Mediconnect Project Analysis & Software Development Plan

Document Analysis Summary

I have systematically analyzed 11 comprehensive documents totaling over 400 pages of requirements, technical specifications, and project context for the Mediconnect healthcare platform.

Documents Analyzed

Document	Type	Key Content	Priority
Mediconnect_Design_Spec.md	Technical Spec	Patient experience prototype, 9 screen flows	Primary
Mediconnect_MVP_Formatted_Report.pdf	Requirements	Complete MVP with RBAC, 48 screens, API specs	Primary
Medi connect starter documentation.docx	Business Spec	HaaS operating model, care pathways, data models	Primary
pdr1.pdf	Technical PRD	Detailed wireframes, user journeys, risk analysis	Primary
pdr appendum.pdf	Technical Addendum	Complete database schema, RLS policies, OpenAPI specs	Primary
doc addon.pdf	Requirements	Technical requirements, MoSCoW prioritization	Primary
conversation.pdf	Context	Previous analysis of 35 documents, 4-tier categorization	Secondary
conversation (1).pdf	Guidelines	Wireframe creation methodology, RBAC mapping	Secondary
conversation (2-3).pdf	Status	Project status, locked decisions, completed artifacts	Secondary

Key Conflicts, Gaps, and Risks Identified

⚠️ Critical Conflicts

1. **Timeline Discrepancies**
   - Some documents: “end of year” completion
   - MVP Report: 3-month development timeline
   - **Resolution Needed**: Confirm official timeline

2. **Payment Integration Scope**
   - MVPs state: “no in-app payments in MVP”
   - Business docs discuss: payment models and revenue
   - **Resolution Needed**: Clarify payment scope for V1

3. **Specialist Rollout Timing**
   - Some docs: specialists in Phase 1 (MVP)
   - Others: specialists in Phase 2 (post-MVP)
   - **Resolution Needed**: Confirm specialist inclusion in MVP

🔍 Critical Gaps

1. **Missing Option-1 Code Bundle**: Referenced in requirements but not provided
2. **Current State Assessment**: No clear documentation of existing vs. needed components
3. **Team Structure**: Limited information about development team composition
4. **Detailed Project Timeline**: High-level 3-month mention without sprint breakdowns
5. **Testing Strategy**: Minimal QA and testing approach documentation

🚨 High-Priority Risks

1. **WhatsApp Single Point of Failure**: Critical dependency for auth and video calls
2. **Aggressive 3-Month Timeline**: Complex healthcare platform with strict regulations
3. **Budget Constraints**: $300-500/month for 10k users may be insufficient
4. **Regulatory Compliance**: Kenya healthcare regulations not fully defined
5. **Partner Network Capacity**: Risk of insufficient pharmacy/specialist availability

---

SOFTWARE DEVELOPMENT PLAN

Mediconnect Healthcare Platform V1

Executive Summary, Technical Architecture, and Delivery Strategy

---

A) Executive Summary

Product Vision

Mediconnect is a mobile-first, Healthcare-as-a-Service (HaaS) platform designed to democratize quality healthcare access across Kenya. Unlike traditional marketplace models, Mediconnect functions as a comprehensive service provider where General Practitioners orchestrate the entire patient journey from initial triage through specialist care, prescription fulfillment, and diagnostic services.

Target Users & Problem Statement

Primary Users: Urban and peri-urban Nairobi residents (18-45 years) with smartphone access and WhatsApp familiarity who face fragmented healthcare systems, long wait times, unsafe medication dispensing, and limited specialist access.

Core Problem: Healthcare access in Kenya suffers from fragmented provider networks, weak care co-ordination, counterfeit medication risks, and limited specialist availability, leading to delayed care, inappropriate treatments, and poor health outcomes.

Value Hypothesis & Success Metrics

V1 Value Hypothesis: By providing GP-orchestrated care through familiar WhatsApp channels combined with a secure mobile app, we can deliver trusted, accessible healthcare that reduces patient cognitive load while ensuring clinical safety and care continuity.

Quantitative Success Metrics:
- **User Growth**: 500 → 1,000 → 10,000 MAU over 3 months
- **Technical Performance**: 99.5% uptime, <500ms P75 API latency
- **Clinical Quality**: 85%+ consultation completion, 15±2 min GP sessions
- **Partner Network**: 20+ verified pharmacies, 15+ specialists
- **User Engagement**: 70% 30-day retention, 4.2/5 satisfaction rating

Qualitative Success Metrics:
- Seamless WhatsApp integration reduces onboarding friction
- GP-curated referrals increase patient trust and outcomes
- Verified pharmacy network reduces counterfeit medication concerns
- AI intake reduces GP discovery time and improves care quality

Out of Scope (V1)

- Native in-app video calling (WhatsApp-only for MVP)
- SMS/email authentication methods
- Patient-facing provider marketplaces
- Direct specialist booking by patients
- Audio/video consultation recording
- Home delivery logistics
- Insurance provider integration
- In-app payment processing
- Multi-region deployment beyond Kenya

B) Requirements & Scope

Functional Requirements (User Stories with Acceptance Criteria)

Epic 1: Patient Onboarding & Authentication

As a patient, I want to easily create an account and verify my identity so that I can access healthcare services securely.

User Stories:
1. **Language Selection**
- As a patient, I want to select my preferred language (English/Swahili) so that I can use the app in my native language

- **AC**: Language persists throughout the app experience
- **AC**: All patient-facing text displays in selected language

1. **WhatsApp OTP Authentication**
   - As a patient, I want to verify my phone number via WhatsApp so that I can access the platform securely
   - **AC**: OTP delivered via WhatsApp template within 5 seconds (95% success rate)
   - **AC**: 6-digit code with 5-minute TTL and 5 attempt limit
   - **AC**: Failed attempts trigger progressive lockout with clear retry guidance

2. **Profile Setup**
   - As a patient, I want to complete my basic profile so that GPs have necessary context for care
   - **AC**: Required fields: full name, date of birth, gender, emergency contact
   - **AC**: Profile validation prevents users under 18 from proceeding
   - **AC**: Profile completion creates secure 30-day session

Epic 2: AI-Powered Intake & GP Consultation

As a patient, I want an intelligent intake process and seamless GP consultation so that I receive appropriate care efficiently.

User Stories:
1. **AI Intake Questionnaire**
- As a patient, I want to complete a brief health assessment so that the GP understands my condition before the consultation
- **AC**: Adaptive questionnaire completes in ≤90 seconds (P50)
- **AC**: AI generates structured summary visible to both patient and GP
- **AC**: Red-flag conditions trigger priority routing to available GP

1. **GP Video Consultation**
   - As a patient, I want to connect with a qualified GP via WhatsApp video so that I can receive professional medical advice
   - **AC**: WhatsApp call link delivered within 5 seconds (95% success rate)
   - **AC**: 15-minute standard session with 12-minute warning prompt
   - **AC**: In-app chat thread remains active for notes and follow-up

2. **Consultation Extension for Referrals**
   - As a patient, I want additional time with my GP if specialist referral is needed so that I can make informed decisions
   - **AC**: Session automatically extends to 25 minutes when referral accepted
   - **AC**: GP presents 3-4 curated specialist options (local/online)
   - **AC**: Patient selection triggers automated booking coordination

Epic 3: Prescription & Pharmacy Management

As a patient, I want secure digital prescriptions and easy pharmacy access so that I can obtain authentic medications safely.

