HYDRAHUNT Codebase Audit Report

Audit Date: February 25, 2026
Repository: https://github.com/C-Jay69/HYDRAHUNT_STITCH_Z_190126.git
Project Path: /home/ubuntu/hydrahunt

Executive Summary

HYDRAHUNT is a resume management platform with a cyber-tactical theme ("Career Warfare AI Platform"). The codebase has a **substantial foundation** with Next.js 16, TypeScript, Prisma ORM, and basic API routes already implemented. However, it requires significant work to become a fully functional, deployable platform.

Current State: ⚠️ Partially Functional (60-70% Frontend, 40% Backend)

- ✅ UI/UX design framework established (cyber-tactical theme)
- ✅ Basic database schema defined (Prisma/SQLite)
- ✅ Frontend components and routing structure in place
- ⚠️ Backend API routes exist but lack full implementation
- ⚠️ No authentication system actually working
- ❌ No database migrations run / no actual data persistence
- ❌ No environment configuration files present
- ❌ AI services not properly configured
- ❌ Resume parsing incomplete
- ❌ No deployment configuration

File Inventory

Core Application Files

File	Purpose	Status
`package.json`	Dependencies & scripts	✅ Complete
`index.html`	Entry HTML	✅ Complete
`App.tsx`	Legacy React Router app (Vite)	⚠️ Duplicate of Next.js
`index.tsx`	Legacy entry point	⚠️ Duplicate
`vite.config.ts`	Vite configuration	⚠️ Not used (Next.js project)
`next.config.ts`	Next.js configuration	✅ Basic setup
`tailwind.config.ts`	Tailwind CSS config	✅ Complete
`tsconfig.json`	TypeScript config	✅ Complete
`prisma/schema.prisma`	Database schema	✅ Well-designed

Next.js App Directory (`src/app/`)

File	Purpose	Status
`page.tsx`	Landing + Dashboard + Editor	✅ **Largest file (700+ lines)**
`layout.tsx`	Root layout	✅ Basic setup
`login/page.tsx`	Login/Signup page	⚠️ UI only, no real auth
`globals.css`	Global styles	✅ Complete
`api/route.ts`	Base API	❓ Purpose unclear
`api/resumes/route.ts`	Resume CRUD API	✅ Implemented
`api/resumes/[id]/route.ts`	Single resume ops	✅ Implemented
`api/analyze/route.ts`	AI Analysis API	✅ Uses Z-AI SDK
`api/analyze/transition/route.ts`	Career transition API	✅ Uses Z-AI SDK

Services (`services/`)

File	Purpose	Status
`gemini.ts`	Google Gemini AI integration	⚠️ Requires API key
`fileExtraction.ts`	PDF/DOCX text extraction	✅ Browser-based
`storage.ts`	Hybrid storage (local + Supabase)	⚠️ Supabase not configured
`supabase.ts`	Supabase client	❌ Missing credentials
`stripe.ts`	Stripe payment integration	❓ Not reviewed
`docxGenerator.ts`	Export to DOCX	❓ Not reviewed

Components (`src/components/ui/`)

Full shadcn/ui component library installed (50+ components):
- ✅ Button, Input, Textarea, Card, Badge, Tabs, Dialog, etc.
- ✅ All Radix UI primitives available

Reference HTML Designs (`hydrahunt_*/`)

15 static HTML mockups representing different pages/features:

Directory	Page Mockup	Has Screenshot
`hydrahunt_landing_page/`	Landing page	✅ `screen.png`
`hy-drahunt_resume_forge_weaponry/`	Resume editor	✅ `screen.png`
`hy-drahunt_strike_analysis_critique/`	Resume analysis	✅ `screen.png`
`hy-drahunt_kill_list_dashboard/`	Job tracking dashboard	✅ `screen.png`
`hy-drahunt_career_territory_map/`	Career transition	✅ `screen.png`
`hydrahunt_pricing_tiers/`	Pricing page	✅ `screen.png`
`hy-drahunt_ai_interview_drills_1/`	Interview prep (v1)	✅ `screen.png`
`hy-drahunt_ai_interview_drills_2/`	Interview prep (v2)	✅ `screen.png`
`hydrahunt_ai_payload_forge/`	AI content generation	✅ `screen.png`
`hy-drahunt_mission_briefing_email/`	Email templates	✅ `screen.png`
`hy-drahunt_version_vault_archive_1/`	Version history (v1)	✅ `screen.png`
`hy-drahunt_version_vault_archive_2/`	Version history (v2)	✅ `screen.png`
`hy-drahunt_strike_mission_log_1/`	Application log (v1)	✅ `screen.png`
	Application log (v2)	✅ `screen.png`

Directory	Page Mockup	Has Screenshot
`hydrahunt_strike_mission_log_2/`		
`hydrahunt_success_target_down/`	Success celebration	✅ `screen.png`

Types & Constants

File	Purpose	Status
`types.ts`	Root types (legacy)	⚠️ Duplicate
`src/types/hydranhunt.ts`	Main type definitions	✅ Comprehensive
`constants.ts`	Root constants (legacy)	⚠️ Duplicate
`src/constants/hydranhunt.ts`	Templates & mock data	✅ Complete

