users customize their practice sessions before conducting an interview.

3.5.5.7 Mock Interview Session

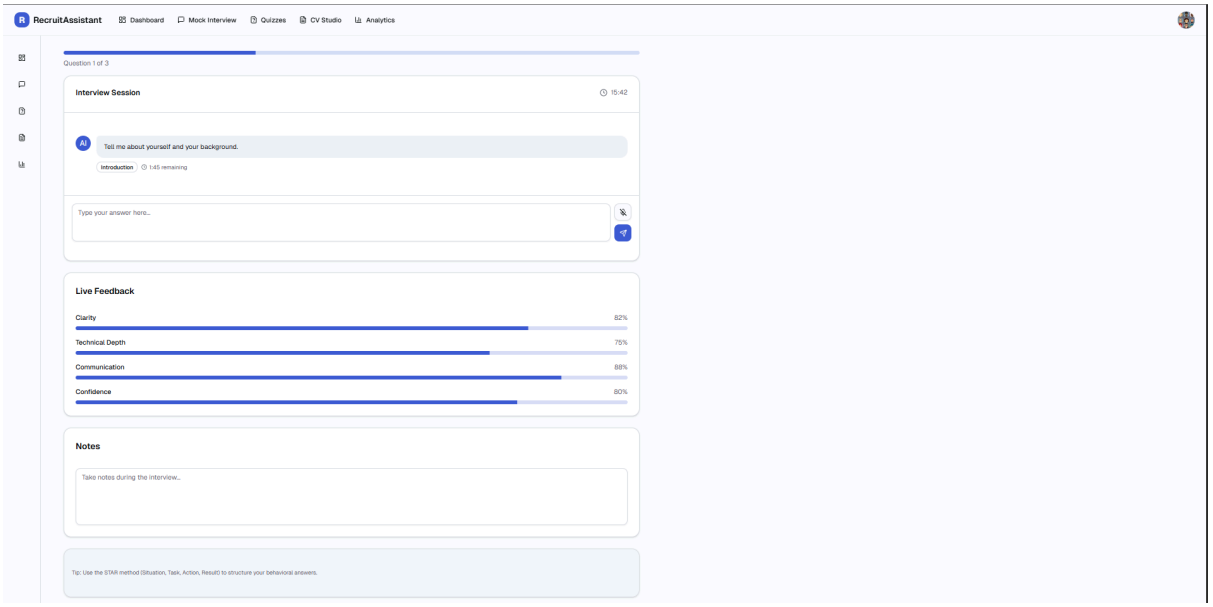

Fig 19: Mock Interview Session

The Active Interview Session screen is the interactive interface where the actual question-and-answer process takes place. It is designed to mimic a chat-based interview while providing real-time data to the user.

3.5.5.8 Mock Interview Feedback

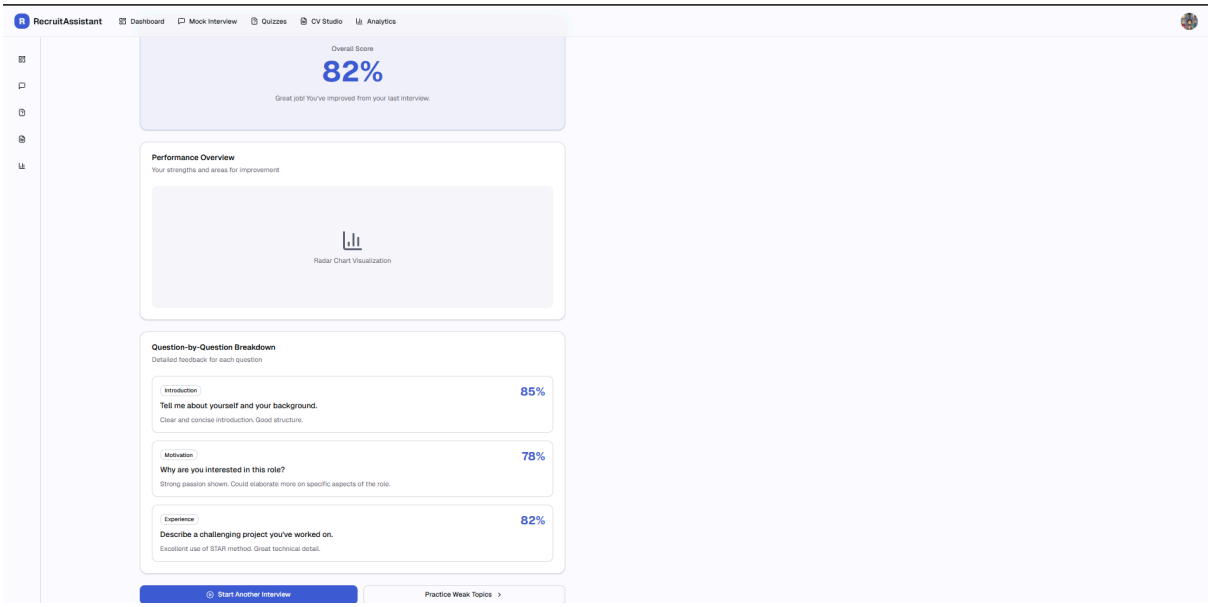

Fig 20: Mock Interview Feedback

The final feedback screen displays the overall score and detailed feedback for the questions asked during the interview.

3.5.5.9 Quizzes

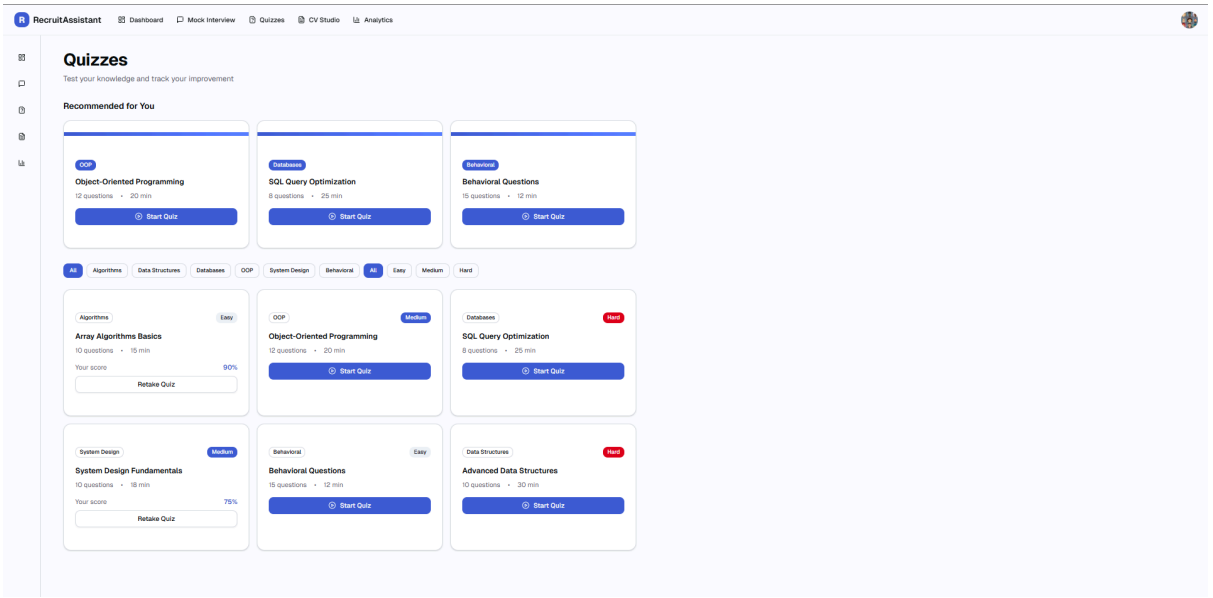

Fig 21: Quizzes

The quizzes section features multiple quizzes covering various topics and differing levels of difficulty. Recommended quizzes based on the previous feedback.

