

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

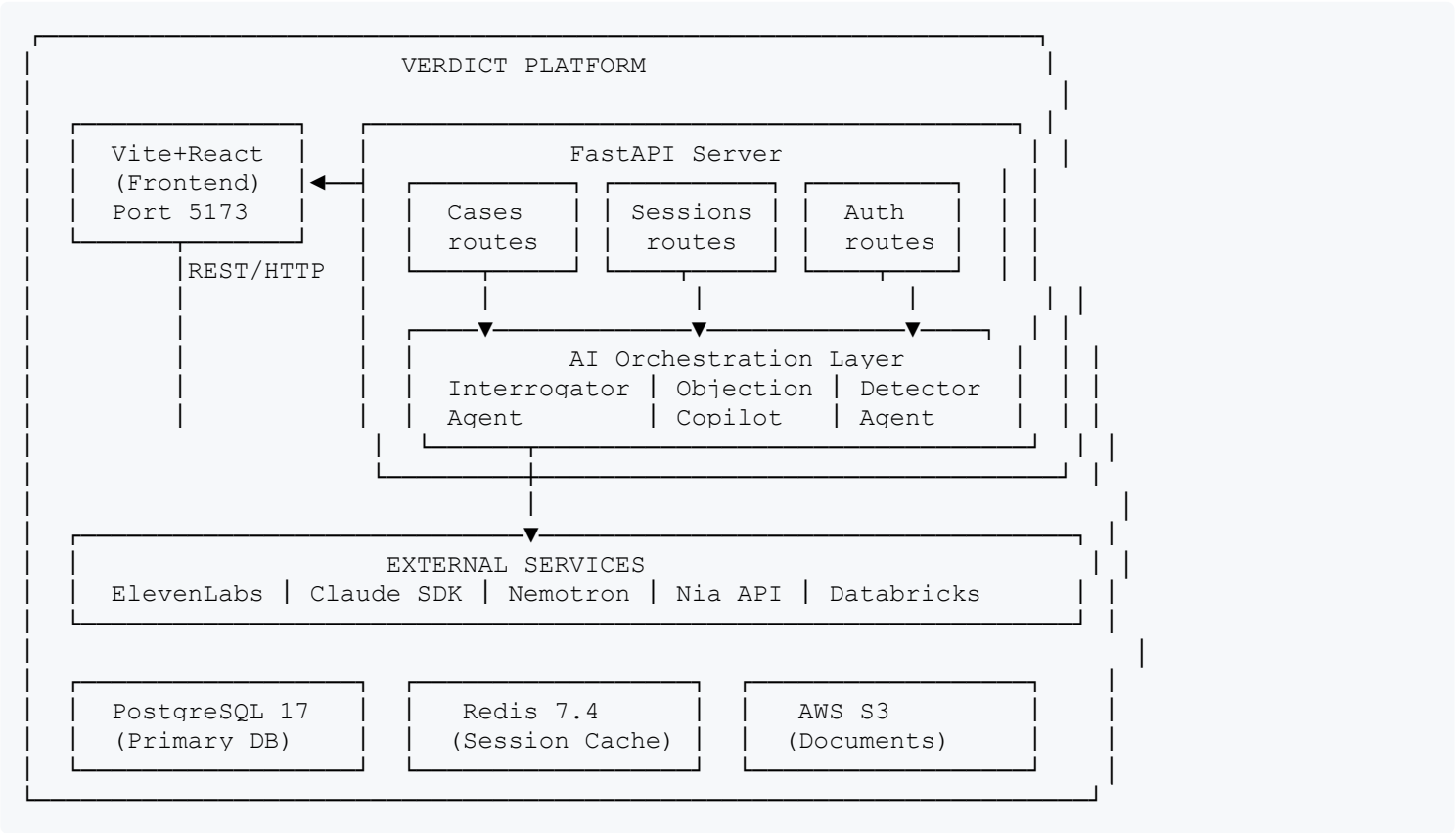
TABLE OF CONTENTS

- 1. [Stack Overview](#)
- 2. [Frontend Stack](#)
- 3. [Backend Stack](#)
- 4. [AI & External Service Integrations](#)
- 5. [Database Schema & Data Layer](#)
- 6. [DevOps & Infrastructure](#)
- 7. [Development Tools](#)
- 8. [Environment Variables](#)
- 9. [Package.json Scripts](#)
- 10. [Dependencies Lock](#)
- 11. [Security Considerations](#)
- 12. [Version Upgrade Policy](#)

