TECH_STACK.md — VERDICT

AI-Powered Deposition Coaching & Trial Preparation Platform
Version: 1.0.0 — Hackathon Edition | February 21, 2026
Team: VoiceFlow Intelligence | Track: AI Automation — August.law Sponsor Track

---

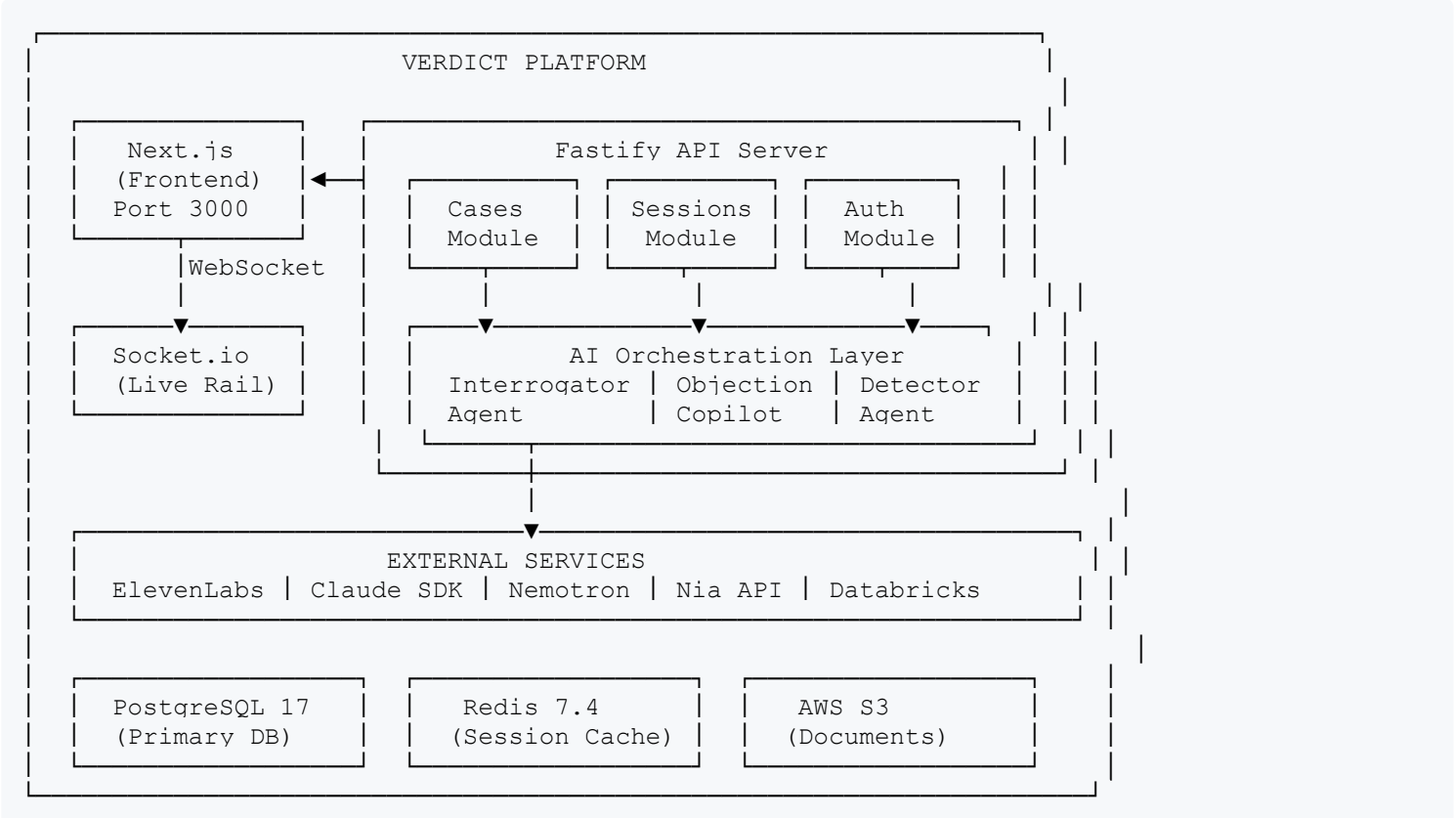
TABLE OF CONTENTS

---

1. STACK OVERVIEW

Architecture Pattern

Pattern: Modular Monolith → Microservices-ready
Rationale: A monolith ships in 48 hours. The module boundaries are drawn to extract into microservices post-hackathon without rewriting business logic.



Deployment Strategy

Environment	Frontend	Backend	Database
Hackathon (MVP)	Vercel (hobby)	Railway	Supabase (PostgreSQL) + Upstash (Redis)
v1.0 Commercial	Vercel (Pro)	Fly.io (dedicated)	AWS RDS PostgreSQL + ElastiCache Redis
v2.0 Scale	Vercel Enterprise	AWS ECS (containerized)	AWS RDS Multi-AZ + ElastiCache cluster

Architecture Justification

Decision	Rationale
Next.js full-stack over separate SPA + API	Single deployment for hackathon; App Router handles both server-side rendering and API routes. Attorney dashboard needs SSR for fast initial load.
Fastify over Express	2x throughput on JSON-heavy AI response streaming; native TypeScript support; schema validation built-in. Objection Copilot fires ≤1.5s — every ms matters.
PostgreSQL over MongoDB	Case data is highly relational (firms → cases → witnesses → sessions → flags → briefs). ACID compliance mandatory for legal data integrity.
Redis over in-memory	Session events buffered every 60 seconds (PRD §8.4). Redis pub/sub enables live alert fan-out across WebSocket connections without data loss on server restart.
Socket.io over raw WebSocket	Auto-reconnect, room-based event isolation (one room = one session), fallback to long-polling if WS blocked by enterprise firewalls.
Modular monolith over microservices	48-hour build window with team of 4. Module boundaries (Cases, Sessions, Auth, AI) drawn so each can be extracted to its own service post-hackathon without rewriting interfaces.

---

2. FRONTEND STACK

Framework

Next.js 15.1.6

- Docs: https://nextjs.org/docs
- License: MIT
- Reason: App Router provides server components for attorney dashboard (fast SSR), client components for live session real-time UI, and built-in API routes for BFF pattern. Native streaming with React Suspense matches the Interrogator Agent's streaming response pattern.
- Alternatives rejected:
  - **Vite + React SPA**: No SSR — attorney dashboard loads slowly; worse initial LCP for enterprise users.
  - **Remix**: Excellent for forms but WebSocket integration is less native; smaller ecosystem for UI component libraries.
  - **SvelteKit**: Team unfamiliar; faster to ship with React ecosystem given 48-hour window.

Language

TypeScript 5.7.3

- Docs: https://www.typescriptlang.org/docs/
- License: Apache-2.0
- Reason: Strict typing across AI API response shapes (Nemotron JSON output, ElevenLabs event payloads) catches integration bugs at compile time rather than runtime during live session. Required for Prisma client type safety.
- Config: `"strict": true`, `"noUncheckedIndexedAccess": true`, `"exactOptionalPropertyTypes": true`

Styling

Tailwind CSS 3.4.17

- Docs: https://tailwindcss.com/docs
- License: MIT
- Reason: Utility-first enables rapid hackathon iteration without context-switching to CSS files. JIT mode means zero dead CSS in production bundle. Three-panel live session layout requires precise responsive breakpoints — Tailwind's `lg:grid-cols-[220px_1fr_320px]` pattern handles this cleanly.
- Alternatives rejected:
  - **CSS Modules**: Too verbose for 48-hour build; no shared design tokens out of the box.
  - **Styled Components / Emotion**: Runtime CSS-in-JS adds overhead during live session rendering; hydration mismatch risk with Next.js App Router.

UI Components

shadcn/ui 2.1.8 (component collection, not a library)

- Docs: https://ui.shadcn.com/docs
- License: MIT
- Reason: Copy-paste components give full ownership of the code — no locked-in library versions to manage. Alert, Badge, Card, Tabs, Dialog, and Progress components map directly to VERDICT's UI inventory. Components are Radix UI primitives under the hood, ensuring WCAG 2.1 AA compliance (PRD §8.3).
- Alternatives rejected:
  - **Material-UI 6**: Heavy; Google aesthetic doesn't fit legal enterprise visual language.
  - **Chakra UI**: Requires ChakraProvider wrapping; conflicts with Next.js App Router server components.
  - **Headless UI**: Less complete — missing table, combobox, and data display components needed for brief viewer.