3.5.5.10 Quiz Session

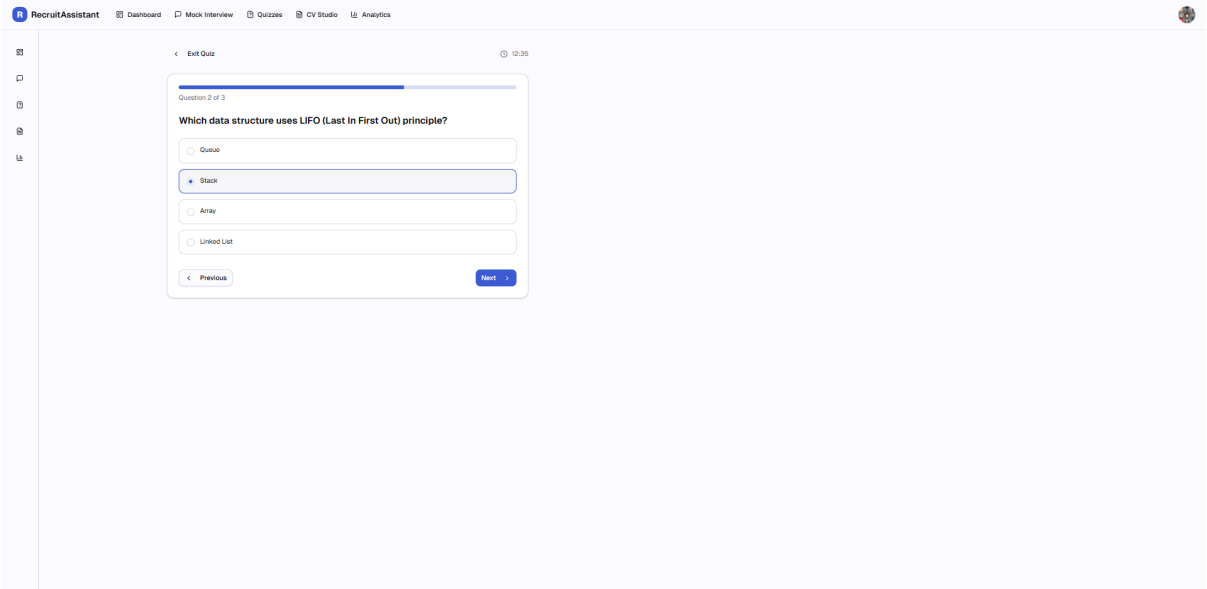

Fig 22: Quiz Session

Multiple-choice quizzes based on the selected topics.

3.5.5.11 Quiz Results Page

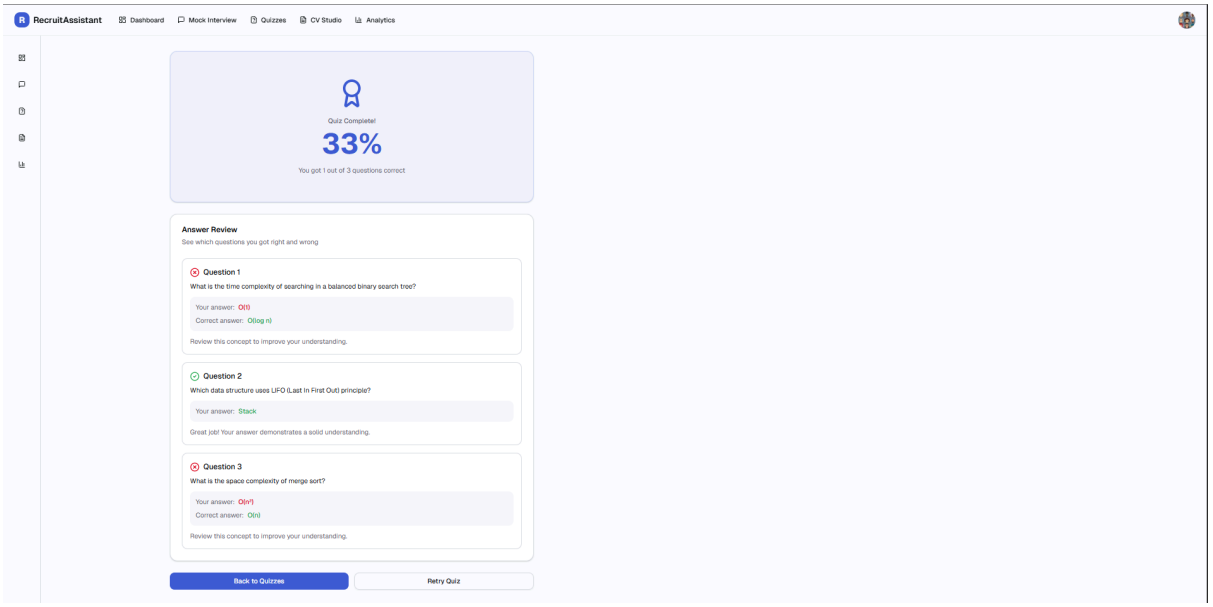

Fig 23: Quiz Results Page

Final results page for the quizzes. This page displays the questions that the user answered correctly and incorrectly. The user can retry the quiz.

