Patient Case Notes System Design Document

Summary

This document outlines the design for a national electronic health record system module that enables NHS doctors to create and manage patient case notes through both manual entry and automated transcription of scanned documents. The system is designed to be secure, scalable, GDPR compliant, and extensible for future analytics and integration.

Tech & System Design

A. System Architecture

Core Components and Services

1. **Frontend Application (React + TypeScript)**

- **Purpose**: NHS-style user interface for doctors to create, view, and manage case notes
- **Features**:
  - Prescription pad-style interface familiar to NHS doctors
  - Real-time form validation and auto-save
  - File upload interface for scanned documents
  - Responsive design for various devices

  b. **Backend API Service (Node.js + Express + TypeScript)**

  Purpose: RESTful API for case notes management and authentication

  Features:
  - JWT-based authentication with HTTP-only cookies
  - Role-based access control (doctors, admins, etc.)
  - File upload handling and validation
  - Database operations and data persistence
  - Audit logging for GDPR compliance

3. **OCR/AI Processing Service (Worker)**

- **Purpose**: Automated transcription of scanned case notes
- **Features**:
  - Tesseract.js for OCR processing

  - PDF text extraction
  - Image preprocessing and enhancement
  - Structured data extraction
  - Quality validation and confidence scoring

4. **Database Layer (PostgreSQL)**

- **Purpose**: Secure storage of patient data and case notes
- **Features**:
  - Encrypted at rest
  - Audit trails for all data modifications
  - Backup and disaster recovery
  - Data retention policies

5. **File Storage Service (AWS S3)**

- **Purpose**: Secure storage of scanned documents and images
- **Features**:
  - Encrypted storage
  - Access control and logging
  - Automatic lifecycle management
  - CDN integration for performance

6. **Authentication & Authorization Service**

- **Purpose**: NHS Identity integration and role management
- **Features**:
  - NHS Identity federation
  - Multi-factor authentication
  - Session management
  - Audit logging

Technology Stack

Component	Technology	Rationale
Frontend	React + TypeScript + Tailwind CSS	Modern, performant, NHS design system compatibility

Backend	Node.js + Express + TypeScript	Fast, Non blocking I/O, strong ecosystem, type safety
Database	PostgreSQL + Redis	ACID compliance, JSON support, caching
File Storage	AWS S3 + CloudFront	Scalable, secure, cost-effective
OCR Processing	Tesseract.js + Custom AI	Open-source, customizable, accurate
Infrastructure	AWS ECS + Fargate	Serverless, auto-scaling, managed
Monitoring	AWS CloudWatch + DataDog	Comprehensive observability
CI/CD	GitHub Actions + AWS CodePipeline	Automated, secure deployments

Data Flow

Manual Case Notes Flow:

1. Doctor logs in via NHS Identity or via application interface
2. Doctor creates new case note in web interface
3. Form data validated and auto-saved
4. Case note stored in PostgreSQL with audit trail
5. Real-time updates to patient record
6. Notifications sent to relevant healthcare staff

Scanned Case Notes Flow:

1. Doctor uploads scanned document (PDF/image)
2. File validated and stored in S3 with encryption
3. OCR worker processes document asynchronously
4. AI extracts structured data (patient ID, date, diagnosis, etc.)

5. Extracted text presented to doctor for review/editing
6. Final case note stored in database with original file reference
7. Quality metrics logged for continuous improvement

B. Scalability & Infrastructure

Scaling Strategy for NHS Hospitals

1. **Microservices Architecture**

- **Service Decomposition**: Separate services for authentication, case notes, file processing, and analytics
- **Independent Scaling**: Each service can scale based on demand
- **Load Balancing**: AWS Application Load Balancer for traffic distribution
- **Auto-scaling**: ECS Fargate with auto-scaling policies

2. **Database Scaling**

- **Read Replicas**: Multiple read replicas for query distribution
- **Sharding Strategy**: Geographic sharding by NHS region
- **Connection Pooling**: PgBouncer for connection management
- **Caching Layer**: Redis for frequently accessed data

3. **File Processing Pipeline**

- **Queue-based Processing**: AWS SQS for OCR job queuing
- **Worker Auto-scaling**: ECS Fargate workers scale based on queue depth
- **Batch Processing**: Process multiple documents in parallel
- **Priority Queuing**: Emergency cases processed first

