# MockMate AI

A project report stage I submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer

by

**Altamash Chougle (05)**

**Atharva Harane (15)**

**Darsh Kamble (23)**

**Sneha Singh (64)**

Under the guidance of

**Dr. A. R. Sonule**


Department of Computer Engineering

A. C. Patil College of Engineering, Kharghar, Navi Mumbai

University of Mumbai

2025-2026

Jawahar Education Society's

A. C. Patil College of Engineering, Kharghar

# CERTIFICATE

This is to certify that the project entitled

## MockMate AI

by

**Altamash Chougle (05)**

**Atharva Harane (15)**

**Darsh Kamble (23)**

**Sneha Singh (64)**

is partially completed for the degree of Bachelor of Engineering as prescribed by University of Mumbai.

| | |
|---|---|
| _____ | _____ |
| Guide | HOD |
| _____ | _____ |
| Examiner 1 | Examiner 2 |

_____

Principal

# DECLARATION

We declare that this written submission represents our ideas and does not involve plagiarism. We have adequately cited and referenced the original sources wherever others' ideas or words have been included. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented, fabricated, or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action against us by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: _____

Altamash Chougle (05)

Atharva Harane (15)

Darsh Kamble (23)

Sneha Singh (64)

# Abstract

MockMate AI is a voice-based interview platform that integrates an AI-powered ATS Resume Analyzer and AI-driven mock interviews to improve student placement outcomes. The Resume Analyzer evaluates uploaded resumes against job descriptions to produce an overall ATS score (0–100), highlights matched and missing keywords, flags formatting issues (images, tables, header/footer), checks section correctness (for example, missing Skills section or inconsistent date formats), suggests 6–10 exact bullet phrases drawn from the job description for direct insertion, and auto-generates an ATS-friendly DOCX version with the inserted keywords highlighted. The Interview Module—implemented with a React frontend, React Router, Puter.js client logic, Firebase authentication, and voice agents such as Vapi and Google Gemini—allows students to choose HR, Technical, or Hybrid modes, conduct live voice interviews, receive real-time transcripts and instant AI feedback, and track progress on a centralized dashboard. Complemented by a modern responsive UI/UX and secure authentication, MockMate AI delivers personalized, data-driven feedback and analytics to guide iterative practice. By combining automated resume optimization, simulated voice interviews, and actionable performance metrics, the system aims to bridge the gap between academic preparation and industry expectations, increasing students' confidence and employability.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Background

The transition from college to corporate employment requires both technical capability and effective self-presentation. Despite possessing strong academic and project experience, many students fail to progress through the initial stages of the recruitment process because of resume filtering errors or underdeveloped interview skills. The first obstacle typically arises from automated screening systems — commonly known as Applicant Tracking Systems (ATS) — which reject resumes that do not match the required keywords and structure formats. The second obstacle appears during interviews, where students often struggle to deliver confident, structured, and content-rich responses.

MockMate AI has been conceptualized to address these two challenges through an integrated platform that combines a **Resume Analyzer** and an **AI Interviewer with Feedback**. Instead of incorporating a coding practice module, MockMate AI emphasizes communication and presentation readiness. The Resume Analyzer uses Natural Language Processing (NLP) to assess keyword coverage, structural completeness, and formatting compatibility with ATS software. Meanwhile, the AI Interviewer simulates realistic HR and technical interviews, offering voice- based interaction and personalized, data-driven feedback. This approach directly helps students strengthen their employability profile in measurable ways.

The system aligns with the broader educational goal of enhancing employability by making feedback objective and accessible. Through automated, quantifiable scoring and detailed suggestions, students can progressively refine both their resumes and their interview performances.

Table 1.1: Common ATS Filtering Factors and Their Impacts

| ATS Issue | Impact Level | Example or Description |
|---|---|---|
| Missing Keywords | High | Job-related terms not found (e.g., “React”, “REST API”) |
| Image Elements | Medium | ATS fails to parse embedded images or charts |
| Complex Formatting | High | Multi-column layouts confuse parsers |
| Non-standard Fonts | Low | Unrecognized fonts may reduce readability |

### 1.1.1 Motivation

In modern recruitment, the gap between student capability and industry perception often results from poor presentation rather than lack of skill. A technically sound resume might be discarded before human review simply because of improper formatting or missing domain keywords. Similarly, a capable candidate might underperform in interviews because of limited exposure to realistic, time-bound questioning environments. This situation calls for tools that bridge this employability gap through structured, automated, and intelligent guidance.

MockMate AI directly responds to this gap by focusing on two high-impact areas: optimizing resumes for ATS discoverability and enhancing interview communication through simulated, voice-driven sessions. The motivation behind developing such a system lies in democratizing access to professional-level feedback that is typically available only through paid consultancy or corporate training programs. By automating resume optimization and feedback generation, the system ensures scalability and accessibility for all students.

The motivation also extends to improving placement outcomes in academic institutions. By integrating MockMate AI into pre-placement training programs, colleges can help student’s systematically enhance their job-readiness profiles while providing quantifiable progress metrics over time.

Table 1.2: Example: Resume–JD Keyword Matching Summary

| Keyword / Requirement | JD Count | Matched in Resume |
|---|---|---|
| “React” | 4 | 1 |
| “Firebase” | 2 | 1 |
| “ATS optimization” | 1 | 0 |

### 1.1.2 Significance of Study

The significance of this study lies in its dual focus on real-world recruitment readiness and measurable student improvement. Unlike traditional mock interviews or resume workshops that rely on subjective human evaluation, MockMate AI leverages AI models for objective, consistent scoring and feedback generation. This ensures that every student receives personalized yet standardized evaluation across sessions.

Furthermore, the platform introduces an empirical framework for assessing student growth. The **ATS Score** quantifies resume quality based on keyword match rate and formatting metrics, while the **Interview Feedback Score** captures performance in communication, structure, and content delivery. Together, these indicators establish a quantitative benchmark that institutions can track longitudinally.

The project's broader significance is its contribution to employability research — particularly in the application of AI to career readiness. By combining NLP, voice analytics, and resume optimization, MockMate AI demonstrates how intelligent automation can reduce bias, enhance feedback precision, and ultimately, increase placement success rates.

Table 1.3: Scoring Framework Used in MockMate AI

| **Metric** | **Definition** | **Range / Weight** |
|---|---|---|
| ATS Score | Resume keyword and formatting score | 0–100 (Weighted) |
| Feedback Score | Voice + Content + Structure quality | 0–100 (Weighted) |
| Improvement Index | Score difference between sessions | +/– Range |
| Keyword Density | Percentage of matched JD terms | 0–1 (Normalized) |

Table 1.4: Interview Evaluation Components

| **Component** | **Description** |
|---|---|
| Content Quality | Relevance and correctness of response content |
| Voice Fluency | Smoothness and pacing of spoken responses |
| Answer Structure | Logical flow (introduction, body, conclusion) |
| Keyword Usage | Incorporation of job-relevant terms |
| Confidence Level | Detected energy and tone modulation |

## 1.2 Project Scope

The scope of MockMate AI is carefully defined to concentrate on the most impactful components of placement preparation: resume analysis and interview readiness. It intentionally excludes coding practice environments or domain-specific technical tests to maintain clarity of focus and resource efficiency. The system's scope extends from individual users (students) to institutional usage in pre-placement labs, allowing both self-paced and guided practice.