3.5.5.11 CV Studio

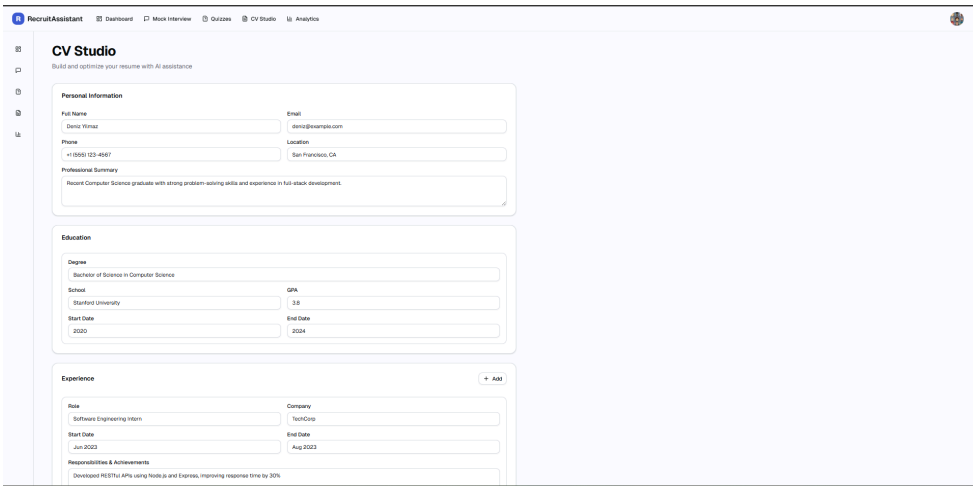

Fig 24: CV Studio

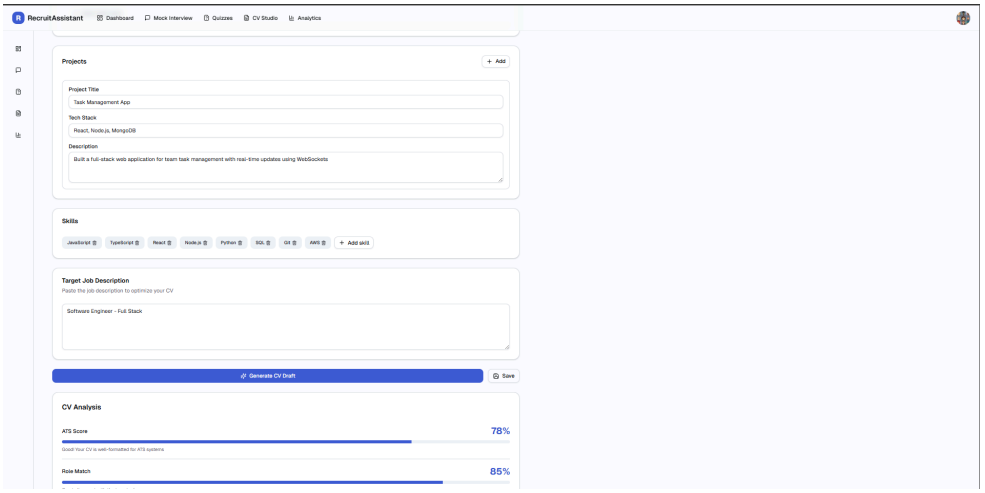

Fig 25: CV Studio

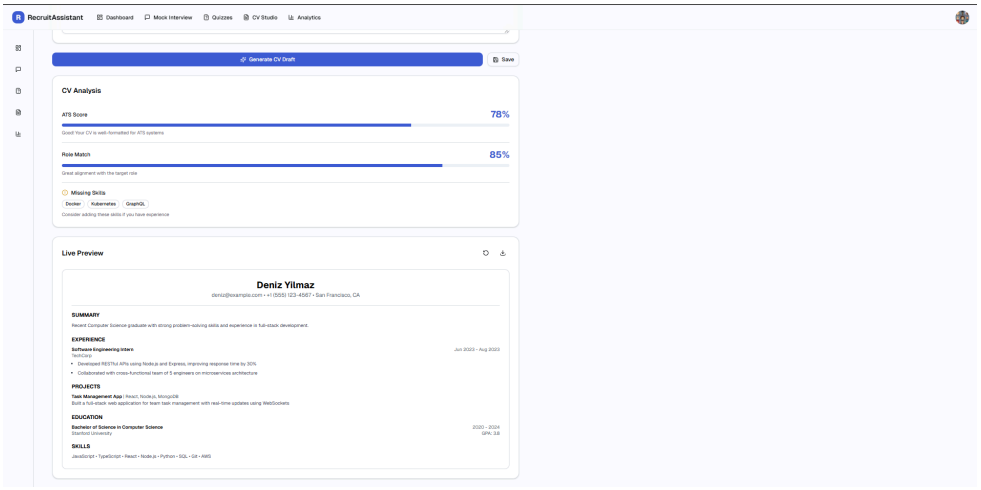

Fig 26: CV Studio

CV Studio is designed for users to create their CVs from scratch. It easily lets the user enter their projects, job experiences etc. Finally, the current CV output is displayed based on the provided information.

3.5.5.12 Analytics page

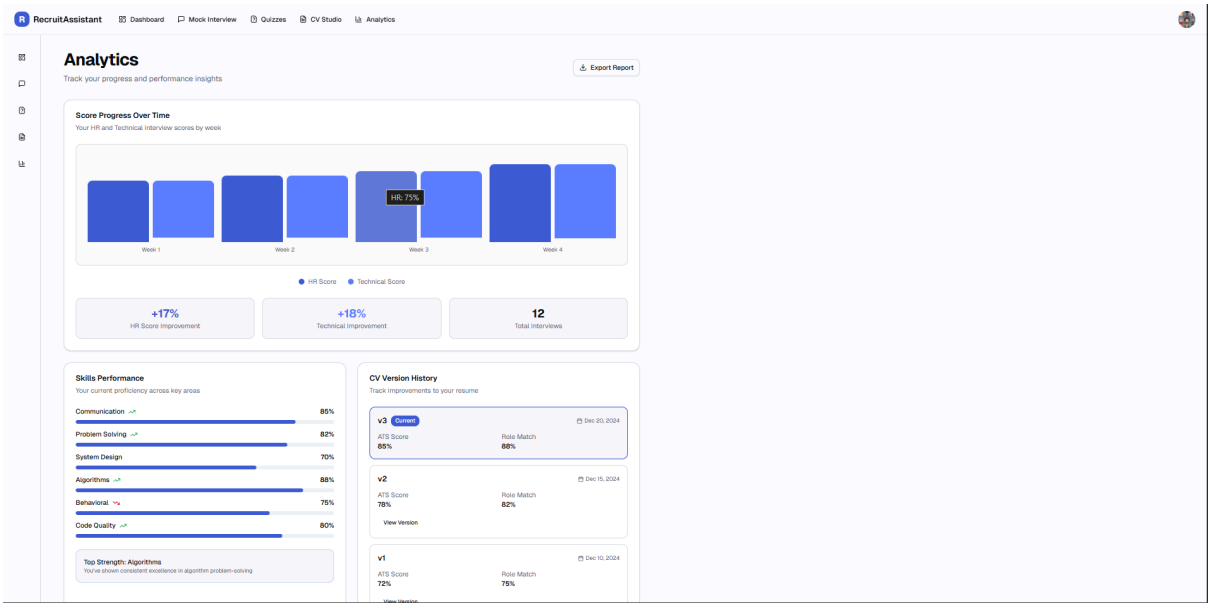

Fig 27: Analytics page

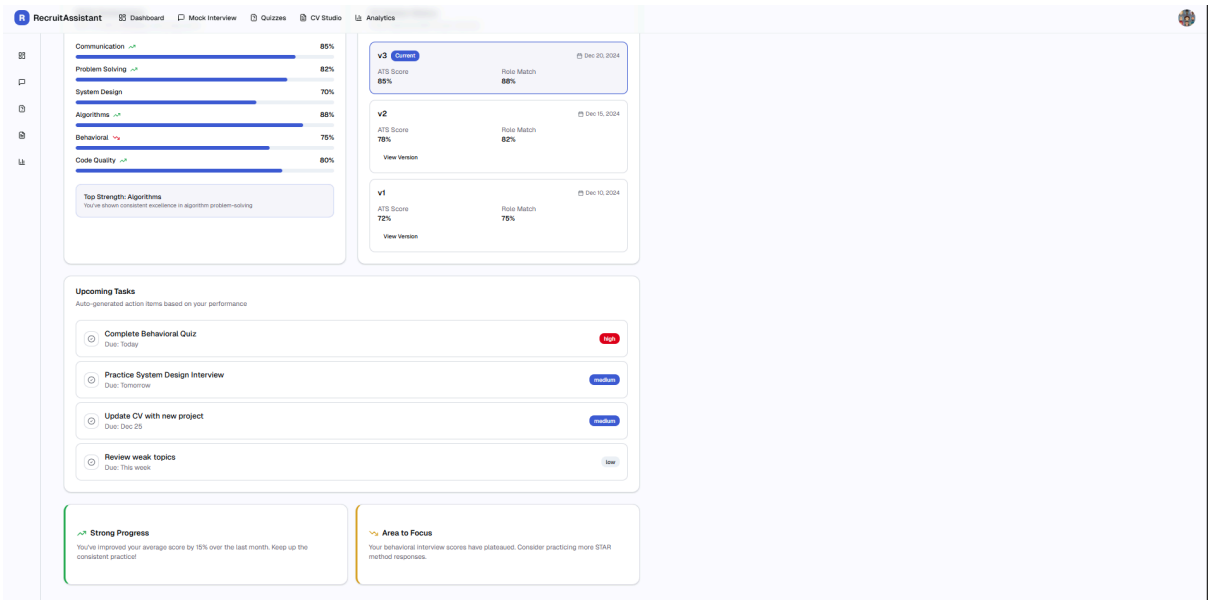

Fig 28: Analytics page

The Analytics page serves as a comprehensive progress report, providing users with a clear view of their improvement over time and the health of their job application materials.