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 |
|---|---|
| Vite + React SPA for frontend | Fastest iteration speed for hackathon; Vite's HMR is near-instant. Lovable-generated codebase uses this stack, so it was the natural choice. Future: add SSR via React Router v7 or migrate to Next.js when SSR becomes a priority. |
| FastAPI over Flask/Django | Native async/await for AI streaming; automatic OpenAPI docs; Pydantic v2 validation built-in. Objection Copilot fires $\leq 1.5s$ — 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. |
| REST + SSE over raw WebSocket (current) | Simpler to implement in hackathon window; SSE handles streaming AI responses from Interrogator Agent. Socket.io can be added later for bi-directional live session events. |
| 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

Current implementation: `verdict-frontend/design-first-focus/` (Vite + React SPA, Lovable-scaffolded)

Build Tool & Framework

Vite 5.4.19 + React 18.3.1

- Docs: <https://vitejs.dev/guide/> | <https://react.dev>
- License: MIT
- Reason: Lovable-generated codebase uses this stack. Vite's near-instant HMR maximises iteration speed in the 48-hour hackathon window. React 18 provides Concurrent Features for responsive UI during AI streaming responses.
- Future upgrade path: Migrate to Next.js 15 (App Router) post-hackathon if SSR / initial LCP become a priority.

Language

TypeScript 5.8.3

- Docs: <https://www.typescriptlang.org/docs/>
- License: Apache-2.0
- Reason: Strict typing across API response shapes catches integration bugs at compile time.
- Config: `tsconfig.app.json` (strict mode), `tsconfig.node.json` (Vite config), `tsconfig.json` (root references)

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.

UI Components

shadcn/ui (component collection — Radix UI primitives)

- Docs: <https://ui.shadcn.com/docs>
- License: MIT
- Installed via `components.json`. Full set in `src/components/ui/` (accordion, alert-dialog, badge, button, card, dialog, form, input, select, sheet, sidebar, tabs, toast, and more).
- Reason: Copy-paste components give full ownership. Radix UI primitives ensure WCAG 2.1 AA compliance (PRD §8.3).

Radix UI (via shadcn/ui — various `@radix-ui/*` packages)

- License: MIT

Form Handling

React Hook Form 7.61.1

- Docs: <https://react-hook-form.com/docs>
- License: MIT
- Reason: Uncontrolled inputs with minimal re-renders. Per-field validation on blur for case creation, session config, and witness setup forms.

Zod 3.25.x (validation schema, shared concept with backend)

- Docs: <https://zod.dev>
- License: MIT
- Reason: Schema-driven client-side validation aligned with backend Zod schemas.

@hookform/resolvers 3.10.0 — bridges React Hook Form + Zod

HTTP Client & Server State

Axios 1.13.x

- Docs: <https://axios-http.com/docs/intro>

- License: MIT
- Reason: Interceptors for JWT refresh on 401 and request cancellation.

TanStack Query (React Query) 5.83.0

- Docs: <https://tanstack.com/query/v5/docs>
- License: MIT
- Reason: Stale-while-revalidate caching for case list, witness profiles, and brief data. Prevents redundant fetches when navigating between tabs.