Within this defined scope, the platform integrates two primary modules:

- **Resume Analyzer:** Conducts keyword extraction, formatting validation, and automatic DOCX generation with suggestions.
- **AI Interviewer:** Simulates voice-driven interview sessions and provides structured feedback across HR, technical, and hybrid modes.

The system thereby serves as a holistic yet lightweight placement readiness tool that complements existing institutional resources.

## 1.3 System Overview

The MockMate AI system architecture follows a web-based modular design that integrates a React frontend with Firebase for authentication, storage, and hosting. The backend, implemented using FastAPI or Express, handles NLP parsing, ATS scoring, and voice analytics. Cloud-based APIs such as Google Gemini and Vapi SDKs provide conversational inference and real-time speech processing. The resulting architecture supports secure communication between client and server while maintaining scalability through serverless deployments.

Data flows through clearly defined stages — from resume upload to ATS scoring, followed by interview execution, feedback computation, and dashboard visualization. Each component communicates via REST APIs, ensuring interoperability and modular extensibility for future versions.

## 1.4 Methodology and Key Metrics

The methodology of MockMate AI revolves around two distinct yet connected pipelines: **NLP-based Resume Analysis** and **Speech-based Interview Feedback**. Both pipelines rely on machine learning models trained or fine-tuned for task-specific outcomes.

Key evaluation metrics are mathematically formalized as:

$$ATS\_Score = 100 \times \left( \alpha \frac{K_{\text{matched}}}{K_{\text{total}}} + \beta F_{\text{format}} \right) - P_{\text{penalty}}$$

and

$$Feedback\_Score = w_c S_{\text{content}} + w_v S_{\text{voice}} + w_s S_{\text{structure}}$$

where parameters denote the weights of content, voice, and structure scores respectively.

## 1.5 Objectives

The objectives of MockMate AI are both technical and educational. Technically, the system aims to create a high-performing, cloud-integrated resume analyzer and AI interviewer. Educationally, it seeks to enhance employability by offering measurable, iterative feedback loops.

- Develop a keyword-based NLP pipeline for accurate ATS scoring and resume optimization.
- Implement a voice-based AI interviewer for realistic, automated practice.
- Provide exportable ATS-friendly DOCX resumes with inserted recommendations.
- Enable historical tracking and performance visualization through a centralized dashboard.

## 1.6 Expected Outcomes and Limitations

### 1.6.1 Expected Outcomes

The proposed system, **MockMate AI**, is expected to significantly enhance students' employability by improving both resume quality and interview readiness. Through advanced Natural Language Processing (NLP) and Machine Learning (ML) techniques resumes are not rejected

by automated company screening tools due to formatting or keyword issues. The feedback mechanism guides users in restructuring their resumes with relevant action verbs, quantifiable results, and improved readability, which collectively increase their chances of being shortlisted.

Another major outcome is the improvement in interview performance. The AI-driven mock interview module provides students with realistic simulations of HR and technical interview scenarios. During these sessions, the system evaluates multiple parameters such as tone, fluency, response structure, and confidence. Detailed feedback reports, including performance scores and improvement tips, help students identify weak areas and refine their responses over repeated sessions. Over time, this leads to measurable growth in communication clarity, confidence, and professionalism during real interviews.

### 1.6.2 Limitations

While the system provides significant benefits, certain limitations must be acknowledged. The accuracy and responsiveness of MockMate AI depend heavily on third-party NLP and speech recognition APIs. Any latency, network interruption, or API downtime could impact the smoothness of the experience. Furthermore, despite being trained on diverse datasets, the AI may occasionally misinterpret accents or non-standard speech patterns, leading to minor inaccuracies in transcription or analysis. Another limitation arises from the scope of AI evaluation. The feedback generated by the system, though detailed, cannot fully capture nuanced human aspects of interviews such as empathy, interpersonal warmth, or situational judgment. Human interviewers often evaluate candidates based on subtle cues like emotional intelligence, body language, or adaptability, which AI systems cannot yet reliably assess. Thus, the feedback provided should be considered as an assistive guide, not an absolute measure of readiness.

Lastly, while MockMate AI accelerates self-assessment and preparation, it is not intended to replace personalized mentorship or domain-specific learning. Students should complement AI-based feedback with real-world practice, participation in workshops, and expert guidance.

# Chapter 2

# Literature Survey

This chapter reviews prior research in AI-driven interview systems and resume analysis. It explores existing multimodal approaches, highlights their pros and cons, and identifies critical limitations that shape the foundation of *MockMate AI*. The discussion begins with an overview of existing systems, followed by previous work, gap analysis, and concludes with the formal problem statement motivating this study.

## 2.1 Existing System

Over the past decade, significant progress has been made in developing AI-based interview simulation and resume evaluation systems. Researchers have integrated Natural Language Processing (NLP), Machine Learning (ML), and multimodal sensory analysis (voice, text, and visual cues) to improve candidate evaluation and feedback. The 2025 study by Lokesh Jagtap *et al.* introduced a large language model–driven mock interview platform capable of evaluating linguistic and visual behaviors. This system demonstrated promising results in generating adaptive questions and offering instant feedback through real-time dashboards. However, its dependence on high-quality cameras and high computational power limited scalability in low-resource environments.

Similarly, the 2024 work by Nagasawa and colleagues explored adaptive interview strategies for social robots. Their approach used gaze detection, voice modulation, and engagement metrics to dynamically alter questioning patterns. The 2023 paper by Changwoo Kim *et al.* proposed a fairness-aware learning model that fuses audio and visual data while mitigating demographic bias. Although successful in reducing algorithmic bias, the method's heavy reliance on large annotated datasets made it resource-intensive. Similarly, the 2019 TensorFlow-based model by

Hung-Yue Suen *et al.* demonstrated automatic personality recognition using CNNs. *A Practitioner's Approach* (2010), continue to influence system design. Pressman emphasized structured software development and system lifecycle models, which remain essential for designing scalable AI platforms. However, these conventional frameworks lack adaptability for dynamic, cloud-native AI environments.

Table 2.1: Summary of Key Research Papers: Pros and Cons

| Research Paper | Pros | Cons |
|---|---|---|
| **AI-Based Interview Mocker System (2025)** | Integrates NLP and LLMs for real-time multimodal feedback and adaptive question generation. | High computational cost; requires stable network and camera quality. |
| **Adaptive Interview Strategy for Robots (2024)** | Enhances engagement using gaze, speech, and facial cues for adaptive questioning. | Implementation complexity due to real-time multimodal fusion and dataset dependency. |
| **Fairness-Aware Multimodal Learning (2023)** | Introduces bias-mitigation models with adversarial training in multimodal learning. | Requires extensive annotated datasets; performance trade-offs between accuracy and fairness. |
| **TensorFlow-Based Personality Recognition (2019)** | Maps audio-video data to personality traits; supports behavior-based feedback. | Risk of overfitting and ethical concerns regarding personality inference. |
| **Software Engineering: A Practitioner's Approach (2010)** | Provides structured system design and validation principles. | Lacks AI-specific and real-time architectural adaptability. |

## 2.2 Limitations of Existing Systems

Despite advancements, existing systems exhibit key limitations. Most multimodal models demand high-end hardware and computational resources, restricting use in educational and low-bandwidth settings. Many studies also emphasize accuracy over interpretability, making feedback difficult for students to act upon. Furthermore, integration between ATS-based resume screening and AI interview analysis remains limited. Current models either focus exclusively on textual resume optimization or multimodal behavioral analysis, rarely offering an end-to-end employability enhancement pipeline. These constraints underscore the need for a cost-efficient, explainable, and scalable AI system suitable for campus-level deployment.

