TECHNICAL ARCHITECTURE AUDIT

VERDICT PATH - TECHNICAL ARCHITECTURE AUDIT

WHAT IS THIS APPLICATION?

- HIPAA-compliant legal tech platform for civil litigation case management
- Serves three user types: Individual Users (clients/plaintiffs), Law Firms (attorneys), Medical Providers (healthcare practices)
- Uses a pirate/treasure map theme with gamification (coins, badges, achievements)
- Guides users through a 9-stage litigation roadmap

-----*Note: We have access to the codebase. This report focuses exclusively on the code-level technical architecture audit. A functional review will be conducted after the prerequisites are cleared, allowing the application to be run locally.*

TECHNOLOGY STACK

Frontend:

- React Native 0.76.9 with Expo SDK 52
- React 18.3.1, React Navigation 7.x
- Expo modules: expo-av 15.0.2 (deprecated), expo-notifications, expo-calendar, expo-video
- Stripe React Native SDK 0.38.6
- Firebase 12.6.0, AsyncStorage, Luxon

Backend:

- Node.js (>=18.0.0), Express 4.21.2
- PostgreSQL
- JWT authentication, bcryptjs, Multer
- Stripe 19.2.0, Twilio, Nodemailer
- AWS SDK v3 (S3 integration)
- Express-rate-limit, CORS

Infrastructure:

- Database: PostgreSQL (Railway/Neon)
- Deployment: Railway (production), Replit (development)
- File Storage: Local + AWS S3 (partial)

- Push Notifications: Expo Push Notification Service
- Real-time: Firebase Realtime Database

PURPOSE & BUSINESS MODEL

- Helps individuals navigate civil litigation
- Provides educational content, progress tracking, secure document storage
- Communication tools between clients, law firms, and medical providers
- Payment processing for settlements and disbursements

Business Model: Freemium with subscription tiers:

- Individual: Free, Basic ($2.99/mo), Premium ($4.99/mo)
- Law Firms: Free (5 clients), Basic ($49/mo, 6-20 clients), Premium ($99/mo, 21-50 clients)
- Medical Providers: Free (5 patients), Basic ($39/mo, 6-20 patients), Premium ($79/mo, 21+ patients)

IMPLEMENTED FUNCTIONALITIES

- **Authentication & User Management:** Multi-user registration, JWT (httpOnly cookies), password reset, session timeout/warning, account lockout (5 failed attempts), privacy/TOS/EULA acceptance, multi-user accounts for law firms/medical providers, activity logging/audit trails.
- **Litigation Roadmap:** 9-stage interactive journey, progress tracking, coin rewards, remote progress updates (law firms), read-only view (law firms), data entry forms, video tutorial integration (purchase pending).
- **Gamification:** Coins, daily login streaks (5-30 coins), 16 achievements (4 categories), 8 pirate-themed badges (4 rarity levels), leaderboards, coin-to-credit conversion (5,000 coins = $1, $5 lifetime cap), treasure chest, action video modals, 10 avatar types.
- **Push Notifications:** 22 templates (6 system + 10 law firm + 6 medical), Expo push with deep linking, multi-recipient, analytics, device management, timezone-aware quiet hours, functional queuing system, inbox/outbox, detail screens.
- **Task Management:** Attorney-assigned tasks, priorities (L/M/H), due dates, status tracking, coin rewards, templates, detail screens, reminder scheduler.
- **Calendar Integration:** Device calendar sync, event types, bi-directional sync, event sharing, request workflow, medical provider appointment booking, law firm calendar management.
- **Connection Management:** Individuals connect to law firms/medical providers via unique codes, real-time search, connection/disconnection.
- **Payment & Financial:** Stripe integration, subscription management, settlement

negotiations, disbursement processing, bank information management, Stripe Connect onboarding.
- **Chat/Messaging:** Real-time chat (all user types), chat list, conversations, new chat creation, unread counts.
- **HIPAA Compliance (Backend):** AES-256-GCM encryption, audit logging, RBAC (6 roles, 23 permissions), patient consent, account security, secure file storage, document access authorization.
- **Referral System:** Invite codes with coin rewards, validation, statistics.
- **Subscription Management:** Screens for all user types, tier-based feature access, status tracking.
- **Admin Portal:** Dashboard, user management, system diagnostics, activity monitoring.

Note: These are the intended, implemented functionalities. Full verification of working status will be possible once the prerequisite items (e.g., database setup, environment variables) are available to run the application locally.