Routing

react-router-dom 6.30.1

- Docs: <https://reactrouter.com/en/main>
- License: MIT
- Reason: Client-side SPA routing. `ProtectedRoute` component in `src/components/auth/ProtectedRoute.tsx` handles JWT auth gate. Nested layouts via `src/components/layout/` (`AppShell`, `CaseLayout`, `PublicLayout`).

Charts

Recharts 2.15.4

- Docs: <https://recharts.org/en-US/api>
- License: MIT
- Reason: Native React components for the Weakness Map radar chart (P1.2). Pure SVG — accessible and print-friendly.

Notifications / Toasts

Sonner 1.7.4 — toast notifications (`src/components/ui/sonner.tsx`)

Date Handling

date-fns 3.6.0 — date formatting and manipulation

Planned / Future Frontend Additions

The following packages are specified in the PRD but **not yet installed** in the current MVP frontend — add when implementing those features:

| Package | Feature |
|--------------------------------------|---|
| <code>socket.io-client</code> | Live session bi-directional events |
| <code>framer-motion</code> | Alert rail animations, score count-up |
| <code>wavesurfer.js</code> | ElevenLabs audio waveform visualization |
| <code>@mediapipe/tasks-vision</code> | Behavioral Sentinel FACS (P1.4) |
| <code>@react-pdf/renderer</code> | Client-side brief PDF preview |
| <code>zustand</code> | Live session global state store |

3. BACKEND STACK

Runtime

Python 3.12

- Reason: Team's primary language. Excellent async support via asyncio. Native compatibility with all AI/ML SDKs (anthropic, elevenlabs, httpx).
- ASGI server: Uvicorn 0.34.0 with uvloop for high-performance async I/O.

Framework

FastAPI 0.115.6

- Docs: <https://fastapi.tiangolo.com>
- License: MIT
- Reason: Native async/await, automatic OpenAPI docs, Pydantic v2 validation built-in. StreamingResponse handles SSE for Interrogator question streaming. Dependency injection via Depends() maps cleanly to auth middleware pattern.
- Alternatives rejected:
 - **Flask**: Sync-first, requires extra work for async AI streaming.
 - **Django**: Too heavy for API-only service.

WebSocket Server

FastAPI WebSockets (built-in)

- Reason: Native WebSocket support in FastAPI via `WebSocket` class. No additional dependency needed. Redis pub/sub handles fan-out across multiple server instances.

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

SQLAlchemy 2.0.36 + Alembic 1.14.0

- Docs: <https://docs.sqlalchemy.org/en/20/> | <https://alembic.sqlalchemy.org>
- License: MIT / MIT
- Reason: Industry-standard Python ORM. SQLAlchemy 2.0 async API with asyncpg driver matches high-performance async requirements. Alembic handles migration history. All 11 models defined as Python classes in `app/models/`.
- Alternatives rejected:

- **Tortoise ORM**: Less mature, smaller ecosystem.
- **Django ORM**: Tied to Django framework.

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)

redis-py 5.2.1 (async)

- Docs: <https://redis-py.readthedocs.io>
- License: MIT
- Used via: `redis.asyncio.from_url(REDIS_URL)`
- Reason: Session event buffering (60-second auto-save, PRD §8.4). Redis pub/sub for WebSocket horizontal scaling. Rate limiting counters. Witness token storage with TTL (72-hour automatic expiry). Brief share token with 7-day TTL.

Authentication

JWT (JSON Web Tokens)

python-jose[cryptography] 3.3.0 — JWT encode/decode

- Docs: <https://python-jose.readthedocs.io>
- License: MIT

passlib[bcrypt] 1.7.4 — password hashing (bcrypt cost factor 12)

- Docs: <https://passlib.readthedocs.io>
- License: BSD

FastAPI security — HTTPBearer dependency for route protection

- Reason: Stateless auth for attorney sessions. Short-lived access tokens (8 hours) + long-lived refresh tokens. Claims include: `firmId`, `userId`, `role`, `email`.

nanoid 2.0.0 (token generation for witness links and brief share tokens)

- Docs: <https://github.com/puyuan/py-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 the API server 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 the API server.