## 2.3 Previous Work

Recent literature has moved from purely text-based interview evaluation to multimodal systems that combine language models with audio and visual sensing. Multimodal mock-interview systems achieve richer feedback (e.g., posture, eye contact, vocal prosody) and can adapt question trees dynamically based on candidate engagement. Parallel work in ATS research emphasizes robust keyword extraction, formatting robustness, and generation of ATS-friendly documents to improve shortlisting rates. Fairness and interpretability are recurrent concerns — systems that leverage visual or personality inferences must expose reasoning and mitigate biased outcomes.

## 2.4 Gap Analysis

From the surveyed works and their limitations, we identify gaps that inform the design of *Mock-Mate AI*:

1. **High-cost, multimodal dependency:** Many top-performing systems rely on high-quality video streams and large compute resources — a barrier for campus deployment.

2. **Complexity vs. Practicality:** Adaptive multimodal sensing yields richer signals but increases implementation complexity, data needs, and latency.

3. **Fairness and interpretability:** Vision- or personality-based scoring risks bias; systems need conservative, explainable feedback that students can act upon.

4. **ATS–Interview integration:** Few works combine a strong ATS resume optimization pipeline with an interactive voice-based interviewer that gives prescriptive, insertion-ready suggestions for resumes.

## 2.5 Problem Statement

Existing AI-based mock interview and resume analysis tools fail to deliver a unified, accessible, and fair employability enhancement solution for students. Most available systems either emphasize multimodal precision at the cost of accessibility or limit functionality to resume keyword optimization without addressing interview performance. This disconnects leads to fragmented preparation experiences where candidates improve resumes but remain underprepared for real- time interview interactions. A critical gap exists in developing a low-cost, cloud-deployable

platform that integrates both ATS-based resume enhancement and AI-driven voice interviews. Such a system must balance performance accuracy with resource efficiency, ensuring accessibility for students across varying socio-technical backgrounds. Moreover, feedback generated should be interpretable and actionable, emphasizing clarity over complexity.

The system should also ensure fairness by avoiding demographic or linguistic biases. Many personality-prediction models unintentionally encode bias from training datasets, creating un-equal assessments. A responsible design must prioritize transparency and inclusivity, offering every user a fair evaluation irrespective of accent, gender, or background.

Finally, existing systems often lack analytical tools for educational institutions. There is a growing need for aggregated performance dashboards that provide placement cells with data-driven insights into students' progress. This empowers institutions to tailor soft-skill development programs and monitor readiness trends effectively.

# Chapter 3

# Proposed System

## 3.1 Introduction

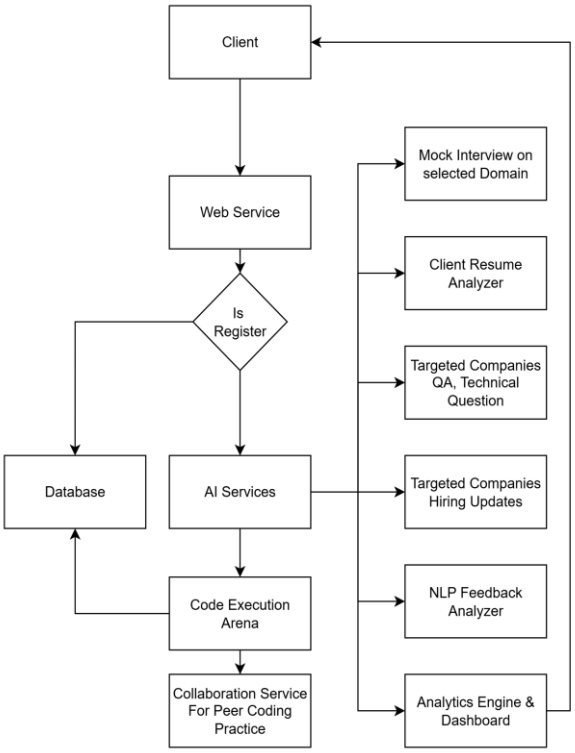
This chapter describes the proposed system for **MockMate AI** — a lightweight, scalable assistant for resume parsing, ATS-style scoring, and voice-based mock interviews. The design balances a developer-friendly local setup for testing with cloud-enabled services for production (LLMs, STT/TTS, storage), and emphasizes modularity so components (frontend capture, backend processing, ML modules and storage) can be swapped independently. The sections that follow present the system design, diagrams that visualize interactions and data flow, hardware and software requirements, system development notes and expected outputs, possible extensions for future work, and concluding remarks summarizing strengths and trade-offs.

## 3.2 System Design

This section gives an overview of the system design with diagrams and short explanations. The design is modular: the frontend captures audio and documents, a backend provides parsing, ML/inference services and persistence, and cloud APIs (LLM, STT/TTS) are used where higher accuracy or scalability is required.

### 3.2.1 System Architecture Overview

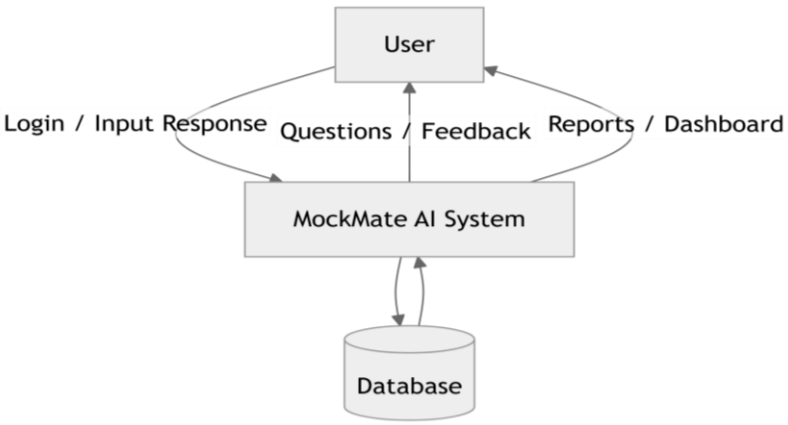
Figure 3.1: System Architecture Diagram




The architecture diagram (Figure 3.1) shows the primary components and their relationships: user-facing frontend (browser or Vapi SDK), backend APIs (FastAPI/Express), storage (Firestore / Cloud Storage), ML services (resume parser, embedder, ATS scorer), and third-party APIs (LLM, STT/TTS).