The page is structured to flow from high-level interview metrics down to specific resume details:

- **Progress Visualization:** The top of the page features a "Score Progress Over Time" bar chart. This graph visualizes how the user's HR Score and Technical Score have changed on a weekly basis, allowing them to identify trends in their performance.
- **Skills Breakdown:** Below the main graphs, the Skills Performance section breaks down the user's abilities into specific categories, such as Communication, Problem Solving, and Algorithms. Each skill has a progress bar and a corresponding percentage score, making it easy to identify top strengths and areas that require additional practice.
- **CV Optimization:** A CV Version History list tracks previous drafts and their scores. This leads into a detailed CV Analysis section, which rates the current resume with an ATS Score (checking for compatibility with hiring systems) and a Role Match percentage.

3.5.5.13 Settings

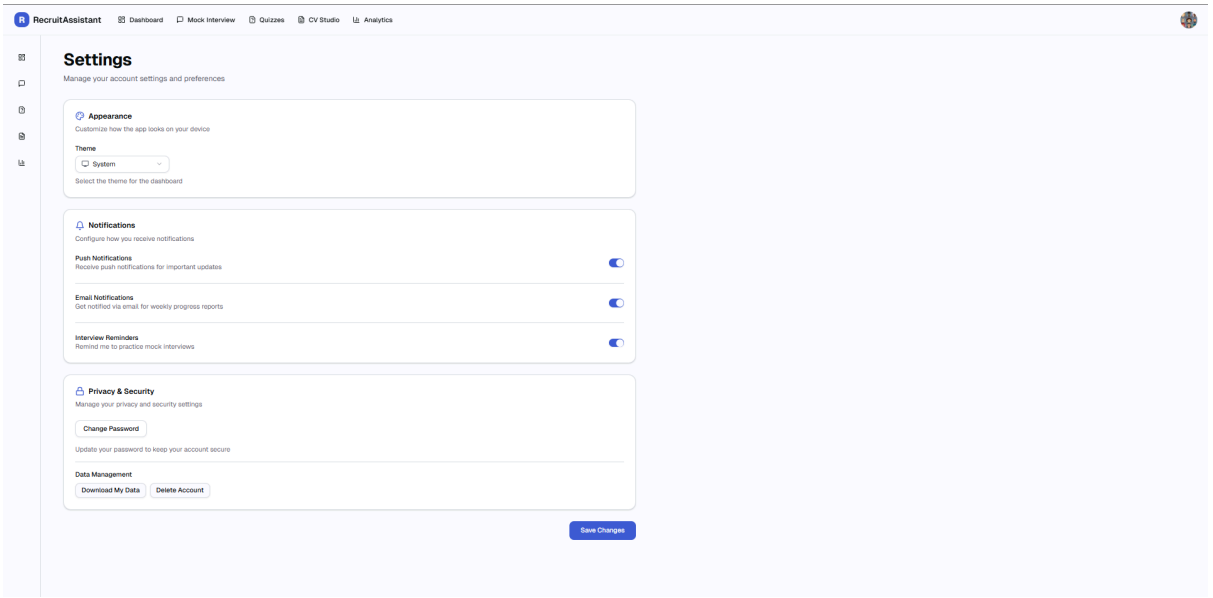

Fig 29: Settings

Setting page giving access to the user for notification specifications, data management, and appearance.

3.5.5.14 Profile

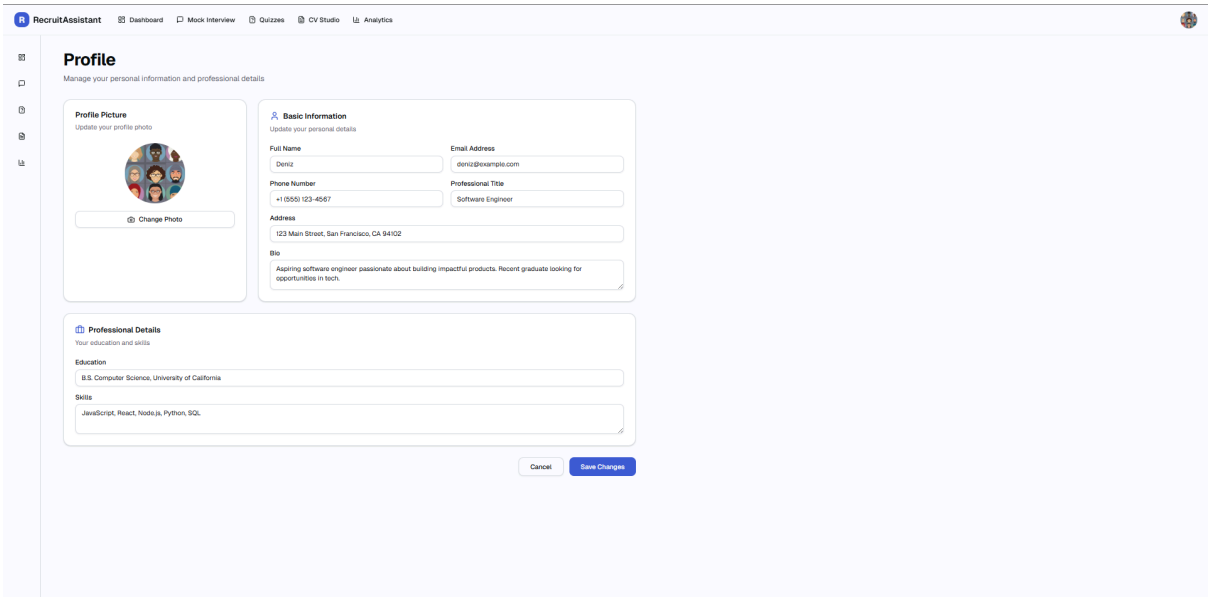

Fig 30: Profile

Profile page for users to update their profile picture, basic information, and professional details.

---

4. Other Analysis Elements

4.1 Consideration of Various Factors in Engineering Design

4.1.1 Constraints

4.1.1.1 Implementation Constraints

- **RecruitAssistant** will be developed as a responsive web application to ensure accessibility across devices.
- **GitHub** will be used as the primary version control system and collaborative environment for the development team.
- The application will be built using Object-Oriented **Programming (OOP)** and modular design principles to ensure code maintainability.
- **AWS S3** will be used to securely store uploaded user content (audio/video), while **PostgreSQL** will serve as the relational database for storing user profiles, CV metadata, and analysis results.
- **The React** framework will be utilized for the user interface design, ensuring dynamic interaction for features like real-time audio recording and dashboard visualization.

- **FastAPI** framework will be used for the backend design due to its high performance, asynchronous capabilities, and ease of integration with Python-based AI models like Whisper.
- **AI models** (Whisper for STT, NLP transformers for evaluation) will be integrated as independent services or modules to maintain system scalability and prevent monolithic bottlenecks.

4.1.1.2 Economic Constraints

- The development process will prioritize **open-source models** (e.g., OpenAI Whisper) and libraries to minimize licensing costs.
- The project will utilize the **AWS Free Tier** for cloud infrastructure (EC2, S3, RDS) wherever possible to keep operational costs zero or minimal during the development phase.
- Since the MVP targets students, the initial business model may rely on university grants or free access, constraining the budget available for high-end GPU servers.
- Scalability costs (e.g., storage fees on AWS S3 or compute costs for inference) must be monitored strictly as the user base grows.