4. **Performance Optimization**

- **CDN**: CloudFront for static assets and processed files
- **Database Indexing**: Optimized indexes for common queries
- **API Caching**: Redis cache for frequently accessed data
- **Compression**: Gzip compression for API responses

Infrastructure Components

1. **AWS Services**

- **Compute**: ECS Fargate for containerized services
- **Storage**: S3 for file storage, RDS for databases

- **Networking**: VPC, ALB, CloudFront CDN
- **Security**: AWS WAF, Shield, KMS for encryption
- **Monitoring**: CloudWatch, X-Ray for tracing

2. **CI/CD Pipeline**

- **Source Control**: GitHub with branch protection
- **Build**: GitHub Actions for automated testing
- **Deploy**: AWS CodePipeline for deployment automation
- **Testing**: Automated unit, integration, and security tests
- **Rollback**: Automated rollback capabilities

3. **Monitoring & Observability**

- **Application Monitoring**: DataDog APM
- **Infrastructure Monitoring**: CloudWatch dashboards
- **Log Management**: CloudWatch Logs with retention policies
- **Alerting**: PagerDuty integration for incident response
- **Performance**: Real User Monitoring (RUM)

Back-of-the-Envelope Capacity Planning

Assumptions

- **Users**: 1M monthly active users (20% daily active = 200k/day)
- **Notes/day**: ~1M total (50% manual, 50% scanned)
- **Scanned file size**: ~2.5 MB average PDF/image
- **Structured text size**: ~50 KB average
- **Peak multiplier**: ×10 over daily average during busy clinic hours
- **OCR speed (CPU)**: ~0.5 pages/sec per vCPU

Throughput estimates

- Notes per day: **1M** → average ~12 notes/sec → peak ~120 notes/sec
- API requests per second (create, autosave, review, finalize, read): average ~230 RPS, peak ~2,400 RPS
- OCR workload: ~60 scanned documents/sec at peak → ~240 vCPUs needed for synchronous turnaround (can be reduced with batching)

Storage & bandwidth (daily)

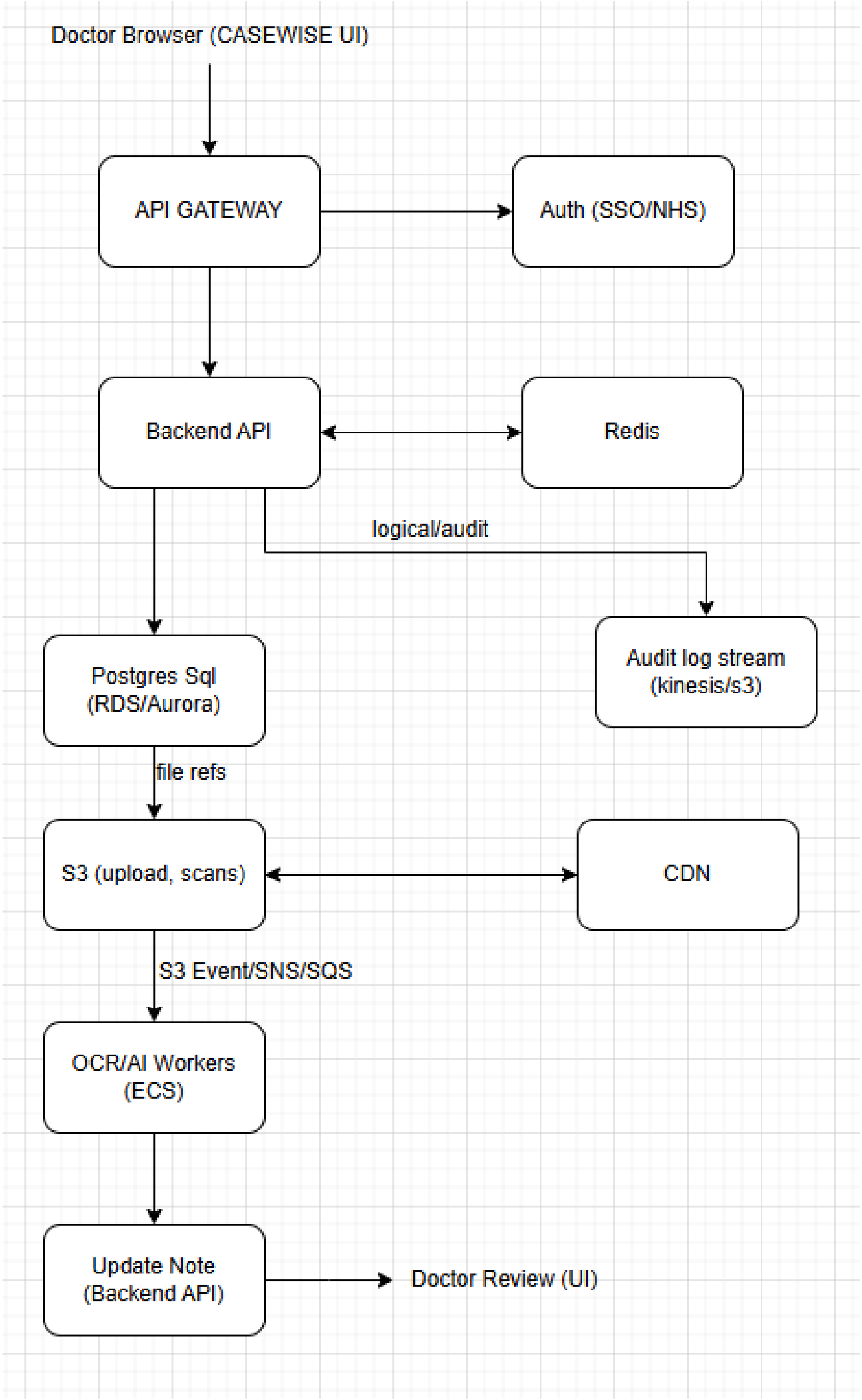
- Scanned files: **1.25 TB/day** (~38 TB/month)
- Structured text: **50 GB/day** (~1.5 TB/month)
- Egress (viewing scanned files): ~375 GB/day (~11 TB/month, mostly CDN-cacheable)