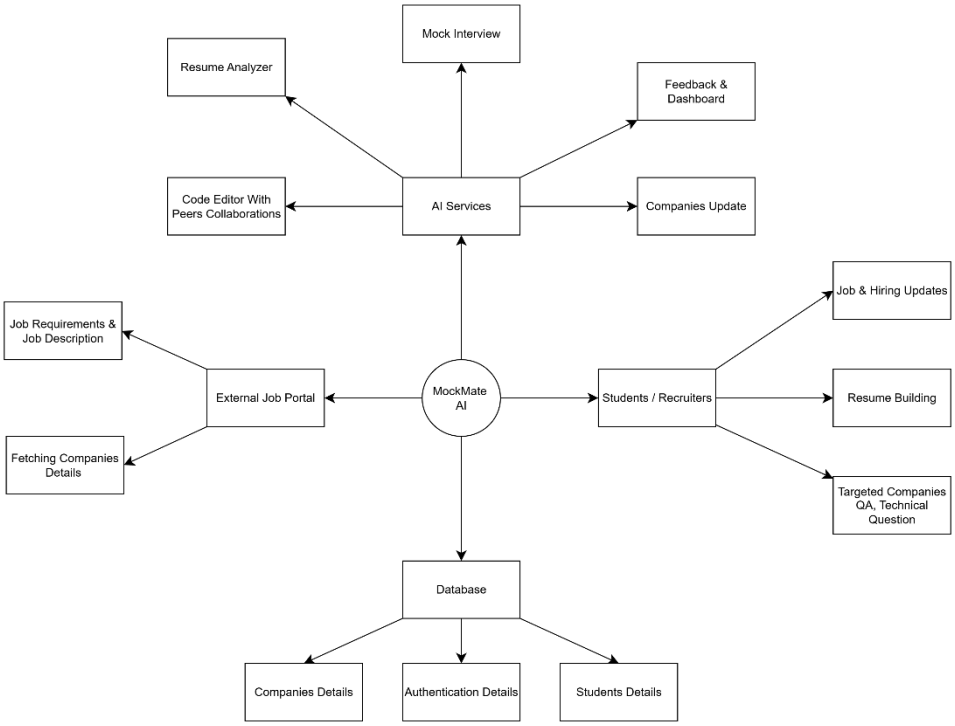
### 3.2.2 Data Flow Diagram

Figure 3.2: Data Flow Diagram (Level 0)



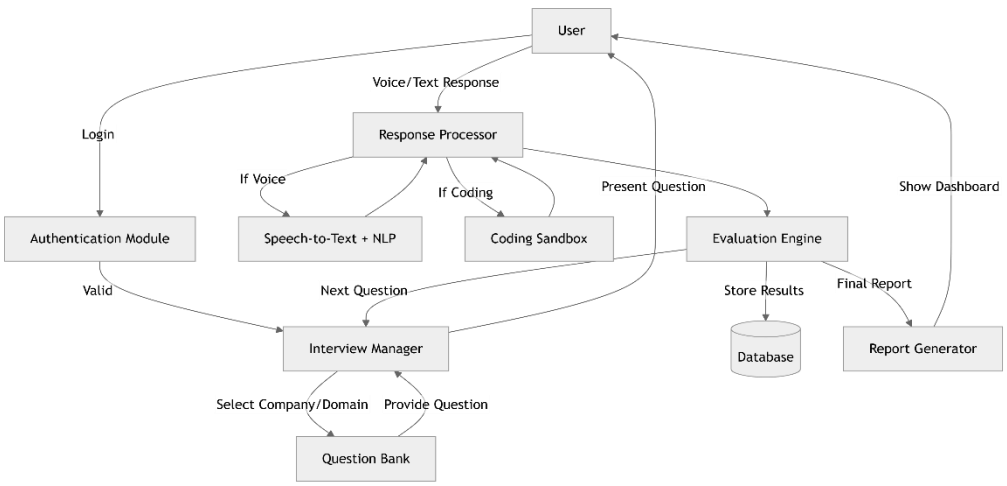
The Level 0 (Figure 3.2) depicts the system as a single process interacting with external entities such as Users, External LLM/STT providers, and Storage. It highlights the major inputs (resumes, audio) and outputs (feedback reports, scored resumes).

Figure 3.3: Data Flow Diagram (Level 1)




The Level 1 (Figure 3.3) decomposes the main system into sub-processes: Capture Module, Parsing & Extraction, Scoring & Feedback, and Storage/Reporting. The diagram clarifies how parsed text flows into scoring and how user sessions and artifacts are stored.

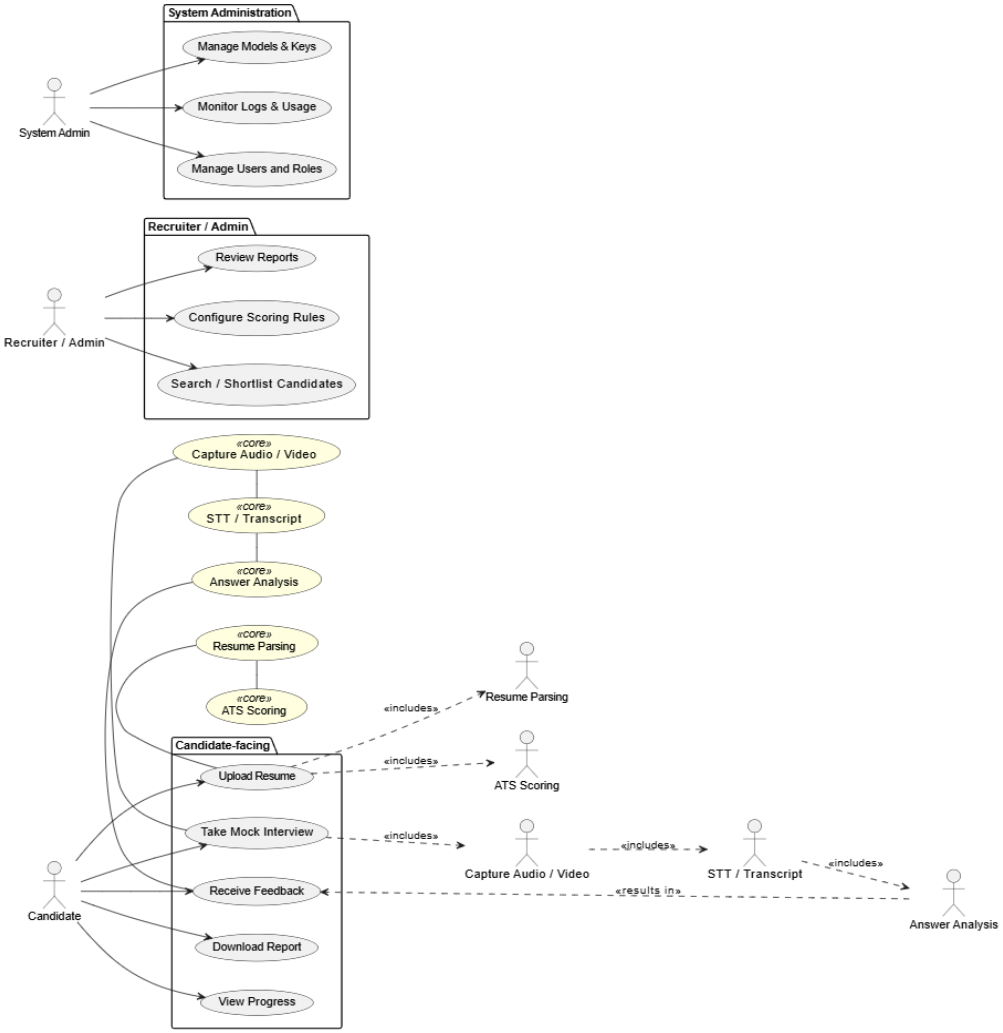
Figure 3.4: Data Flow Diagram (Level 2)



The Level 2 (Figure 3.4) gives finer detail for individual sub-processes (for example, ATS matching, keyword extraction, NER-based resume fields extraction, and transcript alignment for interviews) so implementers can map functions to services or microservices.