User Stories:
1. **Digital Prescription Access**
- As a patient, I want to view my prescription with secure verification options so that I can fulfill it safely
- **AC**: Prescription visible immediately after GP consultation

- **AC**: QR code and PDF download options both available
- **AC**: Clear warning that PDF download permanently disables QR

1. **Verified Pharmacy Discovery**
   - As a patient, I want to find nearby verified pharmacies so that I can obtain authentic medications
   - **AC**: Map displays verified pharmacies within 5km radius
   - **AC**: Top 3 nearest pharmacies highlighted with distance/hours
   - **AC**: Pharmacy selection provides directions and contact options

2. **Prescription Fulfillment Tracking**
   - As a patient, I want to track my prescription status so that I know when it's ready for pickup
   - **AC**: Real-time status updates when pharmacy processes prescription
   - **AC**: Notification sent when prescription ready for collection
   - **AC**: Prescription history accessible for 90 days

Epic 4: Specialist Referral & Diagnostics

As a patient, I want coordinated specialist care and diagnostic services so that I receive comprehensive treatment.

User Stories:
1. **Specialist Appointment Coordination**
- As a patient, I want seamless specialist booking after GP referral so that I can continue my care journey
- **AC**: Appointment confirmed within 4 hours during business hours
- **AC**: 24-hour reminder sent via WhatsApp and in-app
- **AC**: Specialist receives GP notes and intake summary with consent

1. **Diagnostic Test Management**
   - As a patient, I want to easily complete ordered diagnostic tests so that my specialist can make informed decisions
   - **AC**: Test orders transmitted to labs with minimal patient data exposure
   - **AC**: Lab booking instructions provided with preparation guidance
   - **AC**: Results available to both patient and ordering specialist simultaneously

Non-Functional Requirements

Performance SLOs

- **API Response Time**: P75 < 500ms for critical endpoints
- **Chat Load Time**: Last 50 messages load in <1.5 seconds
- **WhatsApp Integration**: 99% successful OTP delivery
- **Concurrent Users**: Support 60+ simultaneous chat sessions

Availability/RTO/RPO

- **Platform Availability**: 99.5% uptime during pilot period
- **Recovery Point Objective (RPO)**: ≤1 hour data loss maximum
- **Recovery Time Objective (RTO)**: ≤4 hours service restoration
- **Backup Strategy**: Automated daily backups with point-in-time recovery

Security/Privacy

- **Data Encryption**: TLS 1.2+ in transit, AES-256 at rest
- **Access Control**: Row-Level Security policies enforcing least privilege
- **PII Minimization**: Partner portals see only necessary data fields

- **Audit Logging**: Comprehensive event tracking for all data access/modifications
- **Session Management**: Secure JWT tokens with configurable expiration

Compliance

- **Healthcare Regulations**: Kenya Medical Board compliance requirements
- **Data Protection**: GDPR-style controls for patient data rights
- **Professional Standards**: GP/Specialist licensing verification
- **Pharmacy Verification**: Licensed partner network only

Accessibility

- **WCAG 2.2 AA**: Minimum 4.5:1 color contrast, proper focus management
- **Mobile Optimization**: Android-first responsive design
- **Language Support**: Complete English/Swahili localization
- **Low-Bandwidth**: Graceful degradation for poor connections

Localization

- **Patient Interfaces**: Full English and Swahili support
- **Cultural Adaptation**: Kenya-specific healthcare terminology
- **Legal Documents**: Terms of Service and Privacy Policy in English
- **Future Expansion**: Architecture supports additional languages

Observability

- **Application Metrics**: Real-time dashboards for user engagement and system health
- **Error Tracking**: Automated alerting for application exceptions and API failures
- **Performance Monitoring**: Latency and throughput metrics for critical user journeys
- **Business Intelligence**: Analytics pipeline for clinical outcomes and operational efficiency

Assumptions, Constraints, and Won’t Do List

Assumptions

1. **WhatsApp Ubiquity**: Target users have WhatsApp installed and are comfortable with video calls
2. **Android Prevalence**: Majority of target users have Android devices
3. **Internet Connectivity**: Users have reliable internet for video consultations
4. **Healthcare Regulations**: Current Kenya telehealth regulations allow proposed model
5. **Partner Willingness**: Pharmacies and labs will join verified network

Constraints

- **Timeline**: 3-month development window for MVP launch
- **Budget**: $300-500 monthly operational budget constraint
- **Geographic Scope**: Nairobi pilot only, no multi-region complexity
- **Team Size**: Limited development team requiring careful scope management
- **Regulatory**: Must comply with Kenya Medical Board requirements

Explicit Won’t Do List

1. **Native Video Calling**: All video via WhatsApp, no in-app video for MVP
2. **Alternative Authentication**: WhatsApp OTP only, no SMS/email options
3. **Patient Provider Marketplace**: No browsing of doctors/pharmacies
4. **Self-Service Specialist Booking**: GP-gated referrals only
5. **Consultation Recording**: No audio/video capture or storage
6. **Home Delivery**: No medication logistics or delivery management

7. **Insurance Integration**: No billing or claims processing
8. **In-App Payments**: No payment processing within the application
9. **AI Diagnosis**: AI for triage/summary only, no autonomous prescriptions

Open Questions (≤10 with Resolution Owners)

Priority	Question	Owner	Due Date	Impact
High	Which Health-Tech Wireframe version is most current among 4 versions?	Product Lead	Week 1	Design implementation
High	Confirm official timeline: 3-month development vs “end of year” references	Project Manager	Week 1	Resource planning
High	Clarify payment scope: MVP excludes payments but business docs discuss revenue	Business Lead	Week 1	Feature scope
High	Specialist rollout: Phase 1 (MVP) vs Phase 2 inclusion?	Clinical Lead	Week 1	Development priority
Medium	USSD/SMS fallback: TBD in specs but implementation details exist elsewhere	Technical Lead	Week 2	Architecture decisions
Medium	Kenya Medical Board specific compliance requirements	Legal/Regulatory	Week 2	Compliance architecture
Medium	Pharmacy partner capacity and onboarding timeline	Operations Lead	Week 2	Launch readiness
Medium	Team structure and development resource allocation	Engineering Manager	Week 1	Sprint planning

Priority	Question	Owner	Due Date	Impact
Low	Backup WhatsApp provider in case of API issues	Technical Lead	Week 3	Risk mitigation
Low	Future feature prioritization beyond MVP scope	Product Lead	Month 2	Roadmap planning

C) Tech Stack Decision Record

Current Tech Stack Analysis (Option-1)

Based on comprehensive document analysis, the following tech stack has been selected and documented across multiple PRDs:

Frontend Components

- **Mobile App**: React Native with Expo framework
- **Web Portals**: Next.js for GP/Specialist/Pharmacy/Diagnostics/Ops interfaces
- **State Management**: React Context API + Zustand for complex state
- **Styling**: Inline styles (per design spec) with responsive design tokens

Backend Infrastructure

- **Runtime**: Node.js with TypeScript for type safety
- **Framework**: Express.js with helmet for security
- **API Gateway**: Managed API Gateway for request routing and rate limiting
- **Authentication**: JWT tokens with Supabase Auth integration

Database & Storage

- **Primary Database**: PostgreSQL with Supabase managed service
- **Security**: Row-Level Security (RLS) policies for access control
- **Real-time**: Supabase Realtime for chat functionality
- **Archival Storage**: Cloudflare R2 (S3-compatible, zero egress fees)
- **Media Storage**: Encrypted file storage for documents and images

