HCTS Platform Architecture Overview

Table of Contents

System Overview

The HCTS (Healthcare Trading System) Platform is a comprehensive B2B marketplace and service management system designed for healthcare providers, insurance companies, and intermediaries. The platform enables secure trading of healthcare services, certificate management, payment processing, and compliance monitoring.

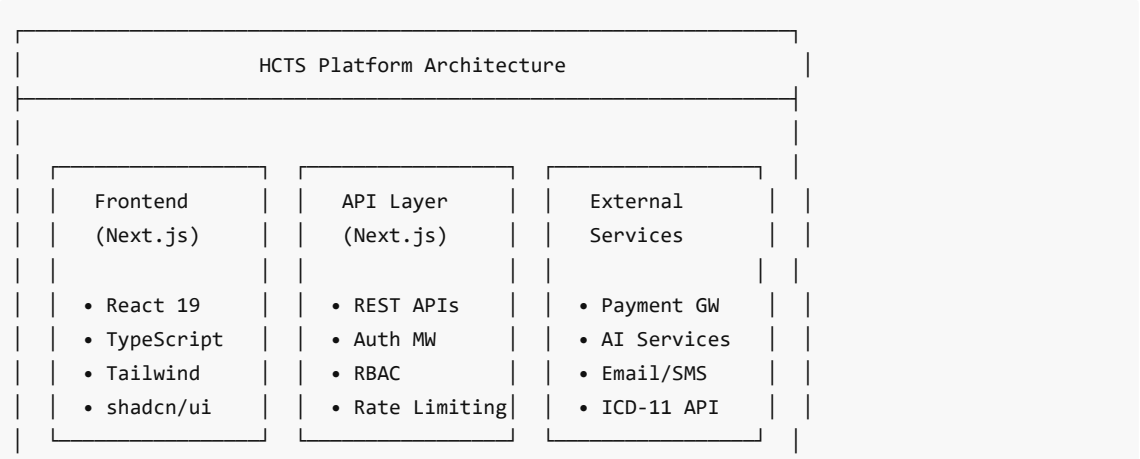
Key Features

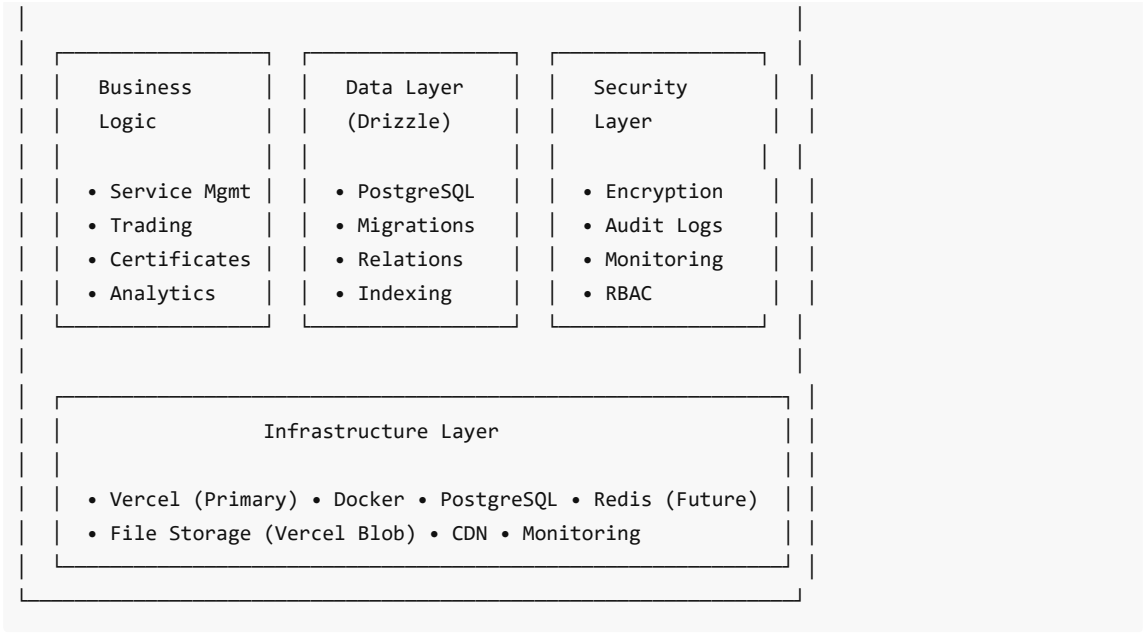
- **Multi-Role User System**: Support for providers, insurance companies, and intermediaries
- **Service Marketplace**: ICD-11 integrated service catalog with dynamic pricing
- **Certificate Management**: Digital certificate generation, verification, and blockchain-style validation
- **Payment Processing**: Multi-gateway payment system with commission management
- **Security & Compliance**: GDPR/HIPAA compliant with comprehensive audit logging
- **Real-time Analytics**: Trading metrics and performance dashboards