4.1.1.3 Ethical Constraints

- **Consent & Transparency:** Users (students) must be explicitly informed about how their voice data and CV information will be processed and stored. Consent must be obtained before any recording begins.
- **Privacy:** Sensitive data, such as phone numbers or addresses in CVs, must be handled securely. Raw audio files should not be shared with third parties without explicit permission.
- **Data Minimization:** Only essential data required for the evaluation (audio transcripts, CV text, scores) should be retained.
- **Bias Mitigation:** The AI evaluation models must be tested to ensure they do not exhibit bias against different accents, speech impediments, or gender.
- **Right to Delete:** Users must have the option to permanently delete their account and associated data (CVs, interview history) from the system.
- **Decision Support:** The application acts as a guide, not a final decision-maker. Users should be informed that the "employability score" is an AI-generated estimate and not a guarantee of hiring.

4.1.1.4 Social Constraints

- **Accessibility:** The user interface should be intuitive for students with varying levels of technical expertise.
- **Constructive Feedback:** The AI analysis must provide constructive, actionable feedback rather than harsh criticism to avoid negatively impacting the confidence of early-career students.
- **Inclusivity:** The system should cater to students from different engineering backgrounds, although the MVP focuses on CS students.

4.1.1.5 Language Constraints

- **Mock Interviews:** The mock interview module and speech-to-text processing (Whisper) will strictly support the **Turkish** language for the MVP phase.
- **User Interface:** The general application interface (menus, dashboards, settings) will be in **English**, aligning with the medium of instruction at the target universities (METU, Bilkent, Hacettepe).

4.1.1.6 Sustainability Constraints

- **Feedback Loop:** Feedback from beta testers (CS students) will be continuously integrated to maintain the application's relevance and usability.
- **Maintainability:** The codebase will follow strict documentation standards to ensure future teams or developers can maintain the system after the initial project phase.
- **Modular Design:** The separation of the AI engine from the core backend allows for easy upgrades (e.g., swapping Whisper for a newer model) without rewriting the entire system. [4]

4.1.1.7 Environmental Constraints

- **Compute Efficiency:** The inference of heavy AI models (like Whisper) will be optimized (e.g., using quantization or smaller model variants) to reduce energy consumption and computational load.
- **Storage Optimization:** Policies will be implemented to automatically archive or delete old raw audio files after transcription is verified, reducing the storage overhead and associated energy costs of cloud data centers

Factor	Effect Level	Explanation
Public health	Low	This project is not directly correlated with public health outcomes.
Public safety	Low	This project has minimal impact on physical public safety.
Public welfare	Medium	RecruitAssistant contributes to student welfare by reducing unemployment anxiety and improving career readiness.

Global factors	Low	The MVP is localized for specific Turkish universities, though the architecture allows for future global scaling.
Cultural factors	High	The project addresses local recruitment norms in Turkey and processes Turkish language nuances, requiring cultural sensitivity in AI feedback.
Social factors	High	The project democratizes access to career coaching, promoting fairness for students who cannot afford private consulting, but must strictly avoid algorithmic bias.
Environmental factors	Low	The project has minimal physical environmental impact, though cloud compute energy consumption is a minor factor.
Economic factors	Medium	By automating CV checks and interview practice, the tool saves time for students and potentially reduces recruitment screening costs for companies in the long run.

Table 1: Effects of Interalyze on various factors

4.1.2 Standards

4.1.2.1 IEEE 830

The IEEE 830 standard provides guidelines for creating comprehensive and structured Software Requirements Specifications (SRS). For **RecruitAssistant**, this standard emphasizes clarity, completeness, and consistency in documenting functional requirements (such as CV generation and mock interviews) and non-functional requirements (like Whisper latency and system availability). By adhering to IEEE 830, Team T2502 aims to avoid ambiguities in the design process and establish a strong foundation for software validation against the customer's needs.

4.1.2.2 ISO/IEC 25010

The ISO/IEC 25010 standard defines a framework for evaluating software quality. In the context of **RecruitAssistant**, this framework is used to ensure the system meets high standards in **usability** (intuitive UI for students), **performance** (real-time transcription latency), **security** (hashing passwords and encrypting sensitive data), and **maintainability** (modular backend structure). It ensures that the platform is reliable and scalable enough to handle concurrent interview sessions.

4.1.2.3 UML 2.5.1 - Unified Modeling Language

UML 2.5.1 is a standardized visual modeling language used to represent and design software systems. **RecruitAssistant** utilizes UML to create detailed use-case diagrams, sequence diagrams (e.g., for the interview recording flow), and class diagrams. These visual models help the developers communicate the complex system architecture—specifically the interaction between the React frontend, FastAPI backend, and AI services—effectively. [3]

4.1.2.4 Object-Oriented Design (OOD) Principles

Object-Oriented Design (OOD) principles focus on structuring software using notions such as objects, encapsulation, inheritance, and polymorphism to enhance modularity and reusability. By adhering to principles like SOLID, **RecruitAssistant** ensures that its key components—such as the User, CV, InterviewSession, and Analytics classes—are loosely coupled. This approach makes the system easy to maintain, extend with new features (like new language support), and scale on cloud infrastructure.

4.2 Risks and Alternatives

High Computational Requirements: One of the significant risks for the development of RecruitAssistant is the high computational load required for real-time speech-to-text (STT) processing and NLP evaluation. Specifically, the inference of the Whisper model for transcribing Turkish interviews requires substantial memory and processing power. If handled inefficiently, this could lead to high latency (lag), negatively affecting the user experience, or exceed the computational limits of our hosting infrastructure. As a Plan B, we will focus primarily on the optimization of our AI pipeline. An example roadmap for achieving sufficient optimization includes the following steps: First, we will utilize model quantization (e.g., converting models from 32-bit floating-point to 8-bit integers) to significantly improve inference speed and reduce memory usage without a noticeable loss in accuracy. Second, we may switch to smaller, distilled versions of the Whisper model (e.g., distil-whisper or base models) instead of the larger variants for the MVP. Finally, if backend processing becomes too costly or slow, we will explore "client-side inference" alternatives or offload heavy processing to local environments during the demo phase to ensure smooth execution.

Time Management: Another risk that must be considered during the project development cycle is effective time management, particularly given the academic workload of the senior year. Although we have largely conceptualized our deadlines and assigned responsibilities accordingly, unexpected hold-ups can still occur. It is possible that integrating the AI engine

with the backend may take longer than anticipated. Our Plan B for this would be to redistribute tasks dynamically; if one member is blocked, others will assist to clear the bottleneck. Furthermore, we will establish contingency periods (buffers) in the project timeline to account for potential delays. Regular weekly progress reviews will allow us to identify bottlenecks early. If necessary, we will prioritize critical MVP features (CV generation and Mock Interview) over "nice-to-have" components (like advanced admin analytics) to ensure that key milestones are achieved on time.

Consent Issues: Since RecruitAssistant requires recording users' voices for mock interviews and processing their personal CV data, consent is a critical issue. If candidates refuse to provide consent for audio recording, the interview module cannot function as intended. As a fallback plan, we will offer a "Text-Based Mode" where users can type their answers to interview questions instead of speaking. This allows them to still benefit from the NLP evaluation logic without sharing biometric voice data. Additionally, we ensure that users can delete all data immediately after the session if they choose not to save their history. We prioritize building a transparent system that gives users full control over their data retention.