External Integrations

- **Communication**: WhatsApp Cloud API for OTP and video calls
- **Authentication**: Firebase Phone Auth as backup to WhatsApp
- **Notifications**: Push notification service (FCM for Android)
- **Maps**: Google Maps API for pharmacy discovery
- **Monitoring**: Sentry for error tracking, DataDog for metrics

Alternative Stack Comparison

Component	Option 1 (Selected)	Option 2	Option 3	Decision Rationale
Mobile Framework	React Native + Expo	Flutter	Native Android	Rapid development, shared codebase, mature ecosystem
Backend	Node.js + Express	Python + FastAPI	Go + Gin	Team expertise, ecosystem maturity, rapid prototyping
Database	PostgreSQL + Supabase	MongoDB + Atlas	MySQL + PlanetScale	ACID compliance, RLS support, healthcare data requirements
Authentication	WhatsApp OTP + Firebase	Auth0	AWS Cognito	Local market WhatsApp penetration, cost efficiency
Storage	Cloudflare R2	AWS S3	Google Cloud Storage	Zero egress fees critical for budget constraints
Real-time	Supabase Realtime	Socket.io	WebSocket	Integrated solution, reduced complexity

Tech Stack Fitness Assessment

Pros of Selected Stack

✅ **Rapid Development**: React Native + Supabase enables fast MVP delivery
✅ **Cost Efficient**: Supabase free tier + R2 zero egress fits budget constraints
✅ **Security Built-in**: RLS policies enforce healthcare data protection
✅ **WhatsApp Integration**: Leverages ubiquitous communication platform
✅ **Scalability**: Supabase can handle 10k MAU target
✅ **Team Familiarity**: JavaScript/TypeScript ecosystem widely known
✅ **Healthcare Compliance**: PostgreSQL ACID properties support audit requirements

Cons/Limitations

❌ **Vendor Lock-in**: Heavy Supabase dependency
❌ **WhatsApp Risk**: Single point of failure for critical flows

❌ **React Native Performance**: Potential issues with complex healthcare UIs
❌ **Limited Customization**: Supabase Auth may not meet all healthcare requirements
❌ **African Infrastructure**: Supabase edge regions may not optimize for Kenya

Team Fit & Hiring Market

- **Development Team**: Strong JavaScript/React ecosystem in Nairobi
- **DevOps Requirements**: Minimal due to managed services approach
- **Learning Curve**: Low for developers familiar with React ecosystem
- **Market Availability**: High availability of React Native developers in Kenya

Cost Analysis (Monthly)

Service	Cost Range	Justification
Supabase Pro	$25	Database, Auth, Real-time, Storage up to 8GB
Cloudflare R2	$15	100GB storage, zero egress fees
WhatsApp Cloud API	$50-100	Based on message volume
Google Maps API	$50-100	Pharmacy location services
Hosting/CDN	$25-50	Next.js deployment, static assets
Monitoring	$25-50	Error tracking, performance monitoring
Total Estimated	$190-340/month	Well within $300-500 budget constraint

Maturity Assessment

- **React Native**: Mature, battle-tested by Meta, Shopify, Microsoft
- **Supabase**: Growing rapidly, backed by Y Combinator, strong community
- **WhatsApp Business API**: Enterprise-ready, used by banks and healthcare
- **PostgreSQL**: 25+ years mature, healthcare industry standard
- **TypeScript**: Industry standard for large JavaScript applications

Recommended Stack for V1

Final Recommendation: Proceed with Option-1 (Current Selection)

Key Decision Factors:
1. **Time-to-Market**: React Native + Supabase enables 3-month MVP timeline
2. **Budget Alignment**: Total cost estimate fits within constraints
3. **Healthcare Requirements**: PostgreSQL RLS meets data protection needs
4. **Local Market Fit**: WhatsApp integration leverages user behaviors
5. **Team Capability**: Strong ecosystem support for skill development

Migration Implications

Future Migration Paths:
- **Database**: PostgreSQL schema portable to AWS RDS, Google Cloud SQL
- **Authentication**: Can migrate from Supabase Auth to Auth0 or custom solution
- **Storage**: R2 S3 compatibility enables easy migration to AWS/GCP
- **Mobile**: React Native allows future iOS development without rewrite

De-risking POC Checklist

Before full development, validate these critical assumptions:

- [ ] **WhatsApp Cloud API**: Verify OTP delivery in Kenya
- [ ] **Supabase Performance**: Load test with 100+ concurrent users
- [ ] **RLS Policies**: Validate complex healthcare access patterns
- [ ] **React Native Healthcare UI**: Test complex forms and chat interfaces
- [ ] **Offline Capability**: Verify app behavior with poor connectivity
- [ ] **Kenya Compliance**: Confirm data residency and regulation alignment
- [ ] **Integration Testing**: WhatsApp + Firebase auth fallback scenarios
- [ ] **Mobile Performance**: Test on low-end Android devices common in Kenya
- [ ] **Real-time Chat**: Stress test Supabase Realtime with medical conversations
- [ ] **File Handling**: Test prescription PDF generation and QR code functionality

---

D) Architecture & Data

Context Diagram

```
graph TB
    Patient[Patient - Mobile App]
    GP[GP - Web Portal]
    Specialist[Specialist - Web Portal]
    Pharmacy[Pharmacy - Web Portal]
    Diagnostics[Diagnostics - Web Portal]
    Ops[Ops Admin - Web Portal]
    Support[Support - Web Portal]

    subgraph "Mediconnect Platform"
        API[API Gateway]
        Backend[Backend Services]
        DB[(PostgreSQL + RLS)]
        Storage[File Storage - R2]
        Realtime[Real-time Engine]
        Auth[Authentication Service]
    end

    subgraph "External Services"
        WhatsApp[WhatsApp Cloud API]
        Maps[Google Maps API]
        Push[Push Notifications]
        Monitor[Monitoring & Alerts]
    end

    Patient --> API
    GP --> API
    Specialist --> API
    Pharmacy --> API
    Diagnostics --> API
    Ops --> API
    Support --> API

    API --> Backend
    Backend --> DB
    Backend --> Storage
    Backend --> Realtime
    Backend --> Auth

    Backend --> WhatsApp
    Backend --> Maps
    Backend --> Push
    Backend --> Monitor

    WhatsApp -.->|OTP & Video| Patient
    Push -.->|Notifications| Patient
    Maps -.->|Pharmacy Locations| Patient
```

High-Level Component Diagram

```
graph TB
    subgraph "Client Layer"
        MobileApp[Mobile App - React Native]
        WebPortals[Web Portals - Next.js]
    end

    subgraph "API Layer"
        Gateway[API Gateway]
        RateLimit[Rate Limiting]
        AuthMiddleware[Auth Middleware]
    end

    subgraph "Business Logic Layer"
        UserService[User Management Service]
        ConsultService[Consultation Service]
        PrescriptionService[Prescription Service]
        ReferralService[Referral Service]
        NotificationService[Notification Service]
        AIService[AI Intake Service]
        AuditService[Audit Service]
    end

    subgraph "Data Layer"
        PostgresDB[(PostgreSQL)]
        RLS[Row Level Security]
        RealtimeDB[Supabase Realtime]
        FileStorage[Cloudflare R2]
        CacheLayer[Redis Cache]
    end

    subgraph "External Integrations"
        WhatsAppAPI[WhatsApp Cloud API]
        MapsAPI[Google Maps API]
        PushService[FCM Push Service]
        MonitoringService[Sentry + DataDog]
    end

    MobileApp --> Gateway
    WebPortals --> Gateway
    Gateway --> RateLimit
    RateLimit --> AuthMiddleware
    AuthMiddleware --> UserService
    AuthMiddleware --> ConsultService
    AuthMiddleware --> PrescriptionService
    AuthMiddleware --> ReferralService
    AuthMiddleware --> NotificationService

    UserService --> PostgresDB
    ConsultService --> PostgresDB
    ConsultService --> RealtimeDB
    PrescriptionService --> PostgresDB
    ReferralService --> PostgresDB
    NotificationService --> WhatsAppAPI
    NotificationService --> PushService
    AIService --> ConsultService
    AuditService --> PostgresDB

    PostgresDB --> RLS
    ConsultService --> FileStorage
    PrescriptionService --> FileStorage

    UserService --> CacheLayer
    ConsultService --> MapsAPI

    Gateway --> MonitoringService
    UserService --> MonitoringService
    ConsultService --> MonitoringService
```