@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.

Validation

Pydantic v2 (2.10.4) + pydantic-settings 2.7.0

- Docs: <https://docs.pydantic.dev/latest/>
- License: MIT
- Request/response schemas in `app/schemas/`
- Settings/env management in `app/config.py` via `BaseSettings`
- Frontend still uses Zod for client-side validation (unchanged)

python-multipart 0.0.20 (file upload handling)

- Docs: <https://multipart.fastapiexpert.com>
- License: Apache-2.0
- Reason: Required by FastAPI for `multipart/form-data` file uploads.

API Rate Limiting

slowapi (Redis-backed rate limiting via FastAPI middleware)

- Reason: Route-level rate limits for AI endpoints (Inconsistency Detector, Objection Copilot). Redis-backed so limits are shared across all server instances.

FastAPI CORSMiddleware (built-in via Starlette)

- Reason: CORS configuration via `app.add_middleware(CORSMiddleware, ...)` — no separate package needed.

4. AI & EXTERNAL SERVICE INTEGRATIONS

Anthropic Claude SDK (Primary AI Orchestration)

anthropic 0.40.0 (Python SDK)

- 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 1.13.5 (Python SDK)

- 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: `httpx 0.28.1` (no official Python 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: `httpx 0.28.1` (async client via `httpx.AsyncClient`)

- 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 (SQLAlchemy / original Prisma spec)

```
// 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:** Alembic (`alembic revision --autogenerate` for development, `alembic upgrade head` 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: python scripts/seed.py
```

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 FastAPI 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 FastAPI router docstrings 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-python@v5 with: python-version: '3.12' - uses: actions/setup-
node@v4 with: node-version: '22.13.0' cache: 'npm' - run: cd verdict-backend && pip install -r requirements.txt -
run: cd verdict-backend && alembic upgrade head env: DATABASE_URL:
postgres://postgres:test@localhost:5432/verdict_test - run: cd verdict-frontend/design-first-focus && npm ci -
run: cd verdict-frontend/design-first-focus && npm run lint - run: cd verdict-frontend/design-first-focus && npm
run test 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 Uvicorn/FastAPI container
- Auto-deploy from `main` via Railway GitHub integration
- Start command: `uvicorn app.main:app --host 0.0.0.0 --port $PORT`
- Internal URL exposed to Vercel via `BACKEND_URL` env var

Production v1.0 (Fly.io):

- 2× `performance-2x` machines (4 CPU, 8GB RAM)
- Deployed in `iad` (Virginia) and `lhr` (London) regions — proximity to US/UK law firms
- WebSocket horizontal scaling via Redis pub/sub (all machines share Redis events)

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: Python `logging` + Unicorn access logs

- 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. `vitest.config.ts` targets all `*.test.ts` files in the frontend `/src`. Backend uses `pytest` with `httpx.AsyncClient` for integration tests.

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

Linters

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 (FastAPI — `.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_PREIGNED_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="pNlnz6obpgDQGcFmaJgB"
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.azure.databricks.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 SCRIPTS

Repository layout (actual):

```
BUILDATHON-2026/
├── verdict-backend/           ← FastAPI (Python)
├── verdict-frontend/
│   └── design-first-focus/   ← Vite + React SPA
└── docs/
```

verdict-backend/ commands (Python/FastAPI)

Start dev server uvicorn app.main:app --reload --port 4000 # Run migrations alembic upgrade head # Seed database python scripts/seed.py # Generate new migration alembic revision --autogenerate -m "description" # Run server (production) python run.py

verdict-frontend/design-first-focus/package.json scripts (actual)

```
{
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "build:dev": "vite build --mode development",
    "lint": "eslint .",
    "preview": "vite preview",
    "test": "vitest run",
    "test:watch": "vitest"
  }
}
```