### 3.2.3 Use Case Diagram

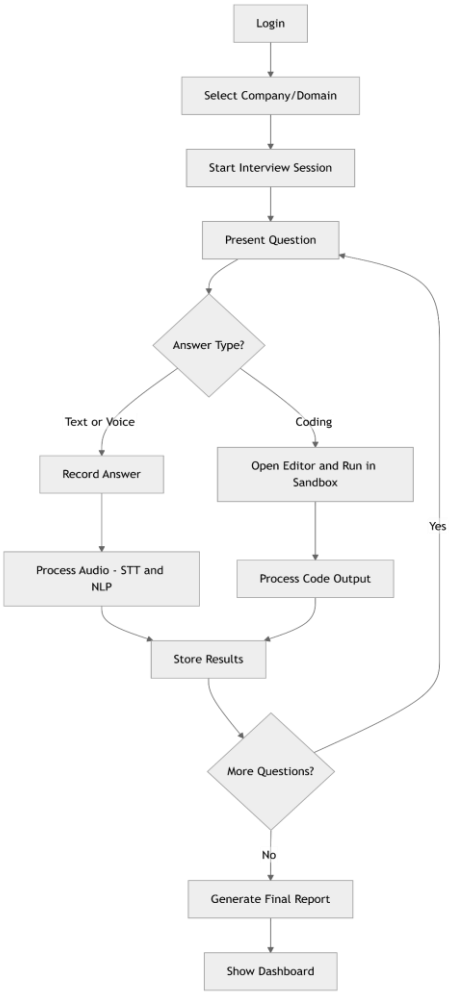
Figure 3.5: Use Case Diagram




The use-case view summarizes actors and their goals: Candidates (upload resume, take mock interviews, receive feedback), Recruiters/Admins (review reports, configure scoring rules), and System Admin (manage models, keys, logs). Use cases explicitly show the interactive features: Resume Upload & Parsing, Mock Interview Session, ATS Feedback, and Report Generation.

### 3.2.4 Activity Diagram

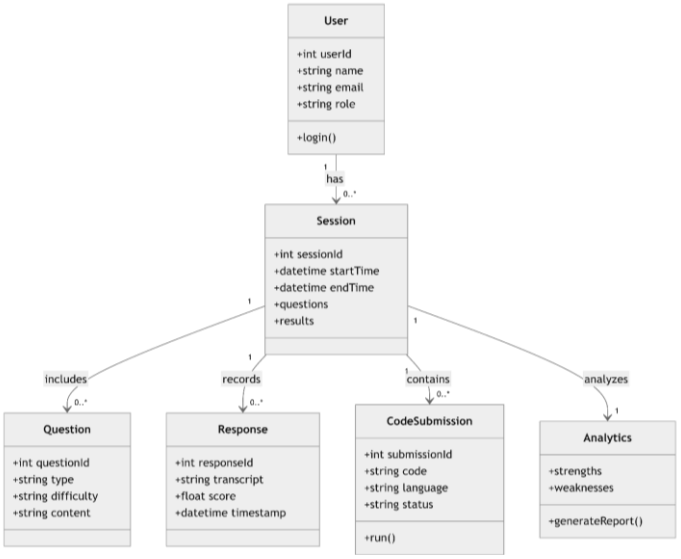
Figure 3.6: Activity Diagram




The activity diagram (Figure 3.6) visualizes end-to-end flows such as “Start Mock Interview” (capture audio → STT → answer analysis → LLM feedback → save report) and “Resume Processing” (upload → extract → score → report). It helps developers’ reason about asynchronous steps and error handling (e.g., retry STT, fallback to alternate provider).

### 3.2.5 Class Diagram

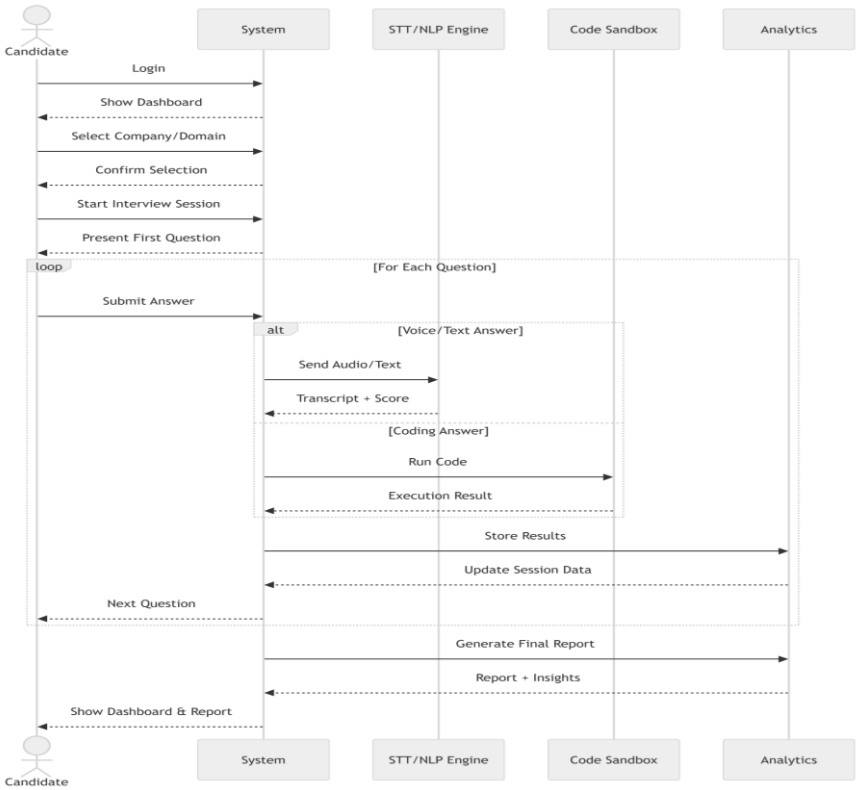
Figure 3.7: Class Diagram




The class diagram (Figure 3.7) captures main domain classes and relationships: User, Resume, Transcript, Session, ScoreReport and service classes (ParserService, ATSScorer, FeedbackService). It is intended as a blueprint for both backend models and API payload shapes.