Sequence Diagrams for Critical Flows

1. Patient Authentication Flow

```
sequenceDiagram
    participant P as Patient Mobile App
    participant API as API Gateway
    participant Auth as Auth Service
    participant WA as WhatsApp API
    participant DB as PostgreSQL
    participant Audit as Audit Service

    P->>API: POST /auth/whatsapp/otp {phone}
    API->>Auth: Generate OTP request
    Auth->>DB: Store OTP hash with TTL
    Auth->>WA: Send OTP template message
    WA-->>P: WhatsApp OTP message
    Auth->>Audit: Log OTP sent event
    Auth->>API: Response: OTP queued
    API->>P: 202 Accepted

    P->>API: POST /auth/whatsapp/verify {phone, code}
    API->>Auth: Verify OTP
    Auth->>DB: Check OTP hash & attempts
    Auth->>DB: Create/update user session
    Auth->>Audit: Log verification success
    Auth->>API: JWT token + user data
    API->>P: 200 OK {token, user}
```

2. GP Consultation with Specialist Referral

```
sequenceDiagram
    participant P as Patient
    participant API as API Gateway
    participant CS as Consultation Service
    participant AI as AI Service
    participant WA as WhatsApp API
    participant RS as Referral Service
    participant NS as Notification Service
    participant DB as PostgreSQL

    P->>API: Start consultation request
    API->>CS: Create consultation
    CS->>AI: Generate intake questions
    AI->>P: Adaptive questionnaire
    P->>AI: Submit intake responses
    AI->>CS: Generate AI summary
    CS->>DB: Store consultation + AI summary

    CS->>CS: Assign available GP
    CS->>WA: Initiate WhatsApp video call
    WA-->>P: Video call invitation
    Note over P,WA: 15-minute consultation

    alt Referral Required
        CS->>CS: Extend session to 25 minutes
        CS->>RS: Present specialist options
        P->>RS: Select specialist
        RS->>DB: Create referral record
        RS->>NS: Notify Ops for booking
        NS->>API: Appointment coordination
    end

    CS->>DB: Complete consultation
    CS->>NS: Send consultation summary
    NS->>P: Consultation complete notification
```

3. Prescription Fulfillment with QR Security

```
sequenceDiagram
    participant P as Patient
    participant API as API Gateway
    participant PS as Prescription Service
    participant PA as Pharmacy Admin
    participant PharmacyAPI as Pharmacy Portal
    participant DB as PostgreSQL
    participant Storage as File Storage
    participant Audit as Audit Service

    PS->>DB: Create prescription record
    PS->>PS: Generate secure QR code
    PS->>Storage: Generate PDF backup
    PS->>DB: Store prescription with QR enabled
    P->>API: View prescription
    API->>P: Display QR + PDF option

    alt QR Code Path
        P->>Pharmacy: Show QR code
        PA->>PharmacyAPI: Scan QR code
        PharmacyAPI->>PS: Verify QR code
        PS->>DB: Validate QR + create claim
        PS->>PharmacyAPI: Return item-only view
        PA->>PharmacyAPI: Confirm dispensing
        PharmacyAPI->>PS: Update claim status
        PS->>Audit: Log dispensing event
    else PDF Download Path
        P->>API: Download PDF
        API->>PS: Generate PDF
        PS->>DB: Set QR enabled = false
        PS->>Storage: Retrieve PDF
        PS->>Audit: Log PDF download + QR disable
        API->>P: Return PDF file
        Note over P: QR permanently disabled
    end
```

API Surface (OpenAPI Outline)

Authentication Endpoints

```
/auth/whatsapp/otp:
  post:
    summary: Send WhatsApp OTP
    parameters: {phone_e164}
    responses: {202: OTP queued, 429: Rate limited}

/auth/whatsapp/verify:
  post:
    summary: Verify OTP and create session
    parameters: {phone_e164, code}
    responses: {200: JWT token + user, 401: Invalid code}
```

Consultation Endpoints

```
/consultations:
  post:
    summary: Create new consultation
    security: [bearerAuth]
    parameters: {patient_id, chief_complaint}
    responses: {201: Consultation created}

/consultations/{id}:
  get:
    summary: Get consultation details
    security: [bearerAuth]
    responses: {200: Consultation data}
  patch:
    summary: Update consultation status
    security: [bearerAuth]
    parameters: {status, specialist_id, ended_at}
```

AI Intake Endpoints

```
/intake:
  post:
    summary: Submit AI intake responses
    security: [bearerAuth]
    parameters: {consultation_id, responses}
    responses: {201: Intake created with AI summary}

/intake/{id}:
  get:
    summary: Get intake summary
    security: [bearerAuth]
    responses: {200: AI summary + responses}
```

Prescription Endpoints

```
/prescriptions:
  post:
    summary: Issue prescription
    security: [bearerAuth]
    parameters: {consultation_id, patient_id, items}
    responses: {201: Prescription with QR + PDF options}

/prescriptions/{id}:
  get:
    summary: Get prescription details
    security: [bearerAuth]
    responses: {200: Prescription data}
```

Pharmacy Integration Endpoints

```
/pharmacy/claims/verify-qr:
  post:
    summary: Verify QR code for prescription
    security: [bearerAuth]
    parameters: {qr_payload, org_id}
    responses: {200: Valid QR with item-only view, 409: QR disabled}

/pharmacy/claims:
  post:
    summary: Submit pharmacy claim
    security: [bearerAuth]
    parameters: {prescription_id, org_id, items_filled}
    responses: {201: Claim created}
```

Data Model (ERD)