Radix UI 1.1.3 (via shadcn/ui)

- Docs: https://www.radix-ui.com/primitives/docs
- License: MIT
- Reason: Accessibility primitives for modals, dropdowns, tabs — WCAG 2.1 AA required by PRD.

State Management

Zustand 5.0.2

- Docs: https://docs.pmnd.rs/zustand/getting-started/introduction
- License: MIT
- Reason: Minimal boilerplate for hackathon speed. Live session store manages: alert queue, session timer, agent status, transcript buffer, WebSocket connection state. Zustand's `subscribeWithSelector` enables components to subscribe to granular slices (e.g., alert rail only re-renders when a new alert arrives, not when timer ticks).
- Alternatives rejected:
  - **Redux Toolkit**: Overkill for this data shape; slice/reducer ceremony slows hackathon velocity.

- **Jotai / Recoil**: Atom model less suited to the imperative event stream from WebSocket (each new alert is a push, not a derived value).
- **React Context**: Re-renders entire subtree on every timer tick — kills live session performance.

Form Handling

React Hook Form 7.54.2

- Docs: https://react-hook-form.com/docs
- License: MIT
- Reason: Uncontrolled inputs with minimal re-renders. Case creation, session configuration, and witness setup forms benefit from per-field validation on blur (PRD §6.5). Integration with Zod for schema-driven validation.

Zod 3.24.1 (validation schema, used on both frontend and backend)

- Docs: https://zod.dev
- License: MIT
- Reason: Single source of truth for form validation schemas shared between client-side React Hook Form and server-side Fastify schema validation. Eliminates duplicated validation logic.

HTTP Client

Axios 1.7.9

- Docs: https://axios-http.com/docs/intro
- License: MIT
- Reason: Interceptors for automatic JWT refresh on 401, automatic `x-truex-auth-signature` HMAC header injection on every request, and request cancellation for long-running ingestion status polls.
- Alternatives rejected:
  - **Native Fetch**: No interceptor support — JWT refresh requires manual wrapping on every call.
  - **TanStack Query**: Excellent for data fetching but adds overhead; live session data flows through WebSocket, not REST polling, so the cache layer provides less value for VERDICT's primary real-time flow.

TanStack Query (React Query) 5.66.0 (for server state caching)

- Docs: https://tanstack.com/query/v5/docs
- License: MIT
- Reason: Attorney dashboard, case list, and witness profile data benefit from stale-while-revalidate caching. Prevents redundant fetches when navigating between tabs in Case Detail view.

Routing

Next.js App Router (built-in, v15.1.6)

- Docs: https://nextjs.org/docs/app
- Reason: File-system routing matches the navigation map exactly. Nested layouts handle the persistent case header (with deposition countdown) across all `/cases/:caseId/*` routes. Middleware handles JWT validation before rendering any protected route.

Real-Time Client

Socket.io-client 4.8.1

- Docs: https://socket.io/docs/v4/client-api/
- License: MIT

- Reason: Auto-reconnect on network drop (PRD §6.3 — offline handling). Room isolation: each session gets a unique room; alerts are fan-out only to the attorney viewing that session. Falls back to HTTP long-polling when enterprise firewalls block WebSockets (common in law firm networks).

PDF Client Rendering

@react-pdf/renderer 4.1.5

- Docs: https://react-pdf.org
- License: MIT
- Reason: Coaching brief PDF generation on the client side for the preview before downloading. Server-side rendering for the downloadable version. Matches the brief's structured sections (score card, inconsistency table, radar chart snapshot).

Facial Landmark Detection (Client-Side Only)

@mediapipe/tasks-vision 0.10.20

- Docs: https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker
- License: Apache-2.0
- Reason: P1.4 Behavioral Sentinel requirement. Runs Face Mesh entirely in-browser via WebAssembly — no video frames transmitted to server. 430-landmark model at ≥15fps on modern hardware. FACS Action Unit computation happens client-side before only the numeric AU vector array is sent to the backend.
- Alternatives rejected:
  - **face-api.js**: Older architecture; lower accuracy on subtle micro-expressions; no official FACS output.
  - **TensorFlow.js BlazeFace**: Detection only, no landmark mesh; insufficient for AU extraction.

Audio Visualization

wavesurfer.js 7.8.11

- Docs: https://wavesurfer.xyz/docs/
- License: BSD-3-Clause
- Reason: ElevenLabs waveform visualization in the live session center panel. Web Audio API integration for amplitude-synced avatar glow. Used for coaching brief audio clip playback.

Animation

Framer Motion 12.4.7

- Docs: https://motion.dev/docs
- License: MIT
- Reason: Alert rail slide-in animations, score card number count-up, radar chart axis animations in coaching brief. Layout animations for the three-panel live session layout on resize.
- Alternatives rejected:
  - **CSS animations only**: Insufficient for physics-based spring animations (alert card slide-in).
  - **GSAP**: License cost for commercial use; overkill for the specific animation set.

Charts (Weakness Map Radar)

Recharts 2.15.0

- Docs: https://recharts.org/en-US/api
- License: MIT
- Reason: Native React components for the Weakness Map radar chart (P1.2). No canvas API — pure SVG means radar chart is accessible and print-friendly in the brief PDF.

---

3. BACKEND STACK

Runtime

Node.js 22.13.0 LTS

- Docs: https://nodejs.org/docs/latest-v22.x/api/
- License: MIT
- Reason: LTS release — security patches until April 2027. Native `fetch` API eliminates one dependency. `AsyncLocalStorage` for request-scoped context (tenant isolation per API call). Worker threads for CPU-bound PDF generation without blocking the event loop.

Framework

Fastify 5.2.1

- Docs: https://fastify.dev/docs/latest/
- License: MIT
- Reason: Schema-first JSON validation via JSON Schema / Zod. Native TypeScript support. 2x throughput vs Express on identical hardware — critical when Objection Copilot must fire ≤1.5 seconds after question delivery. Plugin architecture maps cleanly to VERDICT's module boundaries (Cases, Sessions, Auth, AI, Admin).
- Alternatives rejected:
  - **Express 5**: No built-in schema validation; request lifecycle is less structured; slower JSON serialization.
  - **NestJS**: Excellent architecture but decorator magic + DI container adds boot time; VERDICT's module count doesn't justify it at hackathon scale.
  - **Hono**: Too minimal — lacks plugin ecosystem needed for file upload, SAML, and WebSocket in one framework.

WebSocket Server

Socket.io 4.8.1 (server)

- Docs: https://socket.io/docs/v4/server-api/
- License: MIT
- Reason: Fastify plugin `@fastify/socket.io 3.0.0` integrates cleanly. Redis adapter enables horizontal scaling (multiple Fastify instances share the same session room). Named events map directly to VERDICT's alert types: `objection_alert`, `inconsistency_alert`, `composure_alert`, `interrogator_question`, `session_state_change`.

Database

PostgreSQL 17.2

- Docs: https://www.postgresql.org/docs/17/
- License: PostgreSQL License (permissive)
- Reason: Full ACID compliance for legal data. JSONB columns for Nemotron's flexible `{ contradiction_confidence, prior_quote, page, line }` output shapes. Row-level security policies enforce firm-level data isolation at the database layer (belt-and-suspenders with application-layer isolation). Full-text search via `pg_trgm` for the Prior Sworn Statements searchable index.
- Alternatives rejected:
  - **MySQL 8**: Weaker JSONB support; no row-level security.
  - **MongoDB**: Document model doesn't fit the firm → case → witness → session → flag relational chain; no ACID transactions.
  - **SQLite**: Not suitable for concurrent sessions (≥20 simultaneous, PRD §8.1).

ORM

Prisma 6.3.1

- Docs: https://www.prisma.io/docs
- License: Apache-2.0
- Reason: Schema-first — `schema.prisma` is the single source of truth for database types, shared with TypeScript via generated client. Prisma Migrate handles the full migration history. Connection pooling via `prisma.$pool` with PgBouncer support for production.
- Alternatives rejected:
  - **Drizzle ORM**: Less mature migration tooling; team less familiar.
  - **TypeORM**: Decorator-heavy; struggles with Fastify's non-class-based architecture.
  - **Raw pg / postgres.js**: Too much manual SQL for a 48-hour window; no type safety on query results.