### 3.2.6 Sequence Diagram

Figure 3.8: Sequence Diagram



The sequence diagram (Figure 3.8) shows time-ordered interactions for a typical mock interview: User client → capture audio → upload to backend → backend invokes STT and LLM → backend stores transcript and feedback → client displays results. It highlights synchronous and asynchronous calls and where the system should provide user feedback (progress spinners, notifications).

## 3.3 Hardware and Software Requirements

### 3.3.1 Hardware Requirement

**Minimum (for basic development and testing):**

- CPU: Dual-core (modern) or better (e.g., Intel i3 / AMD Ryzen 3)
- RAM: 8 GB
- Storage: 50 GB free on HDD/SSD
- Peripherals: USB microphone (or built-in mic), headphones
- Network: Stable broadband connection for API access

**Production / Server (if self-hosting inference services):**

- Multi-core server CPU, 32+ GB RAM, NVMe SSD, and GPU for model inference
- Redundant internet link and TLS-capable load balancer

### 3.3.2 Software Requirement

**Operating Systems:**

- Windows 10/11, macOS (Monterey or later), or Linux (Ubuntu 20.04+ / Debian)

**Frontend:**

- Node.js (LTS, e.g., ≥ 18.x) and npm or yarn
- React (v18+), React Router

- Puter.js (for client orchestration as specified)
- Web APIs: Web Speech API / MediaRecorder / WebRTC for browser-based audio capture (fallbacks to Vapi SDK)
- UI tooling: Tailwind / CSS framework as required, ESLint, Prettier

**Backend & Services:**

- Python (3.10+) with virtualenv / pip or Node.js backend (depending on chosen stack)
- Backend framework: FastAPI or Flask (Python) or Express (Node.js)
- NLP & ML libraries: spaCy, Transformers (Hugging Face), sentence-transformers, scikit-learn, numpy, pandas
- Resume parsing and document tooling: python-docx, pdfplumber (or tika/textract for robust PDF extraction)
- Speech processing: integrations with Vapi SDK and/or cloud Speech-to-Text (Google Speech-to-Text) for higher accuracy

**Cloud / Platform Services:**

- Firebase: Authentication, Firestore (database), Firebase Hosting, Cloud Functions (for serverless backend operations)
- Optional cloud hosting: Google Cloud / AWS / Azure for scalable inference or batch processing

**Databases & Storage:**

- Firestore / Realtime Database (Firebase) or equivalent NoSQL store for user/session data
- Cloud Storage (Firebase Storage / S3) for resume and audio blobs

**Dev Tools & Utilities:**

- Version control: Git + GitHub / GitLab / Bitbucket
- Containerization: Docker (optional, for reproducible dev/test environments)

- IDE: VS Code (recommended) or IntelliJ/WebStorm
- API testing: Postman / curl
- CI/CD: GitHub Actions / GitLab CI for automated builds, tests and deployment

**Testing & Quality:**

- Frontend tests: Jest, React Testing Library
- Backend tests: pytest (Python) or Mocha/Jest (Node)
- Static analysis: ESLint (JS/TS), flake8 / mypy (Python) as needed

**Security & Configuration:**

- HTTPS/TLS for all endpoints
- OAuth 2.0 / Firebase Authentication for user sign-in
- Secrets management: environment variables, .env (local) and secret managers (cloud) in production

## 3.4 System Development and Output

### 3.4.1 System Development

System development follows an iterative, test-driven process: first implement capture and storage flows, then resume parsing and basic scoring, and finally add LLM-based feedback and polish the UI. Early milestones should include a working end-to-end demo that accepts a resume, extracts fields, computes a simple ATS score, and generates a downloadable feedback report. Typical outputs include: parsed resume JSON, ATS score breakdown (skills, formatting), interview transcripts with annotated feedback, and a user-facing PDF/DOCX report.