Core Entities & Relationships

```
erDiagram
    users {
        uuid id PK
        text phone_e164 UK
        text role
        uuid org_id FK
        text full_name
        date dob
        text gender
        timestamptz created_at
        timestamptz deleted_at
    }

    organizations {
        uuid id PK
        text type
        text name
        text country_code
        text address
        text phone
    }

    consultations {
        uuid id PK
        text status
        uuid patient_id FK
        uuid gp_id FK
        uuid specialist_id FK
        text chief_complaint
        timestamptz started_at
        timestamptz ended_at
        int capacity_cap_minutes
        boolean referral_extension_applied
    }

    ai_intake {
        uuid id PK
        uuid consultation_id FK
        jsonb summary_json
        boolean red_flags
        timestamptz created_at
    }

    messages {
        uuid id PK
        uuid consultation_id FK
        uuid sender_id FK
        text type
        text body_text
        text media_url
        timestamptz created_at
    }

    referrals {
        uuid id PK
        uuid consultation_id FK
        uuid from_gp_id FK
        uuid to_specialist_id FK
        uuid to_org_id FK
        text status
        text reason
        jsonb options_json
        timestamptz created_at
    }

    prescriptions {
        uuid id PK
        uuid consultation_id FK
        uuid patient_id FK
        uuid prescriber_id FK
        text status
        jsonb items_json
        text qr_code
        boolean qr_enabled
        timestamptz pdf_downloaded_at
        timestamptz created_at
    }

    pharmacy_claims {
        uuid id PK
        uuid prescription_id FK
        uuid org_id FK
        uuid claimant_user_id FK
        text status
        jsonb items_filled_json
        boolean qr_verified
        timestamptz verified_at
    }

    lab_orders {
        uuid id PK
        uuid consultation_id FK
        uuid patient_id FK
        uuid org_id FK
        jsonb tests_json
        text status
        text results_url
        timestamptz created_at
    }

    audit_events {
        uuid id PK
        timestamptz occurred_at
        uuid actor_user_id FK
        uuid actor_org_id FK
        text actor_role
        text action
        text subject_table
        uuid subject_id
        text reason
        jsonb context
    }

    users ||--o{ consultations : "patient_id"
    users ||--o{ consultations : "gp_id"
    users ||--o{ consultations : "specialist_id"
    users }o--|| organizations : "org_id"

    consultations ||--o{ ai_intake : "consultation_id"
    consultations ||--o{ messages : "consultation_id"
    consultations ||--o{ referrals : "consultation_id"
    consultations ||--o{ prescriptions : "consultation_id"
    consultations ||--o{ lab_orders : "consultation_id"

    prescriptions ||--o{ pharmacy_claims : "prescription_id"
    organizations ||--o{ pharmacy_claims : "org_id"
    organizations ||--o{ lab_orders : "org_id"

    users ||--o{ audit_events : "actor_user_id"
    organizations ||--o{ audit_events : "actor_org_id"
```

Integration Points

WhatsApp Cloud API Integration:
- **OTP Delivery**: Template-based messages for authentication
- **Video Call Initiation**: Deep links for GP and specialist consultations
- **Notification Delivery**: Treatment updates and appointment reminders

Google Maps API Integration:
- **Pharmacy Discovery**: Geolocation-based verified pharmacy finder
- **Distance Calculation**: Pharmacy proximity ranking for patients
- **Directions**: Navigation integration for pharmacy visits

Push Notification Service:
- **In-App Notifications**: Real-time consultation updates
- **Background Alerts**: Prescription ready, test results available
- **Fallback Communication**: When WhatsApp delivery fails

File Storage Integration (Cloudflare R2):
- **Prescription PDFs**: Secure document generation and storage
- **Diagnostic Results**: Medical imaging and lab report storage
- **User Documents**: Profile pictures, ID verification uploads

Data Retention/Classification and PII/PHI Handling

Data Classification

Data Type	Classification	Retention Period	Storage Location	Access Level
Chat Messages	PHI	90 days hot, then archive	PostgreSQL → R2	Patient + GP + Support (masked)
Prescriptions	PHI	24 months hot, then archive	PostgreSQL → R2	Patient + Prescriber + Pharmacy (items only)
Diagnostic Results	PHI	7 years	PostgreSQL → R2	Patient + Ordering Specialist
AI Intake Summaries	PHI	Tied to consultation retention	PostgreSQL → R2	Patient + Assigned GP
Audit Logs	Security Critical	2 years hot, 7 years archive	PostgreSQL → R2	Ops + Support (audited access)
User Profiles	PII	Until account deletion	PostgreSQL	Self + GP (assigned) + Support (masked)
WhatsApp OTP Codes	Temporary	5 minutes TTL	PostgreSQL	System only

PII/PHI Handling Policies

Patient Data Protection:
- **Full Access**: Patient (own data), assigned GP (during consultation)
- **Masked Access**: Support team (name initials, last 4 digits of phone)
- **No Access**: Pharmacy admins (medication items only)
- **Minimal Access**: Lab admins (name, order ID, tests, masked phone)

Partner Data Minimization:
- **Pharmacy Portals**: Show only medication details, quantities, instructions
- **Lab Portals**: Show only patient name, order ID, test list, masked phone
- **Specialist Portals**: Show only referred patient data with GP notes

Data Encryption Standards:
- **In Transit**: TLS 1.2+ for all API communications
- **At Rest**: AES-256 encryption for database and file storage
- **Backup Encryption**: All archived data encrypted with managed keys

Access Audit Requirements:
- **Purpose-Based Logging**: All access requires business justification

- **Time-Boxed Sessions**: Support access expires after 30 minutes
- **Role-Based Permissions**: Enforced at database level via RLS
- **Unmasking Events**: Full audit trail when PII masking removed

ADRs (Architecture Decision Records)

ADR-001: WhatsApp-Only Authentication

Status: Accepted
Context: Patient onboarding must be frictionless while maintaining security
Decision: Use WhatsApp OTP as primary authentication, Firebase Phone Auth as backup
Consequences: High conversion rates in WhatsApp-heavy markets, dependency on WhatsApp availability
Alternatives Considered: SMS OTP, Email verification, Social login

ADR-002: GP-Gated Specialist Referrals

Status: Accepted
Context: Healthcare-as-a-Service model requires clinical governance
Decision: All specialist referrals must be initiated by GP, no patient self-booking
Consequences: Better clinical outcomes, slower patient-initiated specialist access
Alternatives Considered: Open marketplace, hybrid model with GP review

ADR-003: QR Disable on PDF Download

Status: Accepted
Context: Prevent duplicate prescription fulfillment and pharmacy fraud
Decision: Downloading prescription PDF permanently disables QR code
Consequences: Enhanced security, reduced flexibility for patients
Alternatives Considered: Time-limited QR, multiple-use QR with tracking

ADR-004: Row-Level Security for Data Protection

Status: Accepted
Context: Healthcare data requires fine-grained access control
Decision: Implement PostgreSQL RLS policies for all sensitive tables
Consequences: Strong security model, increased query complexity
Alternatives Considered: Application-level permissions, separate databases per role

ADR-005: Supabase for Rapid Development

Status: Accepted
Context: 3-month timeline requires rapid development platform
Decision: Use Supabase for database, auth, real-time, and file storage
Consequences: Fast development, vendor lock-in, potential scaling limitations
Alternatives Considered: Custom PostgreSQL + Auth0, AWS Amplify, Firebase

ADR-006: React Native for Cross-Platform Mobile

Status: Accepted
Context: Need mobile app with limited development resources
Decision: Build React Native app targeting Android first
Consequences: Code reuse, good performance, single codebase maintenance
Alternatives Considered: Native Android, Flutter, Progressive Web App

ADR-007: Cloudflare R2 for Archival Storage

Status: Accepted
Context: Healthcare data retention requirements with budget constraints
Decision: Use Cloudflare R2 for archival storage due to zero egress fees
Consequences: Cost optimization, vendor diversity, additional integration complexity
Alternatives Considered: AWS S3, Google Cloud Storage, Azure Blob Storage

---

E) Delivery Plan (Phased)

Phase Structure Overview

The Mediconnect MVP will be delivered through 6 distinct phases over a 3-month timeline, with each phase building upon the previous to ensure a stable, scalable healthcare platform.