Caching & Session Store

Redis 7.4.1 (via Upstash for hackathon, ElastiCache for production)

- Docs: https://redis.io/docs/latest/
- License: RSALv2 / SSPLv1 (server); MIT (client libraries)

ioredis 5.4.1 (Node.js client)

- Docs: https://github.com/redis/ioredis
- License: MIT
- Reason: Session event buffering (60-second auto-save, PRD §8.4). Socket.io Redis adapter for WebSocket horizontal scaling. Rate limiting counters (Inconsistency Detector API call rate). Witness token storage with TTL (72-hour automatic expiry). Brief share token with 7-day TTL.
- Alternatives rejected:
  - **node-redis**: Less mature TypeScript types; ioredis has better Cluster support.
  - **Memcached**: No pub/sub (required by Socket.io adapter); no TTL-based token management.

Authentication

JWT (JSON Web Tokens)

jsonwebtoken 9.0.2

- Docs: https://github.com/auth0/node-jsonwebtoken
- License: MIT
- Reason: Stateless auth for attorney sessions. Short-lived access tokens (8 hours for enterprise, per PRD §6.6) + long-lived refresh tokens. Claims include: `firmId`, `userId`, `role`, `email`.

@node-saml/passport-saml 5.0.1 (SAML 2.0 / SSO)

- Docs: https://github.com/node-saml/passport-saml
- License: MIT
- Reason: Enterprise SSO requirement (Okta, Azure AD, iManage SAML IdPs). PRD §1.3 specifies SAML 2.0 as the primary enterprise auth method.

passport 0.7.0

- Docs: https://www.passportjs.org/docs/
- License: MIT
- Reason: Strategy pattern handles both SAML SSO and local email/password in the same auth middleware pipeline.

bcrypt 5.1.1 (password hashing)

- Docs: https://github.com/kelektiv/node.bcrypt.js
- License: MIT
- Reason: Adaptive hashing for email/password accounts. Cost factor: 12 (see §11 Security).

@fastify/jwt 9.0.1 (Fastify plugin)

- Docs: https://github.com/fastify/fastify-jwt
- License: MIT

nanoid 5.0.9 (token generation for witness links and brief share tokens)

- Docs: https://github.com/ai/nanoid
- License: MIT
- Reason: Cryptographically secure URL-safe token IDs. 24-character tokens provide 144 bits of entropy — sufficient for the 7-day share token and 72-hour witness token.

File Storage

AWS S3 (via AWS SDK v3)

@aws-sdk/client-s3 3.726.1

- Docs: https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/client/s3/
- License: Apache-2.0
- Reason: Up to 200MB document uploads (PRD §P0.4). S3 multipart upload for files >5MB. Pre-signed URLs for direct browser-to-S3 upload (bypasses Fastify for large files). Separate buckets per firm (case data isolation). Lifecycle policy: auto-delete after 90 days (PRD §8.2).
- Alternatives rejected:
  - **Cloudinary**: Image/video CDN — wrong tool for PDF/DOCX.
  - **Vercel Blob**: 500MB max per file vs S3's 5TB; less control over access policies.
  - **Local disk**: Not viable across multiple server instances; not durable.

@aws-sdk/s3-request-presigner 3.726.1

- License: Apache-2.0
- Reason: Generates pre-signed URLs for direct browser uploads without routing through Fastify.

@aws-sdk/lib-storage 3.726.1

- License: Apache-2.0
- Reason: Managed multipart upload for large case documents.

Document Processing

pdf-parse 1.1.1 (PDF text extraction for validation before sending to Nia)

- Docs: https://gitlab.com/autokent/pdf-parse
- License: MIT
- Reason: Pre-screens uploaded PDFs to confirm text is extractable before kicking off the expensive Nia ingestion pipeline. Catches "image-only" PDFs early (PRD error state: "No text content found").

mammoth 1.8.0 (DOCX text extraction)

- Docs: https://github.com/mwilliamson/mammoth.js
- License: BSD-2-Clause
- Reason: Converts uploaded DOCX files to clean text for Nia ingestion without requiring LibreOffice.

puppeteer 22.15.0 (server-side PDF generation for coaching briefs)

- Docs: https://pptr.dev
- License: Apache-2.0
- Reason: Renders the coaching brief React component server-side to PDF with full CSS fidelity. Headless Chrome in a Docker container on Railway/Fly.io. The brief PDF must match the browser view exactly (PRD §P0.5 requirement: brief exportable as PDF).
- Alternatives rejected:
  - **@react-pdf/renderer** (server-side only): Custom PDF DSL doesn't render Recharts radar chart; would require a separate SVG export path.
  - **jsPDF**: Client-side only; cannot run in server environment without a browser.

Email

Resend 4.1.2

- Docs: https://resend.com/docs
- License: MIT
- Reason: Developer-first email API with React Email template support. Transactional emails: brief ready, witness invitation, plateau alert, ingestion complete. 100 emails/day free tier sufficient for hackathon. Excellent deliverability for enterprise law firm email servers.
- Alternatives rejected:
  - **SendGrid**: Heavier SDK; pricing tier jumps at volume.
  - **Nodemailer + SMTP**: Requires managing an SMTP relay; worse deliverability.

react-email 3.0.7 (email templates)

- Docs: https://react.email/docs
- License: MIT
- Reason: Build email templates as React components with TypeScript type safety.

API Rate Limiting & Validation

@fastify/rate-limit 10.2.1

- Docs: https://github.com/fastify/fastify-rate-limit
- License: MIT
- Reason: Route-level rate limits for AI endpoints (Inconsistency Detector, Objection Copilot). Redis-backed so limits are shared across all server instances.

@fastify/multipart 9.0.3 (file upload handling)

- Docs: https://github.com/fastify/fastify-multipart
- License: MIT
- Reason: Streams large document uploads to S3 without buffering the entire file in memory.

@fastify/cors 10.0.2

- Docs: https://github.com/fastify/fastify-cors
- License: MIT

---

4. AI & EXTERNAL SERVICE INTEGRATIONS

Anthropic Claude SDK (Primary AI Orchestration)

@anthropic-ai/sdk 0.36.3

- Docs: https://docs.anthropic.com/en/api/getting-started

- License: MIT
- Model used: `claude-sonnet-4-20250514`
- Role in VERDICT:
  - **Interrogator Agent**: Question strategy, adaptive follow-up generation, topic arc management
  - **Objection Copilot**: FRE rule classification (Leading, Hearsay, Compound, Assumes Facts, Speculation)
  - **Inconsistency Detector**: Semantic comparison of witness answers against Nia-returned prior statements
  - **Review Orchestrator**: Coaching brief synthesis, narrative generation, top-3 recommendations
- Integration pattern: Streaming responses via `stream: true` for the Interrogator (reduces time-to-first-audio). Tool use for structured Nemotron handoff.

ElevenLabs (Voice AI)

elevenlabs 1.14.0

- Docs: https://elevenlabs.io/docs/developer-guides/quickstart
- License: MIT
- Roles:
  - **TTS**: Interrogator Agent voice ("opposing counsel" profile), Review Orchestrator Coach voice
  - **STT**: Witness answer transcription during live session
  - **Conversational AI**: Real-time session management with Voice Activity Detection
- Voice profiles:
  - Interrogator: `"Adam"` — authoritative, measured cadence, legal register
  - Coach: `"Rachel"` — warm, professional, encouraging
- Latency target: <2s from generation to audio start (PRD §8.1)
- WebSocket streaming: ElevenLabs Conversational AI WebSocket for bidirectional real-time session

NVIDIA Nemotron API (Reasoning & Scoring)

HTTP Client: Axios 1.7.9 (no official Node.js SDK — REST API only)

- Docs: https://build.nvidia.com/explore/discover
- API Base: `https://integrate.api.nvidia.com/v1`
- Model: `nvidia/llama-3.1-nemotron-ultra-253b-v1`
- Role: Contradiction confidence scoring, argument strength scoring (P1.3), Behavioral Sentinel multimodal reasoning (P1.4 — fed FACS AU vectors + text transcript simultaneously)
- Fallback: If Nemotron response >5s or errors, falls back to Claude-only scoring with threshold raised from 0.75 to 0.85 (PRD Decision Point 5.4)
- Budget: $40+ API credits from hackathon organizers

Nozomio Nia API (Document Indexing & RAG)