### 3.4.2 Output

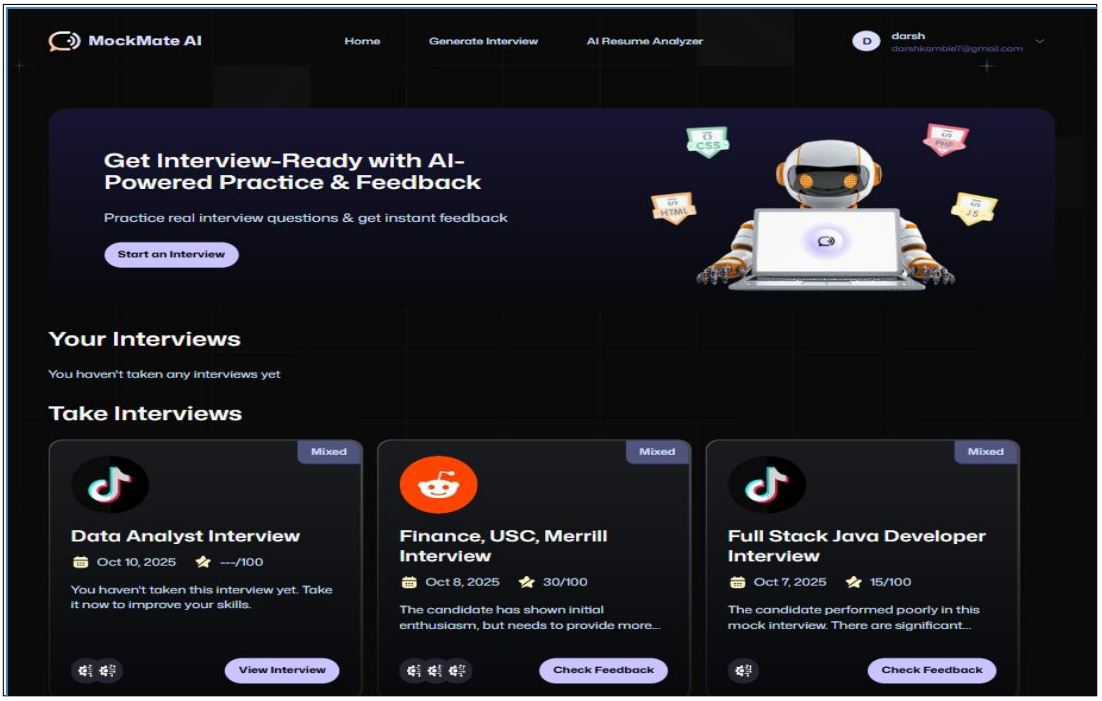



Figure 3.9: Landing page of System

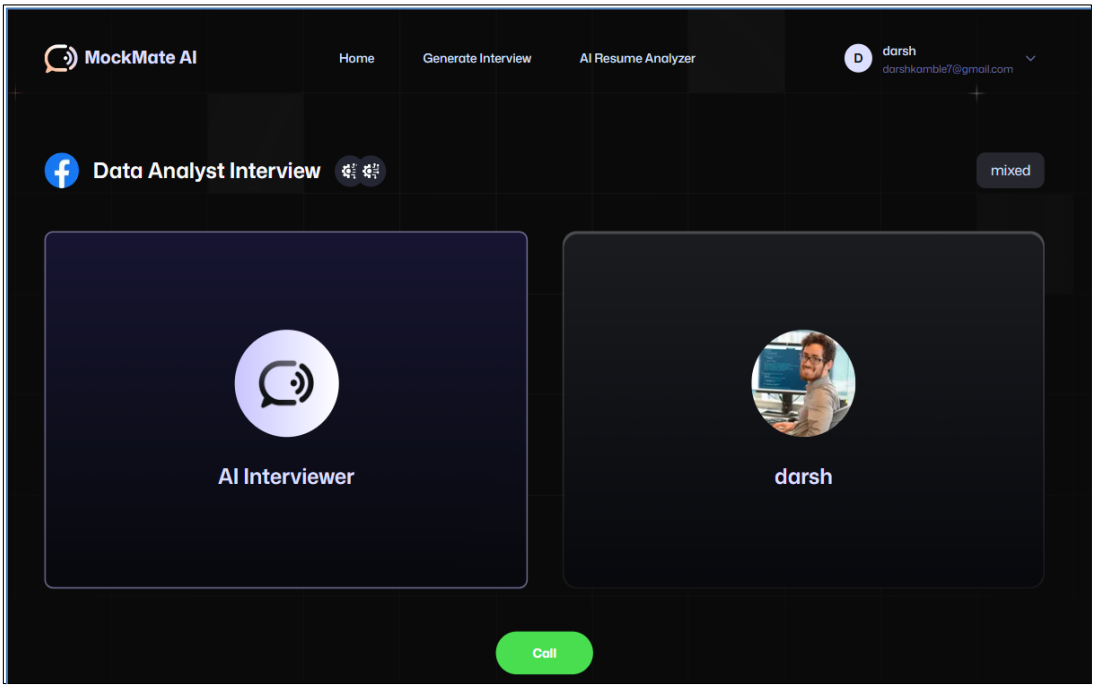



Figure 3.10: AI based Interview Working

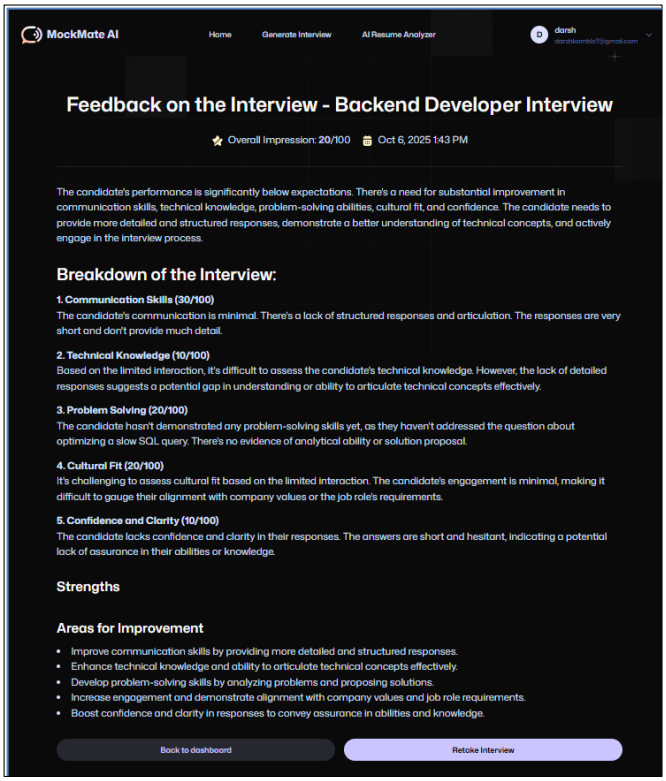



Figure 3.11: AI based Interview Feedback

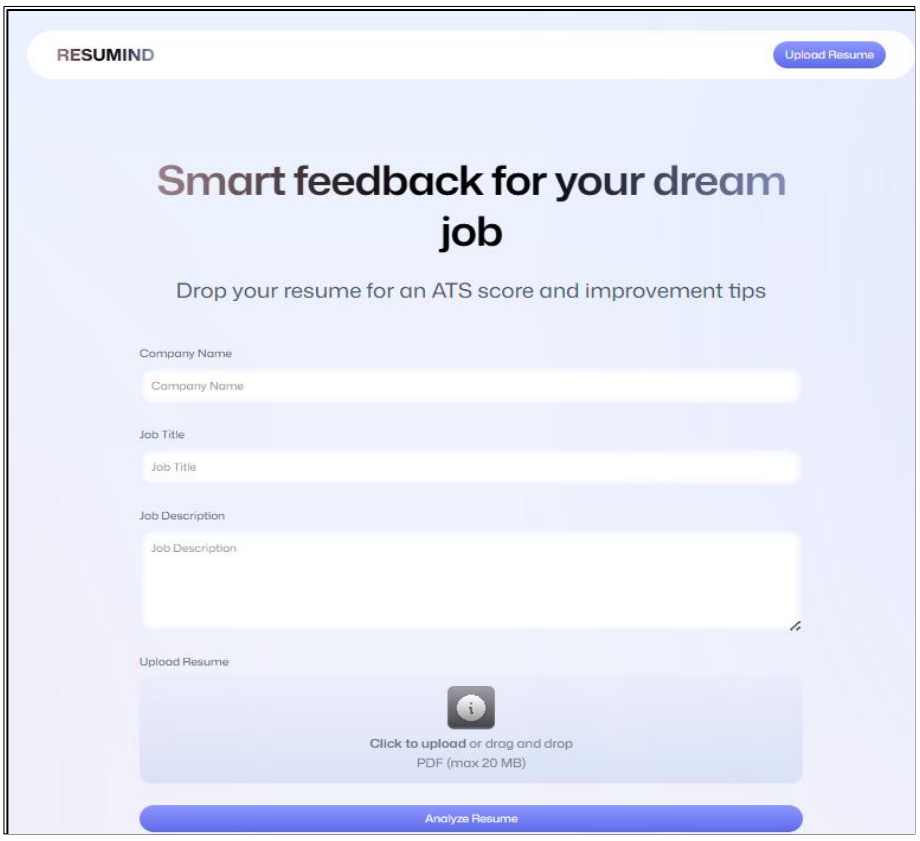



Figure 3.12: Landing page for AI Resume Analyser

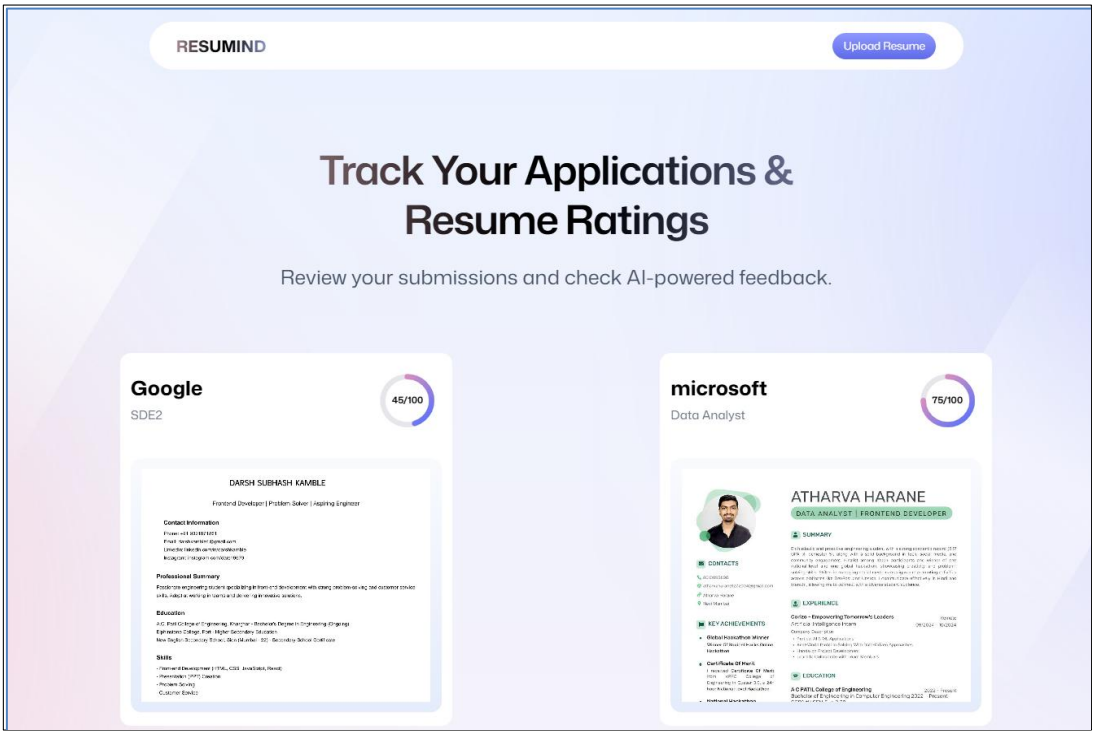



Figure 3.13: AI Resume Analyser Working

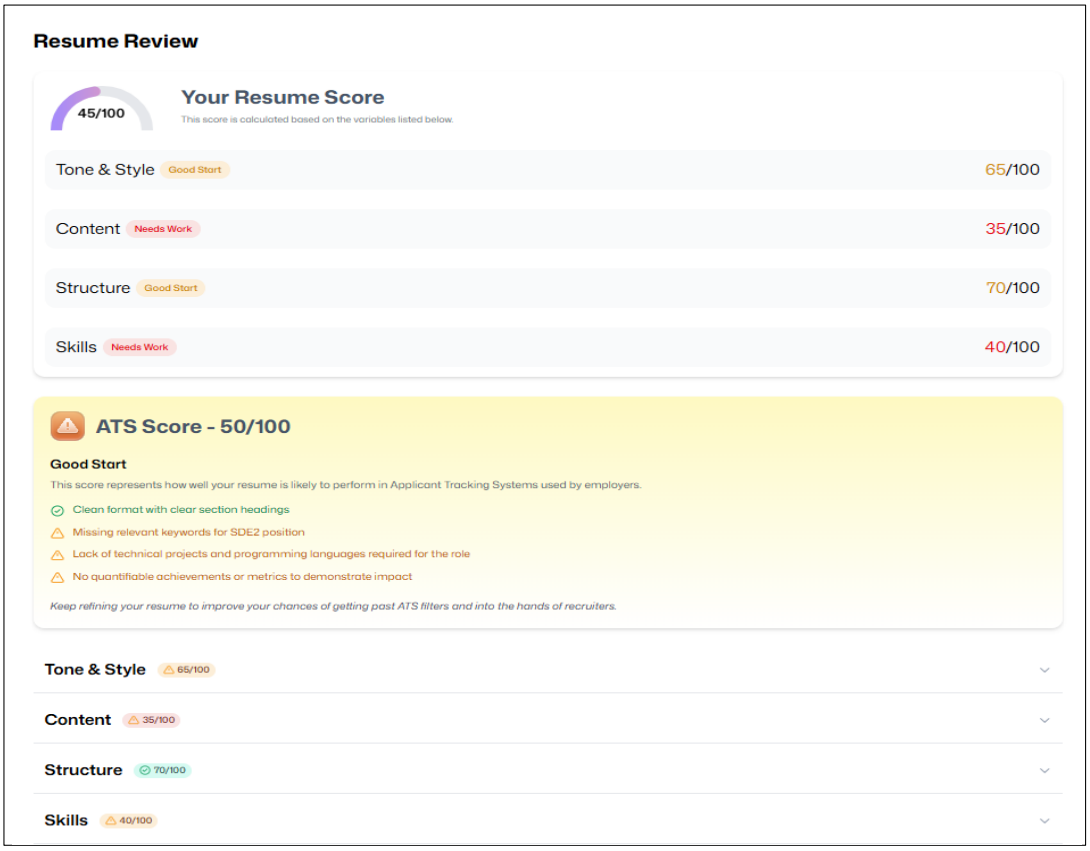



Figure 3.14: Resume Analysis & Review

## 3.5 Future Work

Possible future enhancements include on-device or on-premise LLM inference for privacy sensitive deployments, multilingual STT/TTS support beyond current languages, an extensible rules engine for customizable ATS scoring, and analytics dashboards for institutional deployment (to track student progress or hiring funnel metrics). Multimodal extensions (video-based interview evaluation) and automated anonymized benchmark datasets for continuous model evaluation are natural next steps.

## 3.6 Conclusion

The proposed MockMate AI system is intentionally modular and cloud-friendly, enabling a rapid developer experience while allowing institutions to scale and harden the deployment as needed. By prioritizing robust resume parsing, accurate STT integration, and LLM-backed qualitative feedback, the design targets high-impact functionality with practical resource requirements and clear upgrade paths for future work.

