01 TECH STACK AND ARCHITECTURE

Frontend Layer

Framework and tooling: React + Vite + TypeScript + Tailwind CSS. This combination enabled rapid UI iteration with strict typing and predictable local dev behavior.

Assessment interface uses Monaco Editor to simulate a coding environment and to capture fine-grained input signals (typed vs pasted character analysis).

Recharts powers recruiter-side telemetry visualization for session integrity. The chart is not decorative; it reflects persisted telemetry events from backend records.

Routing and role-based surfaces: candidate pages (upload, assessment, status) and recruiter pages (pipeline, candidate detail, AI audit) are separated by role context.

Backend Layer

FastAPI is used for API design due to typed request/response models and clean service decomposition. SQLAlchemy models back persistence in SQLite for PoC velocity.

Service modules include: resume zone parser, assessment execution flow, telemetry ingestion, dashboard aggregation, baseline evaluator, and glass-box rationale engine.

Model orchestration: Gemini is used for structured rationale generation. A deterministic fallback path maintains continuity if model access fails.

Data Model Overview

users: account identity and role.

candidates: profile, status, parsed resume data, scores, recommendation artifacts, decision notes.

code_submissions: per-question code result and optional forensics artifacts (history, char breakdown, chat response).

proctoring_logs: event stream for integrity signals with severity and timestamps.

Trust and Stability Design

AI rationale reuse via hash-like metadata avoids unnecessary model regeneration and lowers cost/variability.

Fallback rules provide bounded behavior for high-stakes decisions when external AI service is unavailable.

Baseline checklist makes policy constraints visible and machine-checkable on recruiter surface.

Deterministic telemetry inputs (tab switch/paste) are used for demo stability; camera anomalies are supplementary due hardware variability.