Phase 1: Discovery & Inception (Weeks 1-2)

Objectives

- Validate technical assumptions and architectural decisions
- Complete stakeholder alignment on requirements and scope
- Setup development infrastructure and team processes
- Resolve critical open questions and conflicts

Entry Criteria

- [ ] All stakeholder approvals obtained
- [ ] Development team assembled and onboarded
- [ ] Budget and resource allocation confirmed
- [ ] Technical environment access provisioned

Key Activities & Artifacts

Week 1: Requirements Finalization
- Resolve 10 open questions identified in requirements analysis
- Confirm most current wireframe versions among multiple iterations
- Align on final timeline and specialist rollout scope
- Complete stakeholder sign-off on consolidated requirements document

Week 2: Technical Foundation
- Setup development, staging, and production environments
- Configure Supabase project with initial database schema
- Establish WhatsApp Cloud API integration and testing
- Create initial React Native project structure with routing
- Setup CI/CD pipelines and development workflows

Deliverables

- [ ] **Consolidated Requirements Document** - Single source of truth
- [ ] **Technical Architecture Document** - Detailed component specifications
- [ ] **Development Environment Setup** - All environments operational
- [ ] **Team Onboarding Complete** - Developers productive on tech stack
- [ ] **Project Management Framework** - Sprint planning and tracking tools

Exit Criteria

- [ ] All critical open questions resolved with documented decisions
- [ ] Technical spike completed proving WhatsApp + Supabase integration
- [ ] Development team can commit and deploy to staging environment
- [ ] Project tracking shows realistic sprint velocity estimates

Gating Checklist

- [ ] WhatsApp OTP delivery tested and confirmed working in Kenya
- [ ] Supabase RLS policies validated with healthcare access patterns
- [ ] React Native development environment working on team devices
- [ ] Initial API endpoints responding with mock data
- [ ] Team sprint planning completed with story point estimates

Phase 2: Foundation (Weeks 3-4)

Objectives

- Build core authentication and user management systems
- Implement foundational database schema with RLS policies
- Create basic mobile app structure and navigation
- Establish real-time chat infrastructure

Entry Criteria

- [ ] All Phase 1 deliverables completed and approved
- [ ] Development team sprint velocity established
- [ ] Technical infrastructure proven stable
- [ ] API documentation framework established

Key Activities & Artifacts

Week 3: Core Infrastructure
- Implement complete database schema with all tables and relationships
- Deploy Row-Level Security policies for all user roles
- Build WhatsApp OTP authentication flow with error handling
- Create user registration and profile management systems
- Setup comprehensive audit logging framework

Week 4: Mobile App Foundation
- Build React Native app with navigation and routing
- Implement authentication screens and session management
- Create reusable UI components following design specifications
- Setup in-app chat foundation with Supabase Realtime
- Implement English/Swahili localization framework

Deliverables

- [ ] **Complete Database Schema** - All tables with RLS policies deployed
- [ ] **Authentication System** - WhatsApp OTP + Firebase fallback working
- [ ] **Mobile App Core** - Navigation, routing, and basic screens
- [ ] **Chat Infrastructure** - Real-time messaging foundation
- [ ] **Audit Framework** - Comprehensive event logging system

Exit Criteria

- [ ] Users can register and authenticate via WhatsApp OTP
- [ ] All user roles can log into their respective portals
- [ ] Mobile app navigation between screens works smoothly
- [ ] Real-time chat sends and receives messages correctly
- [ ] All database access properly restricted by RLS policies

Gating Checklist

- [ ] Authentication flow tested with 10+ test users
- [ ] Database performance tested with 1000+ concurrent queries
- [ ] Mobile app tested on multiple Android devices and screen sizes
- [ ] Chat messages delivered in <2 seconds under normal conditions
- [ ] Security audit shows no unauthorized data access possible

Phase 3: Build - Core Features (Weeks 5-8)

Objectives

- Implement AI intake questionnaire and GP assignment logic
- Build complete consultation flow with WhatsApp video integration
- Create prescription system with QR code generation and security
- Develop pharmacy discovery and verification workflows

Entry Criteria

- [ ] Phase 2 foundation systems stable and tested
- [ ] AI intake question logic designed and approved
- [ ] WhatsApp video call integration tested successfully
- [ ] Prescription QR code security model validated

Key Activities & Artifacts

Week 5: AI Intake & GP Assignment
- Build adaptive AI intake questionnaire with branching logic
- Implement AI summary generation from patient responses
- Create GP availability tracking and assignment algorithms
- Develop consultation request queue and management system
- Setup notification systems for consultation assignments

Week 6: Consultation Flow
- Implement complete GP consultation workflow
- Integrate WhatsApp video call launching and metadata tracking
- Build in-app chat thread with persistent message history
- Create consultation extension logic for specialist referrals
- Develop consultation summary and note-taking interfaces

Week 7: Prescription Management
- Build prescription creation and item management system
- Implement secure QR code generation with unique tokens
- Create prescription PDF generation with security warnings
- Develop QR code verification and disabling mechanisms
- Setup prescription status tracking and history

Week 8: Pharmacy Integration
- Build pharmacy discovery with geolocation and mapping
- Implement pharmacy verification and onboarding workflows
- Create pharmacy portal with QR scanning capabilities
- Develop item-only prescription views with PII masking
- Setup pharmacy claim processing and status updates

Deliverables

- [ ] **AI Intake System** - Complete questionnaire with smart routing
- [ ] **GP Consultation Platform** - Full consultation workflow with video
- [ ] **Prescription System** - QR generation, PDF creation, security controls
- [ ] **Pharmacy Network** - Discovery, verification, and claim processing
- [ ] **Mobile App V1** - Patient-facing app with core features complete

Exit Criteria

- [ ] Patients can complete AI intake and receive GP assignment
- [ ] GP consultations successfully conducted via WhatsApp video
- [ ] Prescriptions generated with working QR codes and PDF options
- [ ] Pharmacies can verify prescriptions and process claims
- [ ] End-to-end patient journey from intake to prescription works

Gating Checklist

- [ ] AI intake completion time averages <90 seconds
- [ ] GP assignment happens within 30 minutes during business hours
- [ ] WhatsApp video calls launch successfully 95%+ of time
- [ ] QR codes scan correctly and show item-only views
- [ ] PDF download permanently disables QR as designed

Phase 4: Build - Advanced Features (Weeks 9-10)

Objectives

- Implement specialist referral workflow with appointment coordination
- Build diagnostics ordering and results management system
- Create comprehensive notification system across all channels
- Develop partner portals for specialists and diagnostics admins

Entry Criteria

- [ ] Phase 3 core features completed and stable
- [ ] Specialist onboarding process defined and tested
- [ ] Diagnostics partners identified and integration requirements confirmed
- [ ] Notification templates approved for all communication scenarios

Key Activities & Artifacts

Week 9: Specialist Referral System
- Build GP-initiated specialist referral workflow
- Implement curated specialist option presentation (3-4 choices)
- Create specialist portal with referral queue and patient management
- Develop appointment coordination between patients and specialists
- Setup referral status tracking and communication workflows

Week 10: Diagnostics & Notifications
- Implement specialist-only diagnostic test ordering system
- Build diagnostics portal with minimal PII exposure
- Create diagnostic results upload and sharing workflows
- Develop comprehensive notification system for WhatsApp and in-app
- Setup notification templates and delivery tracking

Deliverables

- [ ] **Specialist Referral System** - Complete workflow from GP to specialist
- [ ] **Diagnostics Platform** - Test ordering, processing, and results delivery
- [ ] **Notification System** - WhatsApp and in-app notifications working
- [ ] **Specialist Portal** - Web interface for specialist workflow management
- [ ] **Diagnostics Portal** - Lab interface with security and privacy controls