HTTP Client: Axios 1.7.9 (Nia MCP server integration)

- Docs: Provided via hackathon Nia API documentation
- Role:
  - FRE rules corpus indexing (Objection Copilot knowledge base)
  - Case document indexing (prior sworn statement retrieval for Inconsistency Detector)
  - Semantic search returning top-N prior statement matches with page/line references
- Integration: All 4 agents query Nia for context before generating outputs — Nia is the shared knowledge layer

Databricks (Analytics & Delta Lake)

@databricks/sql 1.10.0

- Docs: https://docs.databricks.com/aws/en/dev-tools/node-sql-driver/
- License: Apache-2.0
- Role:
  - Delta Lake: Session event storage (objection events, inconsistency flags, emotion vectors from Behavioral Sentinel)
  - Delta Live Tables: Real-time Witness Composure Timeline (P1.4 — emotion vectors + audio latency + semantic flags on one time axis)
  - Weakness Map (P1.2): Databricks-powered radar chart queries via SQL
  - MLflow (P2.3): Case outcome analytics (post-hackathon)
- Connection: Databricks workshop credits; SQL warehouse endpoint

MediaPipe (Client-Side Only — Behavioral Sentinel)

@mediapipe/tasks-vision 0.10.20

- Docs: https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker
- License: Apache-2.0
- Execution: WebAssembly in-browser — zero server involvement for raw landmark data
- Model: `face_landmarker.task` (430 landmark points, ≥15fps)
- FACS Action Units computed client-side: AU4 (brow furrow), AU6 (cheek raise), AU12 (lip corner), AU20 (lip stretch), AU45 (blink)
- Only the computed AU numeric vectors are transmitted to backend (never raw video or frames)
- Consent gate: Feature disabled entirely if browser camera permission is denied (PRD Decision Point 5.6)

---

5. DATABASE SCHEMA & DATA LAYER

Core Schema (Prisma)

// prisma/schema.prisma generator client { provider = "prisma-client-js" } datasource db { provider = "postgresql" url = env("DATABASE_URL") } model Firm { id String @id @default(cuid()) name String ssoMetadataUrl String? retentionDays Int @default(90) sentinelEnabled Boolean @default(false) seats Int createdAt DateTime @default(now()) updatedAt DateTime @updatedAt users User[] cases Case[] } model User { id String @id @default(cuid()) firmId String email String @unique name String role Role passwordHash String? // null if SSO-only createdAt DateTime @default(now()) updatedAt DateTime @updatedAt firm Firm @relation(fields: [firmId], references: [id]) ownedCases Case[] @relation("CaseOwner") } enum Role { PARTNER ASSOCIATE PARALEGAL ADMIN } model Case { id String @id @default(cuid()) firmId String ownerId String name String caseType CaseType opposingFirm String? depositionDate DateTime? createdAt DateTime @default(now()) updatedAt DateTime @updatedAt firm Firm @relation(fields: [firmId], references: [id]) owner User @relation("CaseOwner", fields: [ownerId], references: [id]) documents Document[] witnesses Witness[] sessions Session[] } enum CaseType { MEDICAL_MALPRACTICE EMPLOYMENT_DISCRIMINATION COMMERCIAL_DISPUTE CONTRACT_BREACH OTHER } model Document { id String @id @default(cuid()) caseId String firmId String filename String s3Key String fileSize Int // bytes mimeType String docType DocumentType ingestionStatus IngestionStatus @default(PENDING) niaIndexId String? // Nia's internal document ID post-indexing extractedFacts Json? // structured fact extraction result pageCount Int? ingestionError String? createdAt DateTime @default(now()) updatedAt DateTime @updatedAt case Case @relation(fields: [caseId], references: [id]) } enum DocumentType { PRIOR_DEPOSITION MEDICAL_RECORDS FINANCIAL_RECORDS CORRESPONDENCE EXHIBIT OTHER } enum IngestionStatus { PENDING UPLOADING INDEXING READY FAILED } model Witness { id String @id @default(cuid()) caseId String firmId String name String email String role WitnessRole notes String? linkedDocIds String[] // document IDs associated with this witness createdAt DateTime @default(now()) updatedAt DateTime @updatedAt case Case @relation(fields: [caseId], references: [id]) sessions Session[] } enum WitnessRole { DEFENDANT PLAINTIFF EXPERT CORPORATE_REPRESENTATIVE OTHER } model Session { id String @id @default(cuid()) caseId String witnessId String firmId String sessionNumber Int // 1, 2, 3... per witness status SessionStatus @default(CONFIGURED) durationMinutes Int focusAreas FocusArea[] aggressionLevel AggressionLevel @default(STANDARD) objectionCopilot Boolean @default(true) behavioralSentinel Boolean @default(false) witnessToken String? @unique // 72-hour access token sessionScore Int? // 0-100 consistencyRate Float?

transcriptRaw String? startedAt DateTime? endedAt DateTime? createdAt DateTime @default(now()) updatedAt DateTime @updatedAt case Case @relation(fields: [caseId], references: [id]) witness Witness @relation(fields: [witnessId], references: [id]) alerts Alert[] brief Brief? } enum SessionStatus { CONFIGURED LOBBY ACTIVE PAUSED COMPLETE FAILED } enum FocusArea { TIMELINE_CHRONOLOGY FINANCIAL_DETAILS COMMUNICATIONS RELATIONSHIPS ACTIONS_TAKEN PRIOR_STATEMENTS } enum AggressionLevel { STANDARD ELEVATED HIGH_STAKES } model Alert { id String @id @default(cuid()) sessionId String firmId String alertType AlertType questionId String // maps to Q-number in transcript questionTimestamp DateTime questionText String answerText String? frcRule String? // e.g., "FRE 611(c)" priorQuote String? priorDocumentPage Int? priorDocumentLine Int? contradictionConf Float? // Nemotron confidence 0-1 behavioralAuVectors Json? // FACS AU data if Sentinel active impeachmentRisk ImpRisk? attorneyDecision AttyDecision? attorneyNote String? confirmedAt DateTime? createdAt DateTime @default(now()) session Session @relation(fields: [sessionId], references: [id]) } enum AlertType { OBJECTION_COPILOT INCONSISTENCY_DETECTED INCONSISTENCY_SECONDARY // confidence 0.50-0.74 COMPOSURE_ALERT // Behavioral Sentinel } enum ImpRisk { STANDARD HIGH } enum AttyDecision { CONFIRMED REJECTED ANNOTATED } model Brief { id String @id @default(cuid()) sessionId String @unique firmId String sessionScore Int consistencyRate Float improvementDelta Int? // vs session 1 for this witness confirmedFlags Int objectionCount Int composureAlerts Int topRecommendations String[] // array of 3 recommendation strings narrativeText String // full Claude-generated narrative pdfS3Key String? // S3 key for rendered PDF shareToken String? @unique // 7-day expiring token shareTokenExpiresAt DateTime? weaknessMapScores Json? // { timeline: 78, financial: 34, ... } createdAt DateTime @default(now()) updatedAt DateTime @updatedAt session Session @relation(fields: [sessionId], references: [id]) }

Migration Strategy

- **Tool:** Prisma Migrate (`prisma migrate dev` for development, `prisma migrate deploy` for production CI)
- **Naming convention:** `YYYYMMDD_HHMMSS_description` (e.g., `20260221_120000_add_behavioral_sentinel_columns`)
- **Production deploys:** Migration runs in CI before server restart; Prisma's `migrate deploy` is safe for additive changes (add column, add table). Destructive changes (drop column) require a two-phase deploy: first shadow the column, then drop after server restart.
- **Branching:** Each feature branch creates its own migration file; migrations are squashed before merging to `main` .