Hardware and Infrastructure Limitations: A potential risk during the development is the inadequacy of free-tier cloud resources (AWS Free Tier) to handle the heavy AI workloads. Running Whisper and Transformer models requires significant GPU power, which is expensive in cloud environments. If our cloud credits or free-tier limits prove insufficient, this could lead to service outages or the inability to demonstrate the product live. To address this risk, our Plan B involves utilizing alternative accessible resources such as **Google Colab** or **Kaggle Kernels** for model training and testing. For the final demonstration, if cloud costs are prohibitive, we will deploy the AI engine locally on team members' high-performance personal computers (with GPUs) and tunnel the traffic to the backend. This ensures that hardware limitations do not compromise the quality of our final presentation and MVP deliverables.

Risk	Likelihood	B Plan
High Computational Requirements	High	Optimization (Quantization) and using smaller Whisper models.
Time Management	Medium	Redistribute tasks and prioritize MVP features.
Consent Issues	Low	Offer a "Text-Only" interview mode or instant deletion options.

Hardware Inadequacies	Medium	Utilize Google Colab or local inference on high-end PCs.

Table 2: Risks

4.3 Engineering Standards

WP#	Work package title	Leader	Members involved
WP1	Project Specification Report	Mehmet Anıl Yeşil	Cankutay Dündar, Yüksel Barkın Baydar, Emir Ensar Sevil, Deniz Öztürk
WP2	Analysis and Requirements Report	Yüksel Barkın Baydar	Cankutay Dündar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk
WP3	Detailed Design Report	Cankutay Dündar	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk
WP4	Final Report	Deniz Öztürk	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Cankutay Dündar
WP5	Presentation and Prototype Demo	Yüksel Barkın Baydar	Cankutay Dündar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk
WP6	UI Design	Emir Ensar Sevil	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Deniz Öztürk, Cankutay Dündar
WP7	Frontend	Cankutay Dündar	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk
WP8	Backend	Mehmet Anıl Yeşil	Cankutay Dündar, Yüksel Barkın Baydar, Emir Ensar Sevil, Deniz Öztürk
WP9	Machine Learning	Emir Ensar Sevil	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Deniz Öztürk, Cankutay Dündar
WP10	Project Management	Cankutay Dündar	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk

WP11	Presentation and Final Demo	Deniz Öztürk	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Cankutay Dündar

Table 3: Work Packages

WP 1: *Project Specification Report*			
Start date: *Oct 10, 2025* **End date:** *Oct 24, 2025*			
Leader:	Mehmet Anıl Yeşil	**Members involved:**	Cankutay Dündar, Deniz Öztürk, Yüksel Barkın Baydar, Emir Ensar Sevil
Objectives: *This work package indicates the design of the analysis and requirements report and how the workload was distributed between the members.*			
Tasks: ***Task 1.1 Use-Case Model*** ***Task 1.2 Object and Class Model Task 1.3 Dynamic Model*** ***Task 1.4 Sequence Diagrams Task 1.5 Mock-up UI Diagrams Task 1.6 Activity Diagrams Task 1.7 State Diagrams*** ***Task 2.1 Explanation of Flow Task 2.2 Constraints*** ***Task 2.3 Risks and Alternatives*** ***Task 2.4 Project Plan*** **36**			
Deliverables: Analysis and Requirements Report			

Table 4: Work Package 1

WP 2: *Analysis and Requirements Report*			
Start date: *Dec 9, 2025* **End date:** *Dec 19, 2025*			
Leader:	Yüksel Barkın Baydar	**Members involved:**	Cankutay Dündar, Deniz Öztürk, Mehmet Anıl Yeşil, Emir Ensar Sevil
Objectives: *This work package indicates the design of the analysis and requirements report and how the workload was distributed between the members.*			
Tasks: ***Task 1.1 Use-Case Model*** ***Task 1.2 Object and Class Model*** ***Task 1.3 Dynamic Model*** ***Task 1.4 Sequence Diagrams Task 1.5 Mock-up UI Diagrams Task 1.6 Activity Diagrams Task 1.7 State Diagrams*** ***Task 2.1 Explanation of Flow Task 2.2 Constraints*** ***Task 2.3 Risks and Alternatives*** ***Task 2.4 Project Plan*** **36**			
Deliverables: Analysis and Requirements Report			

Table 5: Work Package 2

<table>
<tr><td colspan="4">WP 3: Detailed Design Report</td></tr>
<tr><td colspan="4">Start date: Second Semester End date: Second Semester</td></tr>
<tr><td>Leader:</td><td>Cankutay Dündar</td><td>Members involved:</td><td>Mehmet Anıl Yeşil, Deniz , Yüksel Barkın Baydar, Emir Ensar Sevil</td></tr>
<tr><td colspan="4">Objectives: This work package indicates the design of the detailed design report and how the workload was distributed between the members.</td></tr>
<tr><td colspan="4">Tasks:
Task 1.1 - Current software architecture Task 2.1 Proposed software architecture Task 2.1.1 Subsystem decomposition Task 2.1.2 Hardware/software mapping Task 2.1.3 Persistent data management Task 2.1.4 Access control and security Task 2.3 Subsystem services
Task 2.4 Test Cases
Task 2.5 Consideration of Various Factors in Engineering Design Task 2.6 Teamwork Details</td></tr>
<tr><td colspan="4">Deliverables: Detailed Design Report</td></tr>
</table>

Table 6: Work Package 3

WP 4: *Final Report*			
Start date: *Second Semester* **End date:** *Second Semester*			
Leader:	Deniz Öztürk	**Members involved:**	Cankutay Dündar, Deniz Öztürk, Mehmet Anıl Yeşil, Emir Ensar Sevil
Objectives: *This work package indicates the design of the final report and how the workload was distributed between the members.*			
Tasks: ***1. 1. Introduction*** ***1.2. Requirements Details*** ***2.3. Final Architecture and Design Details*** ***2.4. Development/Implementation Details*** ***2.5. Test Cases and Results*** ***2.6. Maintenance Plan and Details*** ***2.7. Other Project Elements*** ***2.7.1. Consideration of Various Factors in Engineering Design*** ***2.7.2Constraints*** ***2.7.3Standards*** ***2.7.1. Ethics and Professional Responsibilities*** ***2.7.2. Teamwork Details*** ***2.8.3. Conclusion and Future Work***			
Deliverables: Final Report			

Table 7: Work Package 4

WP 5: Presentation and Prototype Demo			
Start date: *Dec 14 2025* **End date:** *Dec 24 2025*			
Leader:	Yüksel Barkın Baydar	**Members involved:**	Cankutay Dündar, Deniz Öztürk, Mehmet Anıl Yeşil, Emir Ensar Sevil
Objectives: *This work package indicates the preparation for demo and the presentation*			
Tasks: ***Task 1.1 Frontend for Demo features*** ***Task 1.2 Backend for Demo Features Task 1.2.1 AI Models for Demo Features Task 2.1 Preparing for the Presentation***			
Deliverables: Demo prototype and presentation			

Table 8: Work Package 5

WP 6: UI Design			
Start date: *September 2025* **End date:** *May 15 2026*			
Leader:	*Emir Ensar Sevil*	**Members involved:**	*Cankutay Dündar, Mehmet Anıl Yeşil, Deniz Öztürk, Yüksel Barkın Baydar*
Objectives: *The designing of the UI components of the app*			

Tasks: ***Task 1.1 Login Page*** ***Task 1.2 Register Page*** ***Task 1.3 Forgot Password Page*** ***Task 1.4 Register Page*** ***Task 1.5 Dashboard*** ***Task 1.6 Mock Interview Page 1*** ***Task 1.7 Mock Interview Page 2*** ***Task 1.8 Mock Interview Page 3*** ***Task 1.9 Quizzes Page*** ***Task 1.10 Quiz Session Page*** ***Task 1.11 CV Studio Page*** ***Task 1.12 Analytics Page*** ***Task 1.13 Settings Page*** ***Task 1.14 Profile Page***
Deliverables: UI Mock-up Diagrams