Convenience scripts (run from repo root with PowerShell)

```
# Start backend dev server (port 4000)
cd verdict-backend
uvicorn app.main:app --reload --port 4000

# Start frontend dev server (port 5173)
cd verdict-frontend/design-first-focus && npm run dev

# Run both in parallel (PowerShell)
Start-Process powershell -ArgumentList "cd verdict-backend; uvicorn app.main:app --reload --po
Start-Process powershell -ArgumentList "cd verdict-frontend/design-first-focus; npm run dev"

# Database operations (from verdict-backend/)
alembic upgrade head           # apply all migrations
python scripts/seed.py        # seed with demo data
alembic history                # view migration history
```

10. DEPENDENCIES LOCK

Frontend Dependencies — verdict-frontend/design-first-focus/package.json (actual installed)

```
{
  "dependencies": {
    "react": "^18.3.1",
```

```

"react-dom": "^18.3.1",
"react-router-dom": "^6.30.1",
"axios": "^1.13.5",
"@tanstack/react-query": "^5.83.0",
"react-hook-form": "^7.61.1",
"@hookform/resolvers": "^3.10.0",
"zod": "^3.25.x",
"recharts": "^2.15.4",
"lucide-react": "^0.462.0",
"sonner": "^1.7.4",
"next-themes": "^0.3.0",
"date-fns": "^3.6.0",
"class-variance-authority": "^0.7.1",
"clsx": "^2.1.1",
"tailwind-merge": "^2.6.0",
"tailwindcss-animate": "^1.0.7",
"vaul": "^0.9.9",
"cmdk": "^1.1.1",
"embla-carousel-react": "^8.6.0",
"input-otp": "^1.4.2",
"react-day-picker": "^8.10.1",
"react-resizable-panels": "^2.1.9",
"@radix-ui/react-accordion": "^1.2.11",
"@radix-ui/react-alert-dialog": "^1.1.14",
"@radix-ui/react-avatar": "^1.1.10",
"@radix-ui/react-checkbox": "^1.3.2",
"@radix-ui/react-dialog": "^1.1.14",
"@radix-ui/react-dropdown-menu": "^2.1.15",
"@radix-ui/react-label": "^2.1.7",
"@radix-ui/react-popover": "^1.1.14",
"@radix-ui/react-progress": "^1.1.7",
"@radix-ui/react-select": "^2.2.5",
"@radix-ui/react-separator": "^1.1.7",
"@radix-ui/react-slider": "^1.3.5",
"@radix-ui/react-slot": "^1.2.3",
"@radix-ui/react-switch": "^1.2.5",
"@radix-ui/react-tabs": "^1.1.12",
"@radix-ui/react-toast": "^1.2.14",
"@radix-ui/react-tooltip": "^1.2.7"
},
"devDependencies": {
  "vite": "^5.4.19",
  "@vitejs/plugin-react-swc": "^3.11.0",
  "typescript": "^5.8.3",
  "tailwindcss": "^3.4.17",
  "autoprefixer": "^10.4.21",
  "postcss": "^8.5.6",
  "eslint": "^9.32.0",
  "typescript-eslint": "^8.38.0",
  "eslint-plugin-react-hooks": "^5.2.0",
  "eslint-plugin-react-refresh": "^0.4.20",
  "vitest": "^3.2.4",
  "@testing-library/react": "^16.0.0",
  "@testing-library/jest-dom": "^6.6.0",
  "jsdom": "^20.0.3",
  "@types/react": "^18.3.23",
  "@types/react-dom": "^18.3.7",
  "@types/node": "^22.16.5",
  "lovable-tagger": "^1.1.13"
}
}

```

Backend Dependencies — `verdict-backend/requirements.txt` (actual installed)

```

fastapi==0.115.6
uvicorn[standard]==0.34.0
sqlalchemy==2.0.36