Seeding Approach

```
// prisma/seed.ts
// Development seeds: 1 firm, 3 users (partner + associate + admin),
// 2 cases (Medical Malpractice, Employment Discrimination),
// 5 documents, 3 witnesses, 3 complete sessions with alerts + briefs
// Run: npx prisma db seed
```

Backup Policy

Environment	Strategy	Frequency	Retention
Hackathon (Supabase)	Supabase managed backups	Daily	7 days
v1.0 (AWS RDS)	Automated RDS snapshots	Daily	35 days
v1.0 (AWS RDS)	Point-in-time recovery	Continuous	7 days
v2.0	RDS Multi-AZ + cross-region replica	Continuous	35 days

Connection Pooling

Hackathon: Prisma built-in connection pool (`connection_limit=10` in DATABASE_URL)
Production: PgBouncer in transaction mode via `DATABASE_URL` pointing to pooler endpoint
Max connections: PostgreSQL configured at 100; PgBouncer pools to 20 per Fastify instance

---

6. DEVOPS & INFRASTRUCTURE

Version Control & Branching

Git + **GitHub** (private repository)

```
 Branching Strategy (Trunk-Based with short-lived feature branches):

   main              ← production-ready; protected; requires PR + 1 approval
   ├── develop       ← integration branch; auto-deploys to staging
   ├── feat/case-ingestion-pipeline
   ├── feat/objection-copilot-agent
   ├── feat/behavioral-sentinel
   └── fix/session-reconnect-websocket

 Commit convention: Conventional Commits
   feat(sessions): add witness token expiry validation
   fix(alerts): prevent duplicate inconsistency flags on retry
   chore(deps): bump @anthropic-ai/sdk from 0.35.0 to 0.36.3
   docs(api): add Fastify route JSDoc for alert endpoints

 Hackathon exception: Direct commits to `main` permitted during 48-hour window.
```

CI/CD

GitHub Actions

.github/workflows/ci.yml — triggered on PR to main and develop name: VERDICT CI on: push: branches: [main, develop] pull_request: branches: [main] jobs: test: runs-on: ubuntu-latest services: postgres: image: postgres:17.2 env: POSTGRES_PASSWORD: test POSTGRES_DB: verdict_test redis: image: redis:7.4.1 steps: - uses: actions/checkout@v4 - uses: actions/setup-node@v4 with: node-version: '22.13.0' cache: 'npm' - run: npm ci - run: npx prisma migrate deploy env: DATABASE_URL: postgresql://postgres:test@localhost:5432/verdict_test - run: npm run lint - run: npm run typecheck - run: npm run test:unit - run: npm run test:integration e2e: runs-on: ubuntu-latest needs: test steps: - uses: actions/checkout@v4 - uses: actions/setup-node@v4 with: node-version: '22.13.0' - run: npm ci - run: npx playwright install --with-deps chromium - run: npm run build - run: npm run test:e2e deploy-staging: needs: [test, e2e] if: github.ref == 'refs/heads/develop' runs-on: ubuntu-latest steps: - uses: actions/checkout@v4 - run: npm run deploy:staging deploy-production: needs: [test, e2e] if: github.ref == 'refs/heads/main' runs-on: ubuntu-latest steps: - uses: actions/checkout@v4 - run: npm run deploy:production

Hosting

Frontend — Vercel

Hackathon:

- Vercel Hobby (free)
- Auto-deploys on `main` push
- Preview deployments on every PR
- Environment variables set in Vercel dashboard

Production v1.0:

- Vercel Pro ($20/month)

- Custom domain: `verdict.law`
- Edge middleware for JWT validation (runs at edge, not in Lambda)
- ISR (Incremental Static Regeneration) for marketing landing page

Backend — Railway (Hackathon) → Fly.io (Production)

Hackathon (Railway):

- Starter plan ($5/month)
- Single Fastify container
- Auto-deploy from `main` via Railway GitHub integration
- Internal URL exposed to Vercel via `BACKEND_URL` env var

Production v1.0 (Fly.io):

- 2× `performance-2x` machines (4 CPU, 8GB RAM — needed for Puppeteer PDF generation)
- Deployed in `iad` (Virginia) and `lhr` (London) regions — proximity to US/UK law firms
- Socket.io horizontal scaling via Redis adapter (all machines share Redis pub/sub)

Deployment command:

fly deploy --config fly.toml --dockerfile Dockerfile.production

Database — Supabase (Hackathon) → AWS RDS (Production)

Hackathon (Supabase):

- Free tier (500MB storage, 2 CPU, 1GB RAM)
- Connection string via `DATABASE_URL`
- Daily backups included

Production v1.0 (AWS RDS):

- `db.t4g.medium` (2 vCPU, 4GB RAM)
- PostgreSQL 17.2
- Multi-AZ disabled (cost optimization) → enable at $500K ARR
- Storage: 100GB gp3, auto-scaling to 500GB

Redis — Upstash (Hackathon) → AWS ElastiCache (Production):

- Hackathon: Upstash free tier (10,000 commands/day)
- Production: `cache.t4g.small` ElastiCache Redis 7.4

Monitoring

Error Tracking: Sentry 8.51.0

- `@sentry/nextjs 8.51.0` (frontend)
- `@sentry/node 8.51.0` (backend)
- Docs: https://docs.sentry.io
- Alerts: PagerDuty on-call for error rate >2% over 5 minutes

Application Performance: Sentry Performance (included in Sentry plan)

- Traces for each Objection Copilot call, Inconsistency Detector call, ElevenLabs STT/TTS
- P95 latency alerts: >1.5s for Objection Copilot, >4s for Inconsistency Detector

Uptime Monitoring: Better Uptime (free plan)

- 1-minute check interval on `/api/v1/health`
- SMS + email alert on downtime

Logging: Pino 9.6.0 (Fastify built-in logger)

- Docs: https://getpino.io
- License: MIT
- Structured JSON logs → Railway log drain → Logtail (hackathon) → AWS CloudWatch (production)
- Log levels: `error` for production, `info` for staging, `debug` for development

Analytics: PostHog 1.6.0

- Docs: https://posthog.com/docs
- License: MIT (self-hosted) / Commercial (cloud)
- Events: `session_started`, `session_ended`, `brief_generated`, `alert_fired`, `brief_shared`

Testing

Unit & Integration: Vitest 2.1.8

- Docs: https://vitest.dev/guide/
- License: MIT
- Reason: 10x faster than Jest for TypeScript; native ESM; compatible with the entire Fastify test ecosystem. `vitest.config.ts` targets all `*.test.ts` files in `/src`.

E2E: Playwright 1.50.1

- Docs: https://playwright.dev/docs/intro
- License: Apache-2.0
- Test scenarios:
  - Full case creation → document upload → witness setup → session config flow
  - Live session: objection alert fires within 1.5s of question delivery
  - Coaching brief generation and PDF download
  - Witness token access (no login required)

Coverage: Istanbul via Vitest

- Minimum coverage thresholds:
  - Statements: 70%
  - Branches: 65%
  - Functions: 75%
  - Lines: 70%

---

7. DEVELOPMENT TOOLS

Linter

ESLint 9.20.0

- Docs: https://eslint.org/docs/latest/
- Config: `eslint.config.mjs` (flat config, ESLint v9 format)

```
// eslint.config.mjs
import tseslint from 'typescript-eslint';
import nextPlugin from '@next/eslint-plugin-next';

export default tseslint.config(
```

```
  {
    files: ['**/*.{ts,tsx}'],
    extends: [
      ...tseslint.configs.strictTypeChecked,
      ...tseslint.configs.stylisticTypeChecked,
    ],
    plugins: { '@next/next': nextPlugin },
    rules: {
      '@typescript-eslint/no-explicit-any': 'error',
      '@typescript-eslint/no-unused-vars': ['error', { argsIgnorePattern: '^_' }],
      '@typescript-eslint/consistent-type-imports': 'error',
      '@next/next/no-img-element': 'error',
      'no-console': ['warn', { allow: ['warn', 'error'] }],
    },
  }
);
```

Formatter

Prettier 3.4.2

- Docs: https://prettier.io/docs/en/
- Config: `.prettierrc`

```
{
  "semi": true,
  "singleQuote": true,
  "tabWidth": 2,
  "trailingComma": "es5",
  "printWidth": 100,
  "bracketSpacing": true,
  "arrowParens": "avoid",
  "plugins": ["prettier-plugin-tailwindcss"]
}
```

prettier-plugin-tailwindcss 0.6.11 (auto-sorts Tailwind class names)

Git Hooks

Husky 9.1.7

- Docs: https://typicode.github.io/husky/
- License: MIT

.husky/pre-commit #!/bin/sh npx lint-staged # .husky/commit-msg #!/bin/sh npx --no-install commitlint --edit "$1"

lint-staged 15.4.3 (runs ESLint + Prettier only on staged files)

```
// .lintstagedrc.json
{
  "*.{ts,tsx}": ["eslint --fix", "prettier --write"],
  "*.{json,md,yaml}": ["prettier --write"],
  "prisma/schema.prisma": ["prisma validate"]
}
```