Table 9: Work Package 6

WP 7: *Frontend*			
Start date: *September 2025* **End date:** *May 2026*			
Leader:	*Cankutay Dündar*	**Members involved:**	Mehmet Anıl Yeşil, Yüksel Barkın Baydar, Deniz Öztürk, Emir Ensar Sevil
Objectives: *The frontend code of the projects*			

Tasks: ***Task 1.1 Login Page*** ***Task 1.2 Register Page*** ***Task 1.3 Forgot Password Page*** ***Task 1.4 Register Page*** ***Task 1.5 Dashboard*** ***Task 1.6 Mock Interview Page 1*** ***Task 1.7 Mock Interview Page 2*** ***Task 1.8 Mock Interview Page 3*** ***Task 1.9 Quizzes Page*** ***Task 1.10 Quiz Session Page*** ***Task 1.11 CV Studio Page*** ***Task 1.12 Analytics Page*** ***Task 1.13 Settings Page*** ***Task 1.14 Profile PageTask*** ***Task 3.1 Artistic Design***
Deliverables The full working frontend of the project

Table 10: Work Package 7

WP 8: *Backend*			
Start date: *September 2025* **End date:** *May 2026*			
Leader:	Mehmet Anıl Yeşil	**Members involved:**	Cankutay Dündar, Yüksel Barkın Baydar, Emir Ensar Sevil, Deniz Öztürk
Objectives: *The backend code of the projects*			

Tasks: ***Task 1.1 Login Backend*** ***Task 1.2 Register Backend*** ***Task 1.3 Forgot Password Backend*** ***Task 1.4 Dashboard Data Endpoints*** ***Task 1.5 Mock Interview Backend 1*** ***Task 1.6 Mock Interview Backend 2*** ***Task 1.7 Mock Interview Backend 3*** ***Task 1.8 Quizzes Backend*** ***Task 1.9 Quiz Session Backend*** ***Task 1.10 CV Studio Backend*** ***Task 1.11 Analytics Backend*** ***Task 1.12 Settings Backend*** ***Task 1.13 Profile Backend*** ***Task 2.1 Database Initialization*** ***Task 2.2 Database Models*** ***Task 3.1 Security Protocols and Standards*** ***Task 4.1 Backend Testing***
Deliverables The full working backend of the project

Table 11: Work Package 8

WP 9: *Machine Learning*			
Start date: *September 2025* **End date:** *May 2026*			
Leader:	*Emir Ensar Sevil*	**Members involved:**	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Deniz Öztürk, Cankutay Dündar
Objectives: *Machine learning tasks of the projects including NLP and CV*			

Tasks: ***Task 1.1 Data Collection*** ***Task 1.2 Data Cleaning and Preprocessing*** ***Task 1.3 Label Definition and Annotation*** ***Task 1.4 Interview Feature Extraction*** ***Task 1.5 Text Embedding Pipeline*** ***Task 1.6 Audio Feature Extraction / Speech-to-Text Integration*** ***Task 1.7 Interview Scoring Model Training*** ***Task 1.8 CV Scoring Model Training*** ***Task 1.9 Quiz Recommendation Model Training*** ***Task 1.10 Training Pipeline Implementation*** ***Task 1.11 Hyperparameter Optimization*** ***Task 1.12 Model Validation and Cross-Validation*** ***Task 1.13 Model Explainability and Interpretability*** ***Task 2.1 Dataset Versioning and Storage*** ***Task 2.2 Synthetic / Augmented Data Generation*** ***Task 3.1 Evaluation Metrics and Benchmark Experiments*** ***Task 4.1 Deployment and Inference Testing***
Deliverables ***D1.1 Authentication Backend (Login, Register, Forgot Password) APIs*** ***D1.2 Dashboard Data Endpoints for RecruitAssistant*** ***D1.3 Mock Interview Backend Services (Session Management & Results)*** ***D1.4 Quizzes and Quiz Session Backend Services*** ***D1.5 CV Studio Backend Services (Create, Update, Retrieve CVs)*** ***D1.6 Analytics Backend Endpoints*** ***D1.7 Settings and Profile Management Backend Services*** ***D1.8 Database Initialization Scripts and Database Models*** ***D1.9 Security Configuration and Protocol Documentation*** ***D1.10 Backend Test Suite and Test Report*** *… (the number of the tasks and deliverables could be different)*

Table 12: Work Package 9

WP 10: *Project Management*			
Start date: *September 2025* **End date:** *May 2026*			
Leader:	*Cankutay Dündar*	**Members involved:**	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Deniz Öztürk
Objectives: *This work package indicates the project management phase*			
Tasks: No specific predefined tasks for this work plan			
Deliverables: No specific deliverables for this work plan			

Table 13: Work Package 10

WP 11: *Presentation and Final Demo*			
Start date: *Second Semester* **End date:** *May 2026*			
Leader:	Deniz Öztürk	**Members involved:**	Yüksel Barkın Baydar, Mehmet Anıl Yeşil, Emir Ensar Sevil, Cankutay Dündar
Objectives: *This work package indicates the launch of the final project as well as the presentation of the full working product*			
Tasks: ***Task 1.1 Testing of final*** ***project Task 1.2 Hosting*** ***Task 1.3 Moving from test database to product database*** ***Task 2.1 Preparing the Structure for the final presentation***			
Deliverables: The final fully working prototype as well as the presentation			

Table 14: Work Package 11

4.4 Ensuring Proper Teamwork

To ensure effective teamwork and seamless collaboration throughout the development of **RecruitAssistant**, we follow a structured and organized approach that leverages modern collaboration tools and clearly defined responsibilities. Our strategy combines shared ownership of work packages, continuous code review, and regular synchronization to keep the whole team aligned.

GitHub is used as the **primary collaboration platform**. Each team member works on dedicated branches, opens pull requests, and requests reviews from peers before merging. This workflow ensures accountability, maintains code quality, and makes every change traceable. Commit frequency and pull-request activity are monitored to reflect continuous progress across all work packages.

We hold **weekly team meetings**, during which we review progress, discuss blockers, and refine upcoming tasks. All important decisions, design notes, and meeting outcomes are documented in a shared **Notion workspace**, providing a single source of truth for the team and simplifying onboarding for new contributors if needed.

Responsibilities are **distributed across work packages**, and leadership is shared rather than centralized. While each WP has a designated leader, team members collaborate across boundaries, assist each other when required, and rotate responsibilities when necessary. Our supervisor, **Prof. Özgür Ulusoy**, is included as an observer on GitHub to transparently track project evolution and provide feedback when needed. Finally, peer evaluation and grading are supported through the **Teammates** platform, reinforcing fairness and recognizing individual contributions within the group

4.5 Ethics and Professional Responsibilities

Issue: Users may not clearly understand what data RecruitAssistant collects (CVs, quiz results, interview recordings) and how these will be processed or stored.
Mitigation: Before any data is uploaded, the platform presents a clear consent screen that explains the purpose of data collection, which modules will access it, how long it will be stored, and how users can request deletion. The system is positioned explicitly as a decision-support and learning tool, not as a replacement for human recruiters.

Issue: The ML models used for CV generation, quiz grading, and mock-interview evaluation may unintentionally introduce bias (e.g., against certain genders, universities, or socio-economic backgrounds).
Mitigation: Sensitive attributes are never used as model features, and training datasets are monitored for imbalance. Fairness-aware practices are applied (e.g., bias checks on

evaluation scores), and model outputs are regularly reviewed by the team to detect and correct unfair patterns.