Exit Criteria

- [ ] GPs can successfully refer patients to specialists with appointment booking
- [ ] Specialists can order diagnostic tests with proper patient data protection
- [ ] Notifications delivered reliably across WhatsApp and in-app channels
- [ ] Partner portals allow efficient workflow management for healthcare providers
- [ ] End-to-end specialist and diagnostics journeys work smoothly

Gating Checklist

- [ ] Specialist referral appointments booked within 4-hour SLA
- [ ] Diagnostic orders transmitted to labs with correct PII minimization
- [ ] Notification delivery rate exceeds 98% for critical messages
- [ ] Partner portal load times under 2 seconds for all key functions
- [ ] Specialist and diagnostics workflows tested by actual healthcare providers

Phase 5: Hardening (Weeks 11-12)

Objectives

- Conduct comprehensive security audit and penetration testing
- Implement full monitoring, alerting, and observability systems
- Perform load testing and performance optimization
- Complete integration testing across all systems and user journeys

Entry Criteria

- [ ] All Phase 4 features completed and functionally stable
- [ ] Security audit scope defined with external auditors if needed
- [ ] Performance testing scenarios identified for 10k user capacity
- [ ] Monitoring and alerting requirements documented

Key Activities & Artifacts

Week 11: Security & Performance
- Conduct comprehensive security audit of all systems and data flows
- Perform penetration testing on authentication and data access controls
- Execute load testing to validate 10k MAU capacity and 60 concurrent users
- Implement performance optimization based on bottleneck identification
- Setup comprehensive monitoring with Sentry error tracking and DataDog metrics

Week 12: Integration & Quality Assurance
- Complete end-to-end integration testing across all user journeys
- Validate all RLS policies prevent unauthorized data access
- Test disaster recovery procedures and backup restoration
- Conduct user acceptance testing with healthcare providers
- Finalize deployment procedures and rollback capabilities

Deliverables

- [ ] **Security Audit Report** - Comprehensive security validation with remediation
- [ ] **Performance Test Results** - Load testing validation for target capacity
- [ ] **Monitoring Dashboard** - Real-time system health and business metrics
- [ ] **Integration Test Suite** - Automated testing for all critical user paths
- [ ] **Deployment Playbook** - Production deployment and rollback procedures

Exit Criteria

- [ ] Security audit shows no critical or high-severity vulnerabilities
- [ ] Performance testing validates system handles target load with SLA compliance
- [ ] All critical user journeys tested and working reliably
- [ ] Monitoring systems provide complete visibility into system health
- [ ] Production deployment procedures tested and validated

Gating Checklist

- [ ] Penetration testing shows no unauthorized data access possible
- [ ] Load testing confirms <500ms P75 latency under target load
- [ ] 99.5% uptime validated through extended stability testing
- [ ] All healthcare provider workflows tested by actual users
- [ ] Disaster recovery tested with <4 hour recovery time confirmed

Phase 6: Launch (Week 13)

Objectives

- Deploy production systems with monitoring and alerting active
- Execute phased user rollout starting with 500 initial users
- Conduct production validation and immediate issue resolution
- Establish ongoing support and operational procedures

Entry Criteria

- [ ] All Phase 5 hardening activities completed successfully
- [ ] Production environment configured and validated
- [ ] Support team trained and ready for user assistance
- [ ] Healthcare partners onboarded and ready for patient flow

Key Activities & Artifacts

Production Deployment & Validation
- Deploy all systems to production environment with zero-downtime procedures
- Execute smoke tests across all critical user journeys in production
- Activate monitoring, alerting, and error tracking systems
- Conduct final security validation of production configuration

Phased User Rollout
- Onboard initial cohort of 50 internal test users for production validation
- Release to first 500 external users with close monitoring and support
- Monitor all success metrics and system performance indicators
- Collect user feedback and address any critical issues immediately

Deliverables

- [ ] **Production System** - Fully deployed and operational platform
- [ ] **User Onboarding Process** - Streamlined user registration and orientation
- [ ] **Support Documentation** - User guides and troubleshooting resources
- [ ] **Operational Runbooks** - Incident response and system maintenance procedures
- [ ] **Launch Metrics Dashboard** - Real-time tracking of success indicators

Exit Criteria

- [ ] Production systems stable with no critical issues
- [ ] First 500 users successfully onboarded and using the platform
- [ ] All success metrics tracking and within expected ranges
- [ ] Support team successfully handling user requests
- [ ] Healthcare partners providing services through the platform

Gating Checklist

- [ ] Production deployment completed with no system downtime
- [ ] First 500 users able to complete end-to-end healthcare journeys
- [ ] System performance meeting all SLA requirements under real user load
- [ ] Support ticket resolution times meeting established targets
- [ ] Healthcare partner feedback positive with no major workflow issues

Sprint-by-Sprint Backlog (First 4 Sprints)

Sprint 1 (Weeks 1-2): Foundation Setup

Sprint Goal: Establish development foundation and resolve critical requirements gaps

Epic	**Story**	**Story Points**	**Dependencies**	**Acceptance Criteria**
Requirements	Resolve 10 critical open questions	8	Stakeholder availability	All questions resolved with documented decisions
Technical Setup	Configure Supabase project and initial schema	13	Supabase account setup	Database accessible with base tables created
Technical Setup	Setup React Native project with navigation	8	Development environment	App runs with basic navigation between screens
Technical Setup	Integrate WhatsApp Cloud API for OTP testing	13	WhatsApp API access	OTP delivery working in Kenya test environment
Process	Establish CI/CD pipeline	8	Repository setup	Code deploys automatically to staging
Sprint Total		**50 points**		

Sprint 2 (Weeks 3-4): Authentication & Core Infrastructure

Sprint Goal: Complete user authentication and core database functionality

Epic	**Story**	**Story Points**	**Dependencies**	**Acceptance Criteria**
Authentication	Build WhatsApp OTP authentication flow	21	WhatsApp API integration	Users can register and login via WhatsApp OTP
Database	Implement complete schema with RLS policies	21	Database design approval	All tables created with proper security policies
Mobile App	Create app navigation and basic UI components	13	Design specification	App navigation works smoothly on Android devices
Chat	Setup Supabase Realtime for messaging	13	Supabase configuration	Messages send and receive in real-time
Localization	Implement English/Swahili language support	8	Translation completion	App displays correctly in both languages
Sprint Total		**76 points**		

Sprint 3 (Weeks 5-6): AI Intake & Consultation Flow

Sprint Goal: Build AI intake questionnaire and GP consultation workflow

Epic	Story	Story Points	Dependencies	Acceptance Criteria
AI Intake	Build adaptive intake questionnaire	21	Question logic design	Questionnaire completes in <90 seconds
AI Intake	Implement AI summary generation	13	AI service integration	Summaries generated and visible to GPs
Consultation	Create GP assignment and queue management	13	GP availability tracking	GPs assigned within 30 minutes during hours
Consultation	Integrate WhatsApp video call launching	21	WhatsApp video API	Video calls launch successfully 95%+ of time
Consultation	Build in-app chat with message persistence	13	Real-time infrastructure	Chat history persists and loads quickly
Sprint Total		**81 points**		

Sprint 4 (Weeks 7-8): Prescription & Pharmacy System

Sprint Goal: Complete prescription management and pharmacy integration

Epic	Story	Story Points	Dependencies	Acceptance Criteria
Prescription	Build prescription creation system	13	Consultation flow complete	GPs can create prescriptions with items
Prescription	Implement QR code generation and security	21	Security model approval	QR codes generated uniquely and securely
Prescription	Create PDF generation with disable logic	13	PDF service setup	PDF download permanently disables QR
Pharmacy	Build pharmacy discovery with maps	13	Google Maps API	Patients can find nearby verified pharmacies
Pharmacy	Create pharmacy portal with QR scanning	21	Pharmacy onboarding	Pharmacies can verify prescriptions via QR
Testing	End-to-end testing of core patient journey	8	All features complete	Patients can complete intake through prescription
Sprint Total		**89 points**		