Technology Stack

- **Frontend**: Next.js 15, React 19, TypeScript, Tailwind CSS, shadcn/ui
- **Backend**: Next.js API Routes, PostgreSQL, Drizzle ORM
- **Authentication**: Better Auth with Google OAuth
- **AI Integration**: Vercel AI SDK with OpenAI
- **Security**: Custom security middleware, RBAC, encryption
- **Deployment**: Vercel-ready with Docker support

Architecture Diagram





Core Components

Frontend Layer

- **Framework**: Next.js 15 with App Router
- **UI Library**: shadcn/ui components with Radix UI primitives
- **Styling**: Tailwind CSS with custom design system
- **State Management**: React hooks and context
- **Forms**: React Hook Form with Zod validation

API Layer

- **Authentication**: Better Auth with session management
- **Authorization**: Role-Based Access Control (RBAC)
- **Rate Limiting**: Custom middleware with Redis support
- **Validation**: Zod schemas for request/response validation
- **Error Handling**: Centralized error boundary system

Business Logic Layer

- **Service Management**: CRUD operations for healthcare services
- **Trading Engine**: Transaction processing and commission calculation
- **Certificate System**: Digital certificate generation and verification
- **Payment Processing**: Multi-gateway integration with webhooks
- **Analytics Engine**: Real-time metrics and reporting

Data Layer

- **Database**: PostgreSQL with Drizzle ORM
- **Schema**: Comprehensive relational schema with indexes
- **Migrations**: Automated schema versioning
- **Caching**: Redis integration for performance optimization

Security Layer

- **Authentication**: Multi-factor authentication support

- **Authorization**: Granular permission system
- **Encryption**: Data encryption at rest and in transit
- **Audit Logging**: Comprehensive security event tracking
- **Compliance**: GDPR/HIPAA compliance features

Database Design

Core Tables

User Management

- `user` - Better Auth user table
- `users` - Extended user profile with roles
- `profiles` - Organization and contact information
- `userConsents` - GDPR consent tracking
- `dataProcessingRecords` - Data processing audit trail

Service Management

- `services` - Healthcare service catalog
- `serviceComponents` - Composite service relationships
- `icd11Categories` - ICD-11 classification system
- `tradingAnalytics` - Service performance metrics

Trading System

- `transactions` - Service purchase transactions
- `payments` - Payment processing records
- `cart` - Shopping cart functionality
- `certificates` - Digital certificate management

Security & Compliance

- `auditLogs` - System audit trail
- `securityEvents` - Security monitoring events
- `securityIncidents` - Incident tracking
- `rolesPermissions` - RBAC permission matrix
- `rateLimitRecords` - Rate limiting data

Key Relationships

```
users (1) ——— (N) profiles
users (1) ——— (N) services (provider)
users (1) ——— (N) transactions (buyer/seller)
services (1) ——— (N) transactions
transactions (1) ——— (1) payments
transactions (1) ——— (1) certificates
users (1) ——— (N) rolesPermissions (via role)
```

Database Features

- **Indexing Strategy**: Optimized indexes for performance
- **Foreign Key Constraints**: Data integrity enforcement
- **JSON Fields**: Flexible metadata storage
- **Audit Triggers**: Automatic audit logging

- **Partitioning**: Future scalability considerations

Security & Compliance

Authentication & Authorization

- **Multi-Factor Authentication**: Support for TOTP and hardware keys
- **Session Management**: Secure session handling with refresh tokens
- **Role-Based Access Control**: Granular permission system
- **API Key Management**: For system integrations

Data Protection

- **Encryption**: AES-256 encryption for sensitive data
- **Data Masking**: PII protection in logs and analytics
- **Secure File Storage**: Encrypted file uploads with access controls
- **Data Retention**: Configurable retention policies

Compliance Features

- **GDPR Compliance**: Consent management and data portability
- **HIPAA Compliance**: Healthcare data protection
- **Audit Logging**: Comprehensive activity tracking
- **Data Processing Records**: Legal compliance documentation

Security Monitoring

- **Real-time Alerts**: Suspicious activity detection
- **Intrusion Detection**: Pattern-based security monitoring
- **Rate Limiting**: DDoS protection and abuse prevention
- **Security Dashboards**: Real-time security metrics

API Structure

REST API Endpoints

Authentication (`/api/auth`)

- `POST /api/auth/[...all]` - Better Auth endpoints
- `GET /api/user/role` - Get current user role
- `POST /api/verification/status` - User verification status

Services (`/api/services`)

- `GET /api/services` - List available services
- `POST /api/services` - Create new service (providers)
- `GET /api/services/[id]` - Get service details
- `PUT /api/services/[id]` - Update service
- `GET /api/services/provider/[id]` - Provider's services
- `GET /api/services/stats` - Service statistics