HIGH-LEVEL CODE AUDIT

STRENGTHS

- Architecture: Clear separation of concerns (routes, controllers, services, middleware), modular structure (22 controllers, 19 services, 15 middleware files), three user types handled consistently, database schema well-structured (33+ tables)
- Security: HIPAA compliance infrastructure in place, AES-256-GCM encryption service functional, audit logging comprehensive, JWT authentication with httpOnly cookies, CORS protection with origin validation, rate limiting implemented, account lockout after failed attempts, document access authorization checks
- Feature Breadth: Many features implemented (60+ API endpoints), gamification elements working, multi-platform support (iOS, Android), 60+ screens
- Documentation: Extensive documentation files, security audit documents, deployment guides, database schema documentation

CRITICAL ISSUES

- Incomplete Core Features: Medical document upload frontend disabled (backend functional), core feature for medical providers, impact: medical providers cannot use app effectively through UI
- App source code needs Expo SDK upgradation. Currently it is using an older version of Expo SDK.
- Repository Folder structure: A common package.json utilised for both backend and app source code. It needs to be separated in different repositories for better organization.

- Technical Debt: 1,097+ console.log/console.error statements in backend, 29+ TODO/FIXME comments, debug code left in production, missing configuration files (drizzle.config.json), impact: performance degradation, log clutter, maintenance difficulty
- Error Handling Inconsistencies: Some endpoints missing explicit status codes, inconsistent error response formats, some try-catch blocks don't handle all error types, impact: difficult debugging and inconsistent user experience
- Database Management: Missing schema management configuration (drizzle.config.json), no clear migration strategy documented, risk of schema drift, impact: potential database inconsistencies

MEDIUM PRIORITY ISSUES

- Code Quality: Excessive console logging (1,097+ instances), debug statements in production, inconsistent code style, some functions too large (App.js is 2,300+ lines)
- Testing: No visible test suite, no automated testing infrastructure, manual testing only, risk of regressions
- Performance: No caching strategy visible, no database query optimization mentioned, large bundle sizes possible, no lazy loading for screens
- Monitoring & Observability: No structured logging (Winston/Pino), no APM, no error tracking (Sentry), difficult to debug production issues
- Environment Configuration: 21+ environment variables, no centralized configuration validation, missing env vars could cause runtime errors, no documentation of required vs optional vars
- Dependency Issues: expo-av is deprecated, needs migration to expo-audio and expo-video before SDK 54, future breaking changes risk

LOW PRIORITY ISSUES

- Code Organization: Some files very large (App.js, some screens), could benefit from more component extraction, some duplicate code patterns
- Documentation: While extensive, some docs are outdated, no API documentation (Swagger/OpenAPI), no component documentation

ASSESSMENT

What's Working Well

- Solid foundation with many features
- Strong security infrastructure
- Good separation of concerns
- Extensive feature set
- Notification queuing system exists and works
- Medical upload backend is functional

Areas Requiring Attention

- Core feature frontend disabled: medical document upload essential for medical providers
- Technical debt accumulating: 1,000+ console.logs, missing configs, 29+ TODOs
- No testing infrastructure: high regression risk
- Large codebase without clear testing strategy
- expo-av deprecation needs addressing before SDK 54

Production Readiness: 5/10

- Security: Good (6/10) - HIPAA infrastructure solid
- Deployment: Broken (2/10) - old code in production
- Features: Incomplete (6/10) - core feature frontend disabled
- Testing: Missing (0/10) - no test suite
- Code Quality: Needs work & optimization (5/10) - too much debug code
- Documentation: Good (8/10) - extensive but some outdated

FINAL VERDICT

- Feature-rich application with solid architecture and security foundations
- Critical deployment issues, incomplete core features (frontend disabled while backend works), and significant technical debt need attention
- Functional for basic use cases but not production-ready for medical providers (core feature frontend disabled) or users expecting all advertised features (many marked "Coming Soon")
- With focused effort on deployment, completing frontend features, and reducing technical debt, this could be a strong production application
- Currently a well-architected prototype that needs completion and polish

Key Findings:

- Backend is more complete than frontend for some features (medical uploads work in backend)
- Notification queuing system exists and is functional
- 1,000+ console.log statements need cleanup
- 29+ TODO items need attention
- expo-av deprecation requires migration planning
- Production deployment pipeline needs fixing

PREREQUISITES

VERDICT PATH - PREREQUISITES & CREDENTIALS REQUIRED

PostgreSQL Database:

- Database connection string (DATABASE_URL)
- Hosted PostgreSQL instance (Railway, Neon, Supabase, AWS RDS, or local)

Stripe Account:

- Stripe account access (test mode for development, live mode for production)
- We will retrieve the Secret Key, Publishable Key, and Webhook Secret from the account

AWS Account:

- AWS account with S3 access
- We will retrieve the Access Key ID, Secret Access Key, S3 bucket name, and region from the account

Firebase Project:

- Firebase project access
- We will retrieve the Service Account JSON, Realtime Database URL, and client-side configuration from the project

Email Service Account:

- SMTP service account (Gmail, SendGrid, Mailgun, AWS SES, etc.)
- We will retrieve the SMTP host, port, username, password, and from address from the account

Twilio Account:

- Twilio account access
- We will retrieve the Account SID, Auth Token, and phone number from the account

*****Please provide an ENV file, so that we can run the code!***