@commitlint/cli 19.7.1 + @commitlint/config-conventional 19.7.0

- Enforces Conventional Commits format on commit messages

IDE

Recommended: VS Code

Required extensions (`.vscode/extensions.json`):

```
{
  "recommendations": [
    "dbaeumer.vscode-eslint",
    "esbenp.prettier-vscode",
    "bradlc.vscode-tailwindcss",
    "prisma.prisma",
    "ms-playwright.playwright",
    "vitest.explorer",
    "eamodio.gitlens"
  ]
}
```

VS Code settings (`.vscode/settings.json`):

```
{
  "editor.formatOnSave": true,
  "editor.defaultFormatter": "esbenp.prettier-vscode",
  "editor.codeActionsOnSave": {
    "source.fixAll.eslint": "explicit"
  },
  "typescript.preferences.importModuleSpecifier": "non-relative",
  "tailwindCSS.experimental.classRegex": [
    ["cva\\(([^)]*)\\)", "[\"'`]([^\"'`]*).*?[\"'`]"]
  ]
}
```

---

8. ENVIRONMENT VARIABLES

Frontend (Next.js — `.env.local`)

App NEXT_PUBLIC_APP_URL="http://localhost:3000" NEXT_PUBLIC_API_URL="http://localhost:4000" NEXT_PUBLIC_APP_ENV="development" # Socket.io NEXT_PUBLIC_SOCKET_URL="http://localhost:4000" # Sentry (frontend) NEXT_PUBLIC_SENTRY_DSN="https://xxx@ooo.ingest.sentry.io/xxx" SENTRY_ORG="voiceflow-intelligence" SENTRY_PROJECT="verdict-frontend" SENTRY_AUTH_TOKEN="sntrys_xxx" # PostHog Analytics NEXT_PUBLIC_POSTHOG_KEY="phc_xxx" NEXT_PUBLIC_POSTHOG_HOST="https://app.posthog.com" # MediaPipe (Behavioral Sentinel) NEXT_PUBLIC_MEDIAPIPE_MODEL_URL="/models/face_landmarker.task" # Feature Flags NEXT_PUBLIC_BEHAVIORAL_SENTINEL_ENABLED="false"

Backend (Fastify — `.env`)

App NODE_ENV="development" PORT="4000" API_URL="http://localhost:4000" FRONTEND_URL="http://localhost:3000" LOG_LEVEL="debug" # Database DATABASE_URL="postgresql://verdict:password@localhost:5432/verdict_dev" DATABASE_POOL_MIN="2" DATABASE_POOL_MAX="10" # Redis REDIS_URL="redis://localhost:6379" # JWT JWT_SECRET="a-256-bit-cryptographically-random-secret-minimum-32-chars" JWT_ACCESS_EXPIRES_IN="8h" JWT_REFRESH_SECRET="another-256-bit-cryptographically-random-secret" JWT_REFRESH_EXPIRES_IN="30d" # SAML SSO SAML_CALLBACK_URL="http://localhost:4000/auth/saml/callback" SAML_CERT="-----BEGIN CERTIFICATE-----\nMII..." # SP certificate # Witness & Brief Tokens (nanoid-generated, no JWT) WITNESS_TOKEN_TTL_HOURS="72" BRIEF_SHARE_TOKEN_TTL_DAYS="7" # AWS S3

AWS_REGION="us-east-1" AWS_ACCESS_KEY_ID="AKIA..." AWS_SECRET_ACCESS_KEY="xxx" S3_BUCKET_NAME="verdict-documents-dev" S3_PRESIGNED_URL_EXPIRES_SECONDS="3600" # Anthropic Claude ANTHROPIC_API_KEY="sk-ant-xxx" ANTHROPIC_MODEL="claude-sonnet-4-20250514" ANTHROPIC_MAX_TOKENS="4096" # ElevenLabs ELEVENLABS_API_KEY="xi_xxx" ELEVENLABS_INTERROGATOR_VOICE_ID="pNInz6obpgDQGcFmaJgB" ELEVENLABS_COACH_VOICE_ID="21m00Tcm4TlvDq8ikWAM" ELEVENLABS_WEBSOCKET_URL="wss://api.elevenlabs.io/v1/convai/conversation" # NVIDIA Nemotron NEMOTRON_API_KEY="nvapi-xxx" NEMOTRON_BASE_URL="https://integrate.api.nvidia.com/v1" NEMOTRON_MODEL="nvidia/llama-3.1-nemotron-ultra-253b-v1" NEMOTRON_TIMEOUT_MS="5000" # Nozomio Nia NIA_API_KEY="nia_xxx" NIA_BASE_URL="https://api.nozomio.com/v1" NIA_FRE_CORPUS_INDEX_ID="verdict-fre-rules-v1" # Databricks DATABRICKS_HOST="https://xxx.azuredatabricks.net" DATABRICKS_TOKEN="dapi_xxx" DATABRICKS_SQL_WAREHOUSE_ID="xxx" DATABRICKS_CATALOG="verdict" DATABRICKS_SCHEMA="sessions" # Resend (Email) RESEND_API_KEY="re_xxx" RESEND_FROM_EMAIL="noreply@verdict.law" RESEND_FROM_NAME="VERDICT Platform" # Sentry (backend) SENTRY_DSN="https://xxx@ooo.ingest.sentry.io/xxx" # Puppeteer (PDF generation) PUPPETEER_EXECUTABLE_PATH="/usr/bin/google-chrome-stable" # production Docker # PUPPETEER_EXECUTABLE_PATH="" # local: auto-detected # Rate Limiting RATE_LIMIT_OBJECTION_COPILOT_PER_MINUTE="120" RATE_LIMIT_INCONSISTENCY_DETECTOR_PER_MINUTE="60" RATE_LIMIT_AUTH_PER_MINUTE="20" RATE_LIMIT_UPLOAD_PER_HOUR="50"

---

9. PACKAGE.JSON SCRIPTS

Root Workspace `package.json`

```
{
  "scripts": {
    "dev": "concurrently \"npm run dev:frontend\" \"npm run dev:backend\"",
    "dev:frontend": "cd apps/frontend && next dev",
    "dev:backend": "cd apps/backend && tsx watch src/index.ts",

    "build": "npm run build:frontend && npm run build:backend",
    "build:frontend": "cd apps/frontend && next build",
    "build:backend": "cd apps/backend && tsc -p tsconfig.build.json",

    "start": "concurrently \"npm run start:frontend\" \"npm run start:backend\"",
    "start:frontend": "cd apps/frontend && next start",
    "start:backend": "cd apps/backend && node dist/index.js",

    "typecheck": "npm run typecheck:frontend && npm run typecheck:backend",
    "typecheck:frontend": "cd apps/frontend && tsc --noEmit",
    "typecheck:backend": "cd apps/backend && tsc --noEmit",

    "lint": "eslint . --max-warnings 0",
    "lint:fix": "eslint . --fix",
    "format": "prettier --write .",
    "format:check": "prettier --check .",

    "test": "npm run test:unit && npm run test:integration",
    "test:unit": "vitest run --project unit",
    "test:integration": "vitest run --project integration",
    "test:e2e": "playwright test",
    "test:e2e:ui": "playwright test --ui",
    "test:coverage": "vitest run --coverage",
    "test:watch": "vitest",

    "db:generate": "prisma generate",
    "db:migrate": "prisma migrate dev",
    "db:migrate:deploy": "prisma migrate deploy",
    "db:push": "prisma db push",
    "db:seed": "tsx prisma/seed.ts",
```

```
    "db:reset": "prisma migrate reset --force",
    "db:studio": "prisma studio",
    "db:validate": "prisma validate",
    "db:format": "prisma format",

    "deploy:staging": "vercel deploy --env-file .env.staging && flyctl deploy --config fly.sta
    "deploy:production": "vercel deploy --prod && flyctl deploy --config fly.production.toml",

    "agents:test:interrogator": "tsx scripts/test-agents/interrogator.ts",
    "agents:test:objection": "tsx scripts/test-agents/objection-copilot.ts",
    "agents:test:inconsistency": "tsx scripts/test-agents/inconsistency-detector.ts",
    "agents:test:all": "tsx scripts/test-agents/run-all.ts",

    "nia:index:fre": "tsx scripts/nia/index-fre-corpus.ts",
    "nia:index:case": "tsx scripts/nia/index-case-documents.ts",

    "prepare": "husky"
  }
}
```