```

```
alembic==1.14.0
asynccpq==0.30.0
psycopq2-binary==2.9.10
redis==5.2.1
pydantic==2.10.4
pydantic-settings==2.7.0
python-jose[cryptography]==3.3.0
passlib[bcrypt]==1.7.4
python-multipart==0.0.20
anthropic==0.40.0
elevenlabs==1.13.5
httpx==0.28.1
python-dotenv==1.0.1
nanoid==2.0.0
boto3==1.35.0
```

Note: Backend: requirements.txt (pip) | Frontend: package-lock.json (npm) — unchanged

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

passlib[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 from passlib.context
import CryptContext pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto", bcrypt__rounds=12)

```
def hash_password(password: str) -> str: return pwd_context.hash(password) def verify_password(password: str, hash: str) -> bool: return pwd_context.verify(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.nvidia.com",
    "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 ($x.y.z \rightarrow x.y.z+1$)

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 ($x.y \rightarrow x.y+1$)

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 ($x \rightarrow x+1$)

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 \rightarrow 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
- **SQLAlchemy 3** (when released): Migration tooling changes expected
- **FastAPI 1.0** (when released): Check for breaking changes to dependency injection

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 \rightarrow Slack notification
2. Engineer on-call evaluates: is the vulnerable code path reachable in VERDICT?
3. If yes: emergency branch \rightarrow fix \rightarrow CI \rightarrow production deploy (skip staging observation window)
4. If no: document reasoning \rightarrow 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 (Alembic) # Note: Alembic has limited 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 `alembic downgrade -1` # Docker image pinning (emergency) # `fly.toml` build section: # `[build]` # `image = "registry.fly.io/verdict-api:20260221-stable"`

Dependency Audit Schedule

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

QUICK REFERENCE

Tech Stack Summary Card

| | | |
|---|--|---|
| VERDICT — Tech Stack at a Glance [✓ = built ○ = planned] | | |
| Frontend: | <div>✓ Vite 5.4.19 + React 18.3.1 + TypeScript 5.8.3</div> <div>✓ Tailwind CSS 3.4.17 + shadcn/ui (Radix UI)</div> <div>✓ react-router-dom 6.30.1</div> <div>✓ TanStack Query 5.83.0 + Axios 1.13.x</div> <div>✓ React Hook Form 7.61.1 + Zod</div> <div>✓ Recharts 2.15.4</div> <div>○ Socket.io-client (live session)</div> <div>○ Framer Motion, wavesurfer.js (polish)</div> <div>○ @mediapipe/tasks-vision (Behavioral Sentinel P1.4)</div> | |
| Backend: | <div>✓ Python 3.12 + FastAPI 0.115.6 (Uvicorn 0.34.0)</div> <div>✓ PostgreSQL 17 via SQLAlchemy 2.0.36 + Alembic 1.14.0</div> <div>✓ Redis 7.4 via redis-py 5.2.1 (async)</div> <div>✓ JWT (python-jose 3.3.0) + passlib[bcrypt] 1.7.4</div> <div>✓ AWS S3 (document storage via boto3 1.35.0)</div> <div>○ FastAPI WebSockets (live session)</div> <div>○ SAML 2.0 SSO, Resend email, brief PDF export</div> | |
| AI Layer: | <div>✓ Claude SDK (anthropic 0.40.0 Python)</div> <div>✓ ElevenLabs (elevenlabs 1.13.5 Python)</div> <div>✓ NVIDIA Nemotron (REST via httpx)</div> <div>✓ Nozomio Nia (REST via httpx)</div> <div>○ Databricks Delta Lake (analytics P1.2+)</div> | |
| Repo: | verdict-backend/ verdict-frontend/ design-first-focus/ docs/ | ← FastAPI (Python, port 4000) ← Vite SPA (port 5173) ← specs & guidelines |
| Hosting: | Railway (backend) + Vercel/Lovable (frontend) [hackathon] | |
| Testing: | Vitest 3.2.4 (frontend) python -m py_compile app/**/*.py (backend) | |
| Logging: | Python logging + uvicorn access logs | |

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