Marketplace (`/api/marketplace`)

- `GET /api/marketplace/services` - Marketplace service listing
- `POST /api/purchase/initiate` - Initiate purchase
- `POST /api/purchase/complete` - Complete purchase

Payments (`/api/payments`)

- `POST /api/payments/initiate` - Initiate payment
- `GET /api/payments/status/[id]` - Payment status
- `GET /api/payments/history/[userId]` - Payment history
- `POST /api/payments/webhook` - Payment gateway webhooks

Certificates (`/api/certificates`)

- `POST /api/certificates/generate` - Generate certificate
- `GET /api/certificates/[id]` - Get certificate
- `GET /api/certificates/user/[userId]` - User's certificates
- `GET /api/certificates/verify/[qrCode]` - Verify certificate
- `GET /api/certificates/download/[id]` - Download certificate

Analytics (`/api/analytics`)

- `GET /api/analytics/[role]/[userId]` - Role-specific analytics
- `GET /api/platform-stats` - Platform-wide statistics

Security & Compliance

- `GET /api/audit/logs` - Audit log access
- `GET /api/security/status` - Security status
- `POST /api/security/report` - Security incident reporting
- `POST /api/gdpr/delete` - GDPR data deletion

API Features

- **OpenAPI Specification**: API documentation
- **Rate Limiting**: Per-endpoint rate limits
- **Request Validation**: Zod schema validation
- **Response Caching**: Redis-based caching
- **Webhook Support**: Real-time event notifications

User Roles & Permissions

Role Definitions

Admin

Permissions:

- Full system access
- User management
- Security monitoring
- System configuration
- Audit log access

Resources:

- `audit_logs` (read)
- `security_events` (read)
- `user_data` (read, write, delete)
- `system` (manage)

Provider

Permissions:

- Service management
- Transaction viewing
- Certificate access
- Profile management

Resources:

- `own_data` (read, write)
- `services` (read, write)
- `transactions` (read)
- `certificates` (read)

Insurance Company

Permissions:

- Service browsing
- Transaction management
- Certificate verification
- Analytics access

Resources:

- `own_data` (read, write)
- `services` (read)
- `transactions` (read, write)
- `certificates` (read)

Intermediary

Permissions:

- Marketplace access
- Transaction facilitation
- Commission management
- Client management

Resources:

- `own_data` (read, write)
- `services` (read)
- `marketplace` (read, write)
- `transactions` (read)

Permission Matrix

Resource	Admin	Provider	Insurance	Intermediary
audit_logs	✅	❌	❌	❌
security_events	✅	❌	❌	❌
user_data	✅	❌	❌	❌
system	✅	❌	❌	❌
own_data	✅	✅	✅	✅

services	✅	✅	✅	✅
transactions	✅	✅	✅	✅
certificates	✅	✅	✅	❌
marketplace	✅	❌	❌	✅

Deployment & Infrastructure

Primary Deployment (Vercel)

- **Frontend**: Vercel serverless functions
- **API Routes**: Vercel API routes
- **Database**: Vercel Postgres
- **File Storage**: Vercel Blob
- **CDN**: Vercel Edge Network

Docker Deployment

- **Containerization**: Full-stack Docker setup
- **Orchestration**: Docker Compose for local development
- **Production**: Kubernetes-ready configuration

Infrastructure Components

- **Load Balancing**: Vercel automatic scaling
- **Caching**: Redis for session and API caching
- **Monitoring**: Application performance monitoring
- **Backup**: Automated database backups
- **CDN**: Global content delivery

Environment Configuration

```
# Database
POSTGRES_URL=postgresql://...

# Authentication
BETTER_AUTH_SECRET=...
GOOGLE_CLIENT_ID=...
GOOGLE_CLIENT_SECRET=...

# AI Integration
OPENAI_API_KEY=...
OPENAI_MODEL=gpt-4

# Security
ENCRYPTION_KEY=...
JWT_SECRET=...

# External Services
PAYMENT_GATEWAY_KEY=...
EMAIL_SERVICE_KEY=...
```

Scaling Considerations

- **Horizontal Scaling**: Stateless API design
- **Database Scaling**: Read replicas and sharding
- **Caching Strategy**: Multi-layer caching (CDN, Redis, application)
- **Microservices**: Modular architecture for future decomposition

Monitoring & Observability

- **Application Metrics**: Response times, error rates, throughput
- **Database Metrics**: Query performance, connection pooling
- **Security Metrics**: Failed login attempts, suspicious activities
- **Business Metrics**: Transaction volume, user engagement

This architecture overview provides a comprehensive view of the HCTS Platform's design and implementation. The system is built with scalability, security, and compliance as primary concerns, supporting the complex requirements of healthcare service trading.