Issue: There is a risk that evaluation scores and feedback could be misinterpreted as "absolute truth" or used in a way that harms users' confidence or career opportunities.
Mitigation: All feedback screens include explanations of what each metric means, confidence levels where applicable, and explicit statements that scores are approximate indicators. Users are encouraged to treat the platform as a practice and coaching environment; recruiters are advised to combine these insights with their own judgment instead of relying solely on automated scores.

Issue: Processing personal career data and interview recordings raises serious privacy and security responsibilities.
Mitigation: User data is protected using secure communication (HTTPS), hashed credentials, and restricted access in the backend. Storage follows GDPR/KVKK principles: data minimization, purpose limitation, and the right to delete one's records. Only authorized team members can access production data for maintenance, and any logs used for debugging are anonymized whenever possible.

Issue: As a CS491/492 project, improper use of third-party code or sources would violate professional and academic ethics.
Mitigation: The team commits to citing all academic papers, open-source libraries, and external datasets, and to respecting license terms. Any reused component is documented in the reports, and original contributions are clearly distinguished from external resources. [7] [8]

4.6 Planning for New Knowledge and Learning Strategies

For RecruitAssistant, we recognize that we must deliberately acquire new knowledge in several areas and link this learning directly to concrete project tasks and milestones. In particular, we need to deepen our understanding of speech-to-text systems (e.g., optimizing Whisper for fast and accurate interview transcription in Turkish and English), transformer-based NLP techniques for evaluating answers, CVs and quizzes, cloud deployment on platforms such as AWS (EC2, S3, managed databases, monitoring), scalable backend design patterns (modular architecture, dependency injection, secure API design), modern React performance practices, and effective data visualization for analytics dashboards. To build this expertise, each team member will follow focused online courses and official documentation for the technologies they own, create small experimental prototypes before integrating new tools into the main codebase, and share what they learn through pair-programming and short internal workshops so knowledge does not remain siloed. We will also use regular meetings with our supervisor to discuss architectural trade-offs and validate our technical choices. By treating these topics as explicit learning goals and continually applying them to RecruitAssistant, we ensure that our technical skills evolve in parallel with the system's functionality and quality.

5. Glossary

1. AI Module: A self-contained component within the backend logic responsible for specific intelligence tasks such as speech transcription, natural language understanding, or question generation.

2. Artificial Intelligence (AI): A branch of computer science that enables machines to perform tasks that typically require human intelligence. In RecruitAssistant, AI is the core driver for evaluating interviews and generating content.

3. Amazon Web Services (AWS): A comprehensive cloud computing platform used by RecruitAssistant for hosting the application (EC2), storing data (S3), and managing databases (RDS).

4. API (Application Programming Interface): A set of protocols and tools that allows different software components to communicate. RecruitAssistant uses RESTful APIs to facilitate communication between the React frontend and the FastAPI backend.

5. Authentication Service: A backend module within FastAPI that verifies user credentials (email/password), manages sessions, and ensures secure access to personalized data like CVs and interview history.

6. Backend: The server-side logic of the application, built using FastAPI. It handles API requests, interacts with the PostgreSQL database, and orchestrates calls to AI models, such as Whisper.

7. Curriculum Vitae (CV): A detailed document highlighting a person's professional and academic history. RecruitAssistant utilizes NLP to automatically generate and tailor these documents based on job descriptions.

8. Dashboard: An interactive visual interface that displays the user's progress, including mock interview scores, quiz results, and skill gap analyses, allowing students to track their improvement over time.

9. Data Retention Policy: Guidelines specifying how long user data (such as interview transcripts and generated CVs) is stored, ensuring compliance with privacy regulations (KVKK/GDPR).

10. Frontend: The user-facing interface of the application, built using React. It allows students to interact with the system by recording interviews, editing CVs, and viewing analytics.

11. GPU (Graphics Processing Unit): A specialized electronic circuit designed to manipulate and alter memory to accelerate the creation of images. In this project, GPUs are crucial for accelerating the inference of the Whisper model, enabling real-time transcription.

12. Machine Learning (ML): A subset of AI that focuses on building systems that learn from data. RecruitAssistant employs ML models to score interview answers and adapt quiz difficulty based on user performance.

13. Minimum Viable Product (MVP): The version of the product with enough features to be usable by early customers (specifically CS students from METU, Bilkent, and Hacettepe) to provide feedback for future development.

14. Mock Interview Module: A core functional component that simulates a real job interview environment. It generates questions, records user audio, and provides feedback without human intervention.

15. Natural Language Processing (NLP): A field of AI that focuses on the interaction between computers and human language. RecruitAssistant uses NLP to parse job descriptions for CV generation and to evaluate the relevance and sentiment of interview answers.

16. PostgreSQL: An open-source relational database management system used by RecruitAssistant to store structured data such as user profiles, interview transcripts, and analytics history.

17. React: A JavaScript library for building user interfaces. It is used to create the responsive web-based frontend of RecruitAssistant, ensuring a smooth user experience across browsers.

18. Scalability: The capability of the system to handle a growing amount of work, such as concurrent interview sessions, by utilizing cloud resources and efficient backend architecture.

19. Security: Measures implemented to protect user data, including password hashing, HTTPS encryption for APIs, and secure storage of sensitive information on AWS.

20. Speech-to-Text (STT): The process of converting spoken language into written text. RecruitAssistant utilizes the Whisper model to transcribe mock interview responses for further analysis and evaluation.

21. Transformer Models: A type of deep learning model adopted in NLP. RecruitAssistant utilizes transformer-based architectures to generate context-aware interview questions and analyze text coherence.

22. User Authentication: The process of verifying the identity of a user connecting to the system, ensuring that each student can only access their own private data and progress reports.

23. Version Control: A system that records changes to a file or set of files over time so that you can recall specific versions later. The team uses GitHub for collaborative development and code management.

24. Web Application: A software program that runs on a web server and is accessed via a web browser. RecruitAssistant is designed as a web application to ensure accessibility for students without requiring high-end local hardware.

25. Whisper: An open-source automatic speech recognition (ASR) system developed by OpenAI. It is the primary engine used by RecruitAssistant for high-accuracy transcription of Turkish interview responses.

6. References

[1] IEEE, “IEEE Recommended Practice for Software Requirements Specifications,” IEEE Std 830-1998, 1998.

[2] ISO/IEC, “Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models,” ISO/IEC 25010:2011, 2011.

[3] Object Management Group (OMG), “Unified Modeling Language (UML), Version 2.5.1,” OMG Specification, Dec. 2017.

[4] R. C. Martin, Clean Architecture: A Craftsman’s Guide to Software Structure and Design. Boston, MA, USA: Prentice Hall (Pearson), 2017.

[5] OWASP Foundation, “OWASP Application Security Verification Standard (ASVS),” OWASP. [Online]. Available: https://owasp.org/www-project-application-security-verification-standard/

[6] OWASP Foundation, “OWASP Top 10 — 2021: The Ten Most Critical Web Application Security Risks,” OWASP, 2021. [Online]. Available: https://owasp.org/www-project-top-ten/

[7] European Union, “Regulation (EU) 2016/679 of the European Parliament and of the Council (General Data Protection Regulation),” Official Journal of the European Union, 2016.

[8] T.C. Resmî Gazete, “6698 Sayılı Kişisel Verilerin Korunması Kanunu (KVKK),” 2016.

[9] A. Radford et al., “Robust Speech Recognition via Large-Scale Weak Supervision,” arXiv:2212.04356, 2022. [Online]. Available: https://arxiv.org/abs/2212.04356

[10] Amazon Web Services (AWS), “AWS Well-Architected Framework,” AWS Whitepaper. [Online]. Available: https://docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html