Database load (PostgreSQL)

- Writes: ~360–480 writes/sec at peak (notes, audit, status)
- Reads: 600–2,400 reads/sec at peak (list/search/review)
- Fits well within Aurora PostgreSQL or tuned RDS PostgreSQL with read replicas and PgBouncer/RDS Proxy

---

Component Interaction Diagram



Scalability & Infrastructure

Sizing for 1M users (Peak load)

- **API layer**: Approximately 24–30 ECS Fargate tasks (0.5–1 vCPU each) to support around 2,400 RPS peak with headroom.

- **OCR workers**: Approximately 60–120 ECS tasks (2–4 vCPU each) to handle peak OCR demand; can burst to around 240 vCPU if synchronous turnaround is required.
- **Database**: Aurora Postgres or RDS Postgres Multi-AZ with at least three read replicas, using PgBouncer or RDS Proxy.
- **Cache layer**: 3-node Redis Multi-AZ cluster.
- **Queue processing**: SQS with depth-based autoscaling for workers.

Deployment, Replication & Consensus Strategy

Control plane / Service orchestration

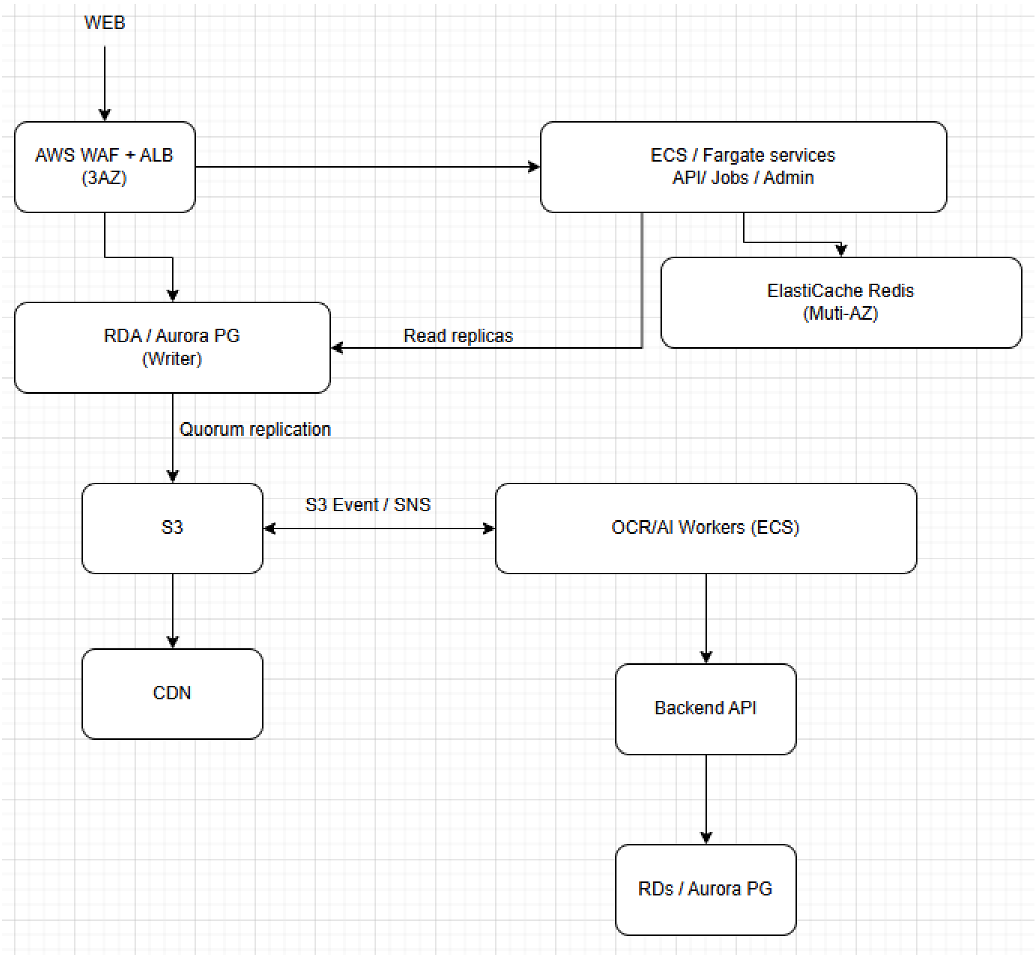
- ECS on Fargate across three AZs
- Blue/Green deployments via CodeDeploy
- ALB with target groups for each service

Data plane

- **PostgreSQL**:
  - Aurora Postgres: quorum writes across six copies (Paxos-like) in three AZs
  - RDS Postgres: Multi-AZ with streaming replication (synchronous for RPO=0, asynchronous to read replicas)
- **Redis**: ElastiCache Multi-AZ with automatic failover
- **S3**: Multi-AZ durability, SSE-KMS encryption
- **SQS**: Managed durability; FIFO queues for ordered processing when needed

---

Deployment / Replication Diagram



Security & GDPR Compliance

1. **Data Protection**

- **Encryption**: AES-256 encryption at rest and in transit
- **Access Control**: Role-based access with least privilege
- **Audit Logging**: Comprehensive audit trails for all data access