---

10. DEPENDENCIES LOCK

Frontend Dependencies (exact versions)

```
{
  "dependencies": {
    "next": "15.1.6",
    "react": "19.0.0",
    "react-dom": "19.0.0",
    "typescript": "5.7.3",
    "tailwindcss": "3.4.17",
    "framer-motion": "12.4.7",
    "zustand": "5.0.2",
    "@tanstack/react-query": "5.66.0",
    "axios": "1.7.9",
    "react-hook-form": "7.54.2",
    "zod": "3.24.1",
    "@hookform/resolvers": "3.10.0",
    "socket.io-client": "4.8.1",
    "recharts": "2.15.0",
    "wavesurfer.js": "7.8.11",
    "@react-pdf/renderer": "4.1.5",
    "@mediapipe/tasks-vision": "0.10.20",
    "@radix-ui/react-alert-dialog": "1.1.6",
    "@radix-ui/react-dialog": "1.1.6",
    "@radix-ui/react-dropdown-menu": "2.1.6",
    "@radix-ui/react-label": "2.1.2",
    "@radix-ui/react-progress": "1.1.2",
    "@radix-ui/react-select": "2.1.6",
    "@radix-ui/react-switch": "1.1.3",
    "@radix-ui/react-tabs": "1.1.3",
    "@radix-ui/react-toast": "1.2.6",
    "@radix-ui/react-tooltip": "1.1.8",
    "class-variance-authority": "0.7.1",
    "clsx": "2.1.1",
    "tailwind-merge": "2.6.0",
    "lucide-react": "0.477.0",
    "@sentry/nextjs": "8.51.0",
    "posthog-js": "1.219.0"
  },
  "devDependencies": {
    "@types/node": "22.13.4",
    "@types/react": "19.0.8",
    "@types/react-dom": "19.0.3",
    "eslint": "9.20.0",
```

```
    "eslint-config-next": "15.1.6",
    "typescript-eslint": "8.24.1",
    "prettier": "3.4.2",
    "prettier-plugin-tailwindcss": "0.6.11",
    "vitest": "2.1.8",
    "@vitejs/plugin-react": "4.3.4",
    "@vitest/coverage-v8": "2.1.8",
    "@playwright/test": "1.50.1",
    "husky": "9.1.7",
    "lint-staged": "15.4.3",
    "@commitlint/cli": "19.7.1",
    "@commitlint/config-conventional": "19.7.0",
    "autoprefixer": "10.4.20",
    "postcss": "8.5.1"
  }
}
```

Backend Dependencies (exact versions)

```
{
  "dependencies": {
    "fastify": "5.2.1",
    "@fastify/cors": "10.0.2",
    "@fastify/jwt": "9.0.1",
    "@fastify/multipart": "9.0.3",
    "@fastify/rate-limit": "10.2.1",
    "@fastify/socket.io": "3.0.0",
    "socket.io": "4.8.1",
    "typescript": "5.7.3",
    "zod": "3.24.1",
    "@prisma/client": "6.3.1",
    "ioredis": "5.4.1",
    "@anthropic-ai/sdk": "0.36.3",
    "elevenlabs": "1.14.0",
    "axios": "1.7.9",
    "@databricks/sql": "1.10.0",
    "@aws-sdk/client-s3": "3.726.1",
    "@aws-sdk/s3-request-presigner": "3.726.1",
    "@aws-sdk/lib-storage": "3.726.1",
    "jsonwebtoken": "9.0.2",
    "@types/jsonwebtoken": "9.0.9",
    "passport": "0.7.0",
    "@node-saml/passport-saml": "5.0.1",
    "bcrypt": "5.1.1",
    "@types/bcrypt": "5.0.2",
    "nanoid": "5.0.9",
    "resend": "4.1.2",
    "react-email": "3.0.7",
    "@react-email/components": "0.0.31",
    "pdf-parse": "1.1.1",
    "mammoth": "1.8.0",
    "puppeteer": "22.15.0",
    "pino": "9.6.0",
    "@sentry/node": "8.51.0",
    "dotenv": "16.4.7",
    "uuid": "11.0.5",
    "@types/uuid": "10.0.0"
  },
  "devDependencies": {
    "prisma": "6.3.1",
    "tsx": "4.19.2",
    "@types/node": "22.13.4",
    "@types/passport": "1.0.17",
    "@types/pdf-parse": "1.1.4",
    "typescript": "5.7.3",
    "typescript-eslint": "8.24.1",
    "eslint": "9.20.0",
```

```
    "prettier": "3.4.2",
    "vitest": "2.1.8",
    "@vitest/coverage-v8": "2.1.8",
    "husky": "9.1.7",
    "lint-staged": "15.4.3"
  }
}
```

---

11. SECURITY CONSIDERATIONS

Authentication Flow

```
ATTORNEY LOGIN (Email/Password):
  1. POST /auth/login { email, password }
  2. bcrypt.compare(password, user.passwordHash) — cost factor 12
  3. IF valid: issue JWT access token (8h) + refresh token (30d, stored in Redis)
  4. SET access token in httpOnly cookie (SameSite=Strict, Secure, Path=/)
  5. Return: { user: { id, name, role, firmId } }

ATTORNEY LOGIN (SSO/SAML):
  1. GET /auth/saml → redirect to firm IdP
  2. IdP authenticates → POST /auth/saml/callback (SAML assertion)
  3. Passport-SAML validates assertion signature
  4. Lookup or provision user by email claim
  5. Issue JWT + refresh token (same as above)

WITNESS ACCESS (Token):
  1. Attorney generates witness token: nanoid(24) → Redis SET witness:{token} sessionId EX 259
  2. Witness receives URL: /witness/session/:sessionId?token=:token
  3. Server: Redis GET witness:{token} → validates sessionId match
  4. Session state updated: witnessJoined=true → token not revoked (witness can reconnect)
  5. Token marked invalidated only on session COMPLETE (one-time use per complete session)

TOKEN REFRESH:
  1. Access token expires → Axios interceptor catches 401
  2. POST /auth/refresh { refreshToken: from httpOnly cookie }
  3. Validate refresh token in Redis (checks: exists, not revoked, user still active)
  4. Issue new access token → return in response header + new httpOnly cookie
  5. Original request retried with new token

LOGOUT:
  1. POST /auth/logout
  2. Redis DEL refresh:{userId}:{tokenId} (revokes this session only)
  3. Clear httpOnly cookie
  4. For "revoke all sessions": Redis SCAN + DEL all refresh:{userId}:* keys
```

Password Hashing

```
// bcrypt cost factor 12 — ~350ms hash time on modern hardware
// Justification: Legal enterprise tool; performance cost acceptable at login
// Do NOT go below cost factor 10 in any environment

import bcrypt from 'bcrypt';

const BCRYPT_ROUNDS = 12;

export const hashPassword = (password: string): Promise<string> =>
  bcrypt.hash(password, BCRYPT_ROUNDS);

export const verifyPassword = (password: string, hash: string): Promise<boolean> =>
  bcrypt.compare(password, hash);
```

Token Expiry Times

Token Type	TTL	Storage	Revocation
JWT Access Token	8 hours	httpOnly cookie (Secure, SameSite=Strict)	None (stateless — short TTL is the control)
JWT Refresh Token	30 days	httpOnly cookie + Redis for revocation	Redis DEL on logout; Redis SCAN DEL on "revoke all"
Witness Practice Token	72 hours	Redis with TTL	Auto-expiry
Brief Share Token	7 days	PostgreSQL `shareTokenExpiresAt`	DB expiry check on every access
SAML Session	IdP-controlled	httpOnly cookie	SAML SLO (Single Logout)

CORS Configuration

```
// apps/backend/src/plugins/cors.ts
await app.register(cors, {
  origin: [
    process.env.FRONTEND_URL!,        // https://verdict.law
    'https://preview.verdict.law',    // Vercel preview URLs — restricted in production
  ],
  credentials: true,                  // Required for httpOnly cookie auth
  methods: ['GET', 'POST', 'PATCH', 'DELETE', 'OPTIONS'],
  allowedHeaders: ['Content-Type', 'Authorization', 'x-request-id'],
  exposedHeaders: ['x-request-id'],
  maxAge: 86400,                       // 24h preflight cache
});
```