Contexts

File	Purpose	Status
`src/contexts/LanguageContext.tsx`	i18n support	✅ Basic impl
`contexts/AuthContext.tsx`	Legacy auth context	⚠️ Not integrated
`contexts/LanguageContext.tsx`	Duplicate language context	⚠️ Duplicate

Database (`db/` , `prisma/`)

File	Purpose	Status
`prisma/schema.prisma`	Database schema	✅ Well-designed

Schema Models:
- `User` - User accounts with subscription tiers
- `Resume` - Resume data with JSON fields for arrays
- `JobStrike` - Job application tracking
- `JobTarget` - Auto-hunt job results
- `AnalysisReport` - AI analysis versioning

Technology Stack Assessment

Frontend

Technology	Version	Status
Next.js	16.1.1	✅ Latest
React	19.0.0	✅ Latest
TypeScript	5.x	✅ Latest
Tailwind CSS	4.x	✅ Latest
shadcn/ui	Full library	✅ Complete
Framer Motion	12.x	✅ Installed
Recharts	2.15.x	✅ Installed
TanStack Query	5.82.x	✅ Installed
Zustand	5.x	✅ Installed

Backend

Technology	Version	Status
Prisma	6.11.1	✅ Installed
SQLite	Default	⚠️ Dev only
Next.js API Routes	16.x	✅ Basic routes
NextAuth.js	4.24.11	❌ Not configured
z-ai-web-dev-sdk	0.0.15	✅ Installed

AI Integration

Service	Status
Z-AI SDK	✅ Used in API routes
Google Gemini	⚠️ Requires API_KEY env
OpenAI	❌ Not integrated

What’s Working vs What Needs to Be Built

✅ Working (Frontend UI Only)

1. Landing page with cyber-tactical design
2. Dashboard layout with sidebar navigation
3. Resume editor with tabs (Personal, Experience, Education, Skills, AI Intel)
4. Basic file upload UI (PDF, DOCX, TXT)
5. Resume preview panel
6. Pricing page UI
7. Login/Signup forms (UI only)
8. Multi-language dropdown (i18n shell)

⚠️ Partially Working

1. **Resume API Routes** - Code exists but database not initialized
2. **AI Analysis Routes** - Uses Z-AI SDK but untested
3. **File Extraction** - Client-side PDF.js/Mammoth.js (no server parsing)
4. **Storage Service** - Hybrid local/Supabase but Supabase not configured

❌ Not Working / Needs Implementation

Authentication & Users

- [ ] NextAuth.js configuration
- [ ] OAuth providers (Google, GitHub)
- [ ] Session management
- [ ] User creation/profile management
- [ ] Protected routes middleware

Database

- [ ] Create `.env` file with DATABASE_URL
- [ ] Run `prisma migrate dev` to initialize
- [ ] Seed initial data
- [ ] Connect API routes to real database
- [ ] Production database (PostgreSQL recommended)

Resume Features

- [ ] Server-side resume parsing (not just client-side extraction)
- [ ] Resume storage in database (currently mock data)
- [ ] Resume versioning
- [ ] Resume templates rendering (PDF export)
- [ ] ATS optimization logic
- [ ] Resume beautification/formatting

AI Features

- [ ] Configure AI API keys (GEMINI_API_KEY or use Z-AI)
- [ ] Implement resume analysis pipeline
- [ ] Career transition analysis
- [ ] Course/qualification recommendations
- [ ] AI-powered improvement suggestions

Payments (Phase 1 optional)

- [ ] Stripe integration configuration
- [ ] Subscription tier enforcement
- [ ] Payment webhooks

Deployment

- [ ] Environment variables setup
- [ ] Production database
- [ ] Vercel/Netlify deployment config
- [ ] CORS and security headers
- [ ] Error handling & logging

Recommended Architecture for Backend/Database

Database Strategy

Development: SQLite (current)

Production: PostgreSQL on Supabase/Neon/PlanetScale

```
┌─────────────────────────────────────────────────────────────┐
│                    HYDRAHUNT ARCHITECTURE                    │
├─────────────────────────────────────────────────────────────┤
│                                                             │
│  ┌──────────────┐     ┌──────────────┐     ┌──────────────┐ │
│  │   Next.js    │────▶│  API Routes  │────▶│   Prisma     │ │
│  │   Frontend   │     │  (/api/*)    │     │    ORM       │ │
│  └──────────────┘     └──────────────┘     └──────────────┘ │
│         │                    │                    │         │
│         │                    │                    ▼         │
│         │             ┌──────────────┐     ┌──────────────┐ │
│         │             │              │     │  PostgreSQL  │ │
│         │             │   Z-AI SDK   │     │  (Supabase)  │ │
│         │             │   / Gemini   │     └──────────────┘ │
│         │             │              │                      │
│         │             └──────────────┘                      │
│         │                                                   │
│         ▼                                                   │
│  ┌──────────────┐                                           │
│  │  NextAuth    │◀──────── OAuth (Google, GitHub)           │
│  └──────────────┘                                           │
│                                                             │
└─────────────────────────────────────────────────────────────┘
```