- **Data Minimization**: Only collect necessary patient data
- **Retention Policies**: Automated data deletion after retention period

2. **NHS Security Standards**

- **NHS Digital Security**: Compliance with NHS Digital security standards
- **Data Residency**: All data stored within UK data centers
- **Network Security**: Private VPC with security groups
- **Vulnerability Management**: Regular security scans and updates
- **Incident Response**: 24/7 security monitoring and response

3. **GDPR Compliance**

- **Data Subject Rights**: Automated tools for data access/deletion requests
- **Consent Management**: Granular consent tracking
- **Data Processing Records**: Detailed records of all data processing
- **Breach Notification**: Automated breach detection and notification
- **Privacy by Design**: Privacy considerations built into system design

Scalability Metrics

Target Capacity:

- **Users**: 150,000+ NHS doctors across UK
- **Case Notes**: 5+ million notes per month
- **File Uploads**: 2+ million scanned documents per month
- **Response Time**: <200ms for API calls, <2s for page loads
- **Availability**: 99.9% uptime SLA

Scaling Triggers:

- **CPU Utilization**: Scale up at 70%, down at 30%
- **Memory Usage**: Scale up at 80%, down at 40%
- **Queue Depth**: Scale workers when queue >100 items
- **Response Time**: Scale when p95 >500ms

Conclusion

This design provides a robust, scalable, and secure foundation for the NHS patient case notes system. The microservices architecture ensures flexibility and maintainability, while the comprehensive security measures ensure GDPR compliance and NHS security standards. The system is designed to grow with the NHS's needs while maintaining performance and reliability.