Rate Limiting

```
// Rate limits per route (backed by Redis)
const rateLimits = {
  'POST /auth/login':                 { max: 20,  timeWindow: '1 minute' },
  'POST /auth/refresh':               { max: 30,  timeWindow: '1 minute' },
  'POST /cases':                      { max: 10,  timeWindow: '1 minute' },
  'POST /cases/:id/documents':        { max: 50,  timeWindow: '1 hour' },
  'POST /sessions/:id/objection':     { max: 120, timeWindow: '1 minute' },
  'POST /sessions/:id/inconsistency': { max: 60,  timeWindow: '1 minute' },
  'POST /sessions/:id/behavioral':    { max: 300, timeWindow: '1 minute' },
  'POST /briefs/:id/pdf':             { max: 10,  timeWindow: '1 minute' },
  'GET /api/*':                       { max: 200, timeWindow: '1 minute' },
};
```

Data Encryption

```
AT REST:
  PostgreSQL: AES-256 encryption via AWS RDS encryption (production)
  S3 documents: SSE-S3 (AES-256, AWS-managed keys) — per-case key policy
  Redis: ElastiCache encryption at rest (AES-256, AWS KMS)
  Behavioral Sentinel AU vectors: AES-256 in PostgreSQL JSONB column

IN TRANSIT:
```

```
  All endpoints: TLS 1.3 minimum (TLS 1.2 rejected)
  WebSocket: WSS (TLS-encrypted Socket.io)
  S3 pre-signed URLs: HTTPS only; expire in 3600 seconds
  Internal service calls (backend → AI APIs): HTTPS only

BEHAVIORAL SENTINEL SPECIFIC:
  - Raw video: NEVER transmitted or stored
  - AU vectors (numeric arrays): transmitted over WSS, stored encrypted in PostgreSQL
  - Auto-deleted at 90-day retention limit alongside session data (Prisma scheduled job)
  - Attorney must explicitly enable per-case; witness must grant camera permission
```

Security Headers (Next.js middleware)

```
// middleware.ts
const securityHeaders = {
  'Strict-Transport-Security': 'max-age=63072000; includeSubDomains; preload',
  'X-Content-Type-Options': 'nosniff',
  'X-Frame-Options': 'DENY',
  'X-XSS-Protection': '1; mode=block',
  'Referrer-Policy': 'strict-origin-when-cross-origin',
  'Content-Security-Policy': [
    "default-src 'self'",
    "script-src 'self' 'unsafe-eval' 'unsafe-inline'",  // required for Next.js
    "connect-src 'self' wss://verdict.law https://api.elevenlabs.io https://integrate.api.nvid
    "media-src 'self' blob:",  // ElevenLabs audio blobs
    "worker-src 'self' blob:",  // MediaPipe WASM worker
  ].join('; '),
  'Permissions-Policy': 'camera=(self), microphone=(self), geolocation=()',
};
```

---

12. VERSION UPGRADE POLICY

Patch Versions (`X.Y.Z → X.Y.Z+1`)

Policy: Apply within 7 days of release if it includes security fixes; within 30 days otherwise.

Process:

1. `npm audit` daily via GitHub Actions (automated Dependabot PRs for patch bumps)
2. CI must pass (unit + integration tests)
3. Deploy to staging → smoke test live session flow → merge to main

No manual approval required for patch versions.

Minor Versions (`X.Y → X.Y+1`)

Policy: Apply within 60 days of release. Review changelog for deprecation notices.

Process:

1. Create feature branch: `chore/upgrade-fastify-5.2-to-5.3`
2. Update `package.json` (exact version, no ranges)
3. Run full test suite: unit + integration + E2E
4. Manual smoke test of: live session (WebSocket), document upload, brief PDF generation, SAML login
5. Staging deploy → 48-hour observation window
6. PR with changelog summary → 1 reviewer approval → merge

Major Versions (`X → X+1`)

Policy: Evaluate quarterly. Migrate only when current major version approaches End of Life or when a breaking change in a major dependency forces it.

Required before major version adoption:

- Official migration guide published by library maintainer
- All critical dependencies compatible with new major (check `npm-check-updates`)
- Full E2E test suite passes with zero regressions
- Staging deploy → 1-week observation period
- Rollback plan documented (previous Docker image pinned in `fly.toml`)
- Team consensus required (2/4 team members must approve)

Critical major upgrades on horizon:

- **Next.js 16** (when released): Evaluate App Router breaking changes
- **Prisma 7** (when released): Migration tooling changes expected
- **Socket.io 5** (when released): Check breaking changes to Redis adapter

Security Patch Policy (Emergency)

Definition: Any CVE rated HIGH (≥7.0) or CRITICAL (≥9.0) affecting a direct or transitive dependency.

Response time:

- CRITICAL: Apply within 24 hours, deploy to production immediately after CI passes
- HIGH: Apply within 72 hours
- MEDIUM/LOW: Next scheduled patch cycle

Process:

1. GitHub Security Advisory triggers Dependabot alert → Slack notification
2. Engineer on-call evaluates: is the vulnerable code path reachable in VERDICT?
3. If yes: emergency branch → fix → CI → production deploy (skip staging observation window)
4. If no: document reasoning → apply in next scheduled patch cycle

Rollback Procedures

Frontend rollback (Vercel) vercel rollback --to [previous-deployment-url] # Backend rollback (Fly.io) flyctl releases list --app verdict-api flyctl deploy --image registry.fly.io/verdict-api:[previous-image-tag] # Database rollback (Prisma) # Note: Prisma Migrate has NO automatic rollback for data migrations # Prevention: all migrations are additive (add column, add table) in v1.0 # Destructive changes require a two-phase deploy and are tracked in MIGRATIONS.md npx prisma migrate resolve --rolled-back [migration_name] # Docker image pinning (emergency) # fly.toml build section: # [build] # image = "registry.fly.io/verdict-api:20260221-stable"

Dependency Audit Schedule

Task	Frequency	Owner	Tool
`npm audit`	Daily (CI)	Automated	GitHub Dependabot
Full dependency review	Monthly	Lead dev	`npm-check-updates`
License compliance check	Quarterly	Lead dev	`license-checker`
Security CVE scan	Weekly	Automated	Snyk (free tier)

QUICK REFERENCE

Tech Stack Summary Card

```
VERDICT — Tech Stack at a Glance
═══════════════════════════════════════════════════════════════
Frontend:   Next.js 15.1.6 + React 19 + TypeScript 5.7.3
Styling:    Tailwind CSS 3.4.17 + shadcn/ui 2.1.8
State:      Zustand 5.0.2 + TanStack Query 5.66.0
Real-time:  Socket.io-client 4.8.1
Charts:     Recharts 2.15.0
Animation:  Framer Motion 12.4.7
Face AI:    @mediapipe/tasks-vision 0.10.20 (client-side)
═══════════════════════════════════════════════════════════════
Backend:    Node.js 22.13.0 LTS + Fastify 5.2.1
Database:   PostgreSQL 17.2 (Prisma 6.3.1)
Cache:      Redis 7.4.1 (ioredis 5.4.1)
Storage:    AWS S3 (@aws-sdk/client-s3 3.726.1)
Auth:       JWT (jsonwebtoken 9.0.2) + SAML 2.0
Email:      Resend 4.1.2
PDF:        Puppeteer 22.15.0 (server) + @react-pdf/renderer 4.1.5 (preview)
═══════════════════════════════════════════════════════════════
AI Layer:   Claude SDK (@anthropic-ai/sdk 0.36.3)
            ElevenLabs (elevenlabs 1.14.0)
            NVIDIA Nemotron (REST via axios 1.7.9)
            Nozomio Nia (REST via axios 1.7.9)
            Databricks (@databricks/sql 1.10.0)
═══════════════════════════════════════════════════════════════
Hosting:    Vercel (frontend) + Fly.io (backend) [prod]
            Vercel + Railway [hackathon]
Testing:    Vitest 2.1.8 + Playwright 1.50.1
Monitoring: Sentry 8.51.0 + Pino 9.6.0 + PostHog
CI/CD:      GitHub Actions
═══════════════════════════════════════════════════════════════
```

TECH_STACK.md — VERDICT v1.0.0 — Hackathon Edition
B2B Deposition Coaching Platform — Team VoiceFlow Intelligence
NYU Startup Week Buildathon 2026 | AI Automation — August.law Sponsor Track