API Endpoints Required

```
Authentication:
POST   /api/auth/[...nextauth]   - NextAuth handlers

Users:
GET    /api/users/me             - Current user profile
PATCH  /api/users/me             - Update profile

Resumes:
GET    /api/resumes              - List user's resumes
POST   /api/resumes              - Create resume
GET    /api/resumes/[id]         - Get single resume
PUT    /api/resumes/[id]         - Update resume
DELETE /api/resumes/[id]         - Delete resume
POST   /api/resumes/upload       - Parse uploaded file

Analysis:
POST   /api/analyze              - General analysis
POST   /api/analyze/ats          - ATS check
POST   /api/analyze/job-fit      - Job matching
POST   /api/analyze/transition   - Career transition

Export:
POST   /api/export/pdf           - Export to PDF
POST   /api/export/docx          - Export to DOCX
```

Gap Analysis for Phase 1 Requirements

Core Features Checklist

Feature	Requirement	Current State	Gap
Resume upload	Handle PDF, DOCX, TXT	⚠️ Client-side only	Need server parsing
Resume parsing	Extract structured data	⚠️ Basic Gemini prompt	Need robust parser
AI ATS optimization	Improve ATS score	❌ Not implemented	Full feature needed
AI beautification	Format and improve	❌ Not implemented	Full feature needed
Resume analysis	Provide insights	✅ API route exists	Test & refine
Resume editing	CRUD operations	✅ UI exists, API exists	Connect to real DB
Career transition	Courses/qualifications	✅ API route exists	Test & refine
User authentication	Login/signup	❌ UI only	Configure NextAuth
Data persistence	Store user data	❌ No DB initialized	Run migrations
Deployment ready	Vercel/Netlify	❌ No config	Create deployment

Critical Blockers

1. **No `.env` file** - Database and AI services cannot function
2. **Database not initialized** - No tables exist
3. **Authentication not configured** - Users cannot create accounts
4. **AI API keys missing** - Analysis features won’t work

Specific Action Items

Immediate (Must Do First)

1. **Create environment configuration**
```
bash
   # Create .env file
   DATABASE_URL="file:./dev.db"
   NEXTAUTH_SECRET="generate-a-secret"
   NEXTAUTH_URL="http://localhost:3000"
   # Add AI keys as needed
```
2. **Initialize database**
```
bash
```

```
   npx prisma generate
   npx prisma db push
```

3. **Configure NextAuth.js**
   - Create `src/app/api/auth/[...nextauth]/route.ts`
   - Add OAuth providers (Google at minimum)
   - Create auth middleware

4. **Test API routes**
   - Verify resume CRUD works with database
   - Test AI analysis endpoints

Phase 1 Implementation Tasks

Priority	Task	Effort
P0	Environment setup (.env)	30 min
P0	Database initialization	30 min
P0	NextAuth.js configuration	2-3 hrs
P1	Resume upload & server parsing	4-6 hrs
P1	Connect frontend to real API	3-4 hrs
P1	AI analysis integration testing	2-3 hrs
P2	Resume templates/PDF export	4-6 hrs
P2	ATS optimization feature	4-6 hrs
P2	Error handling & validation	2-3 hrs
P3	Deployment configuration	2-3 hrs
P3	Production database setup	1-2 hrs

Files to Create

1. `.env` and `.env.example`
2. `src/app/api/auth/[...nextauth]/route.ts`
3. `src/middleware.ts` (auth protection)
4. `src/lib/auth.ts` (auth utilities)
5. `vercel.json` or `netlify.toml`

Files to Clean Up

1. Remove duplicate files in root (`App.tsx` , `index.tsx` , `types.ts` , `constants.ts`)
2. Remove legacy `contexts/` folder (use `src/contexts/`)

3. Remove `vite.config.ts` (not needed for Next.js)
4. Clean up unused `skills/` folder (appears to be template code)

Deployment Recommendations

For Vercel (Recommended)

- Native Next.js support
- Easy environment variable management
- Serverless functions for API routes
- Built-in CI/CD

Database Options

1. **Supabase** - PostgreSQL + Auth + Realtime (free tier available)
2. **Neon** - Serverless PostgreSQL
3. **PlanetScale** - MySQL with branching

Environment Variables Needed

```
DATABASE_URL=
NEXTAUTH_SECRET=
NEXTAUTH_URL=
GOOGLE_CLIENT_ID=
GOOGLE_CLIENT_SECRET=
# Optional
GEMINI_API_KEY=
STRIPE_SECRET_KEY=
```

Conclusion

The HYDRAHUNT codebase has a solid foundation with modern technologies and well-designed UI mockups. The primary gaps are:

1. **Infrastructure** - No environment config, database not initialized
2. **Authentication** - NextAuth installed but not configured
3. **Data Persistence** - All data is currently mock/local storage
4. **AI Integration** - Services defined but not properly connected

Estimated effort to reach MVP (Phase 1): 20-30 hours of development work

The HTML mockups in `hydrahunt_*/` folders provide excellent reference designs that should guide the UI implementation as features are built out.

Generated by HYDRAHUNT Audit System