# Bibliography

[1] Lokesh Jagtap, Suyash Rane, Rahul Dhonnar, Jyoti Pawar, and Umesh Bhadane. Ai-based interview, mocker system. *IJARSCT*, 2025. Fill volume/issue/pages if available.

[2] Changwoo Kim, Woojin Lee, Jinho Choi, Daehun Yoo, and Jongyeon Yoon. Fairness-aware multimodal learning. *IEEE Transactions / Conference Proceedings*, 2023. Add volume/issue/pages as required.

[3] Fuminori Nagasawa, Shogo Okada, Katsumi Nitta, and Takuya Ishihara. Adaptive interview strategy for robots. In *Proceedings of IEEE (conference name)*, 2024. Add exact conference name/location/pages if known.

[4] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 7th edition, 2010. Edit publisher/year to match the edition you used if different.

[5] Hung-Yue Suen, Kuo en Hung, and Chien-Liang Lin. TensorFlow-based automatic personality recognition. In *Proceedings of IEEE (conference name)*, 2019. Add pages/DOI if available.

# Acknowledgments

We sincerely thank our project supervisor, Dr. A. R. Sonule, for his invaluable guidance, feedback, and unwavering support throughout the initial phase of this project. We also express our heartfelt gratitude to our families and friends for their constant encouragement and understanding. Lastly, we would like to acknowledge all individuals associated with this project for their valuable contributions and assistance, which have been instrumental in the progress we have made so far.

Date: _____