Milestones & Timeline

Milestone	Target Date	Success Criteria	Critical Path Dependencies
D1 - Scope & Risks Locked	End Week 2	All open questions resolved, tech stack validated	Stakeholder alignment, WhatsApp integration proven
D2 - Architecture & Plan Validated	End Week 4	Core infrastructure operational, team velocity established	Database schema complete, authentication working
D3 - Core Features Complete	End Week 8	Patient can complete full consultation journey	AI intake, GP consultation, prescription system working
D4 - Launch Readiness	End Week 12	Security audited, performance validated, partners ready	Load testing passed, healthcare providers onboarded
Production Launch	End Week 13	500 users onboarded and using platform successfully	All systems stable, support team operational

Critical Path Analysis

Primary Critical Path: Authentication → AI Intake → GP Consultation → Prescription → Pharmacy Integration
- **Total Duration**: 8 weeks of development + 4 weeks setup/hardening
- **Key Dependencies**: WhatsApp API, Supabase stability, healthcare partner onboarding
- **Risk Mitigation**: Parallel development where possible, early integration testing

Secondary Critical Path: Database Schema → RLS Policies → Partner Portals → Security Audit
- **Key Risk**: Complex healthcare data access patterns may require RLS policy refinement
- **Mitigation**: Early RLS validation with realistic data access scenarios

Slack & Buffer Policy

Time Buffers:
- **Sprint Planning Buffer**: 20% capacity reserved for unplanned work and technical debt
- **Integration Buffer**: 1 week allocated between build and hardening phases
- **Launch Buffer**: 3 days allocated for production deployment and immediate issue resolution

Scope Buffers:
- **MoSCoW Flexibility**: “Should Have” features can be moved to post-launch if needed
- **Feature Flags**: All major features behind flags for gradual rollout and quick rollback
- **Partner Dependencies**: Backup partner options identified for critical integrations

F) Quality, Security & Compliance

Test Strategy

Test Pyramid Structure

```
                     /\
                    /E2E\
                   /Tests\
                  /-------\
                 /Contract\
                / & API   \
               /   Tests   \
              /____________\
             /              \
            /   Unit Tests   \
           /                 \
          /___________________\
```

Unit Testing (Foundation - 70% Coverage)

Coverage Thresholds:
- **Overall Coverage**: Minimum 70%, target 80%
- **Critical Business Logic**: Minimum 90% coverage
- **Healthcare Data Handling**: 100% coverage required

Key Testing Areas:
- **Authentication Logic**: OTP generation, validation, session management
- **AI Intake Processing**: Question logic, summary generation, red flag detection
- **Prescription Security**: QR code generation, validation, disabling mechanisms
- **RLS Policy Enforcement**: Data access controls, role-based permissions
- **Healthcare Data Transformations**: PII masking, data minimization

Testing Tools & Framework:
- **React Native**: Jest + React Native Testing Library
- **Backend Services**: Jest + Supertest for API testing
- **Database**: pg-tap for PostgreSQL-specific testing
- **Mocking Strategy**: Mock external APIs (WhatsApp, Maps) for deterministic testing

API/Contract Testing (Integration - 60 Test Cases)

Critical API Endpoints:

```
Authentication APIs:
  - POST /auth/whatsapp/otp: Rate limiting, phone validation, template delivery
  - POST /auth/whatsapp/verify: Code validation, session creation, audit logging

Consultation APIs:
  - POST /consultations: GP assignment, consultation creation
  - PATCH /consultations/{id}: Status updates, specialist assignment
  - GET /consultations/{id}/messages: Message retrieval with pagination

Prescription APIs:
  - POST /prescriptions: Prescription creation with QR generation
  - GET /prescriptions/{id}: Patient/GP access control validation
  - POST /pharmacy/claims/verify-qr: QR validation and disabling logic

Healthcare Data APIs:
  - POST /intake: AI processing and summary generation
  - POST /referrals: Specialist assignment and booking coordination
  - POST /labs/orders: Diagnostics ordering with PII minimization
```

Contract Testing Strategy:
- **Provider Testing**: Backend services implement agreed contracts
- **Consumer Testing**: Mobile app and web portals consume APIs correctly
- **Schema Validation**: OpenAPI 3.0 spec enforcement for all endpoints
- **Security Testing**: Authentication, authorization, and input validation

End-to-End Smoke Testing (25 Critical Paths)

Core User Journeys:
1. **Complete Patient Onboarding**: Language selection → Consent → OTP → Profile setup
2. **AI Intake to GP Assignment**: Questionnaire completion → AI summary → GP assignment
3. **WhatsApp Video Consultation**: Call initiation → Video session → Notes saving
4. **Prescription with QR Fulfillment**: Prescription creation → QR generation → Pharmacy scanning
5. **Prescription with PDF Fallback**: PDF download → QR disabling → Manual pharmacy fulfillment
6. **Specialist Referral Journey**: GP referral → Patient selection → Appointment booking
7. **Diagnostics Order Flow**: Specialist order → Lab processing → Results delivery

Cross-Browser/Device Testing:
- **Android Devices**: Samsung Galaxy (mid-range), Google Pixel (high-end), generic (low-end)
- **Web Browsers**: Chrome, Firefox, Safari for healthcare provider portals
- **Network Conditions**: 3G, 4G, WiFi scenarios for mobile app reliability

Performance Testing (Budget Allocation)

Load Testing Scenarios:

```
Authentication Load:
  - Target: 100 concurrent OTP requests/minute
  - Success Rate: >95% OTP delivery within 5 seconds
  - Bottleneck Detection: WhatsApp API rate limits

Consultation Load:
  - Target: 60 simultaneous chat sessions
  - Latency: <500ms P75 for message sending/receiving
  - Real-time: Messages delivered within 2 seconds

Database Performance:
  - Target: 1000 concurrent database queries
  - RLS Policy Performance: <100ms overhead
  - Connection Pooling: Efficient resource utilization
```

Performance Budgets:
- **Mobile App Bundle Size**: <10MB for Android APK
- **Page Load Times**: <2 seconds for all partner portals
- **API Response Times**: P75 <500ms, P95 <1000ms
- **Database Query Times**: Complex queries <200ms

Security Testing (SAST/DAST/Dependency Scanning)

Static Application Security Testing (SAST):
- **Tools**: ESLint Security Plugin, Semgrep, SonarQube
- **Coverage**: All TypeScript/JavaScript code for vulnerabilities
- **Focus Areas**: SQL injection prevention, XSS protection, authentication flaws

Dynamic Application Security Testing (DAST):
- **Tools**: OWASP ZAP, Burp Suite Professional
- **Testing Scope**: All API endpoints and web interfaces
- **Security Scenarios**: Authentication bypass, authorization flaws, input validation

Dependency Scanning:
- **Tools**: npm audit, Snyk, GitHub Security Advisories
- **Policy**: Zero known high/critical vulnerabilities in production
- **Update Strategy**: Weekly dependency updates, immediate patch for critical issues

Healthcare-Specific Security Testing:
- **RLS Policy Validation**: Ensure no unauthorized data access across roles
- **PII Masking Verification**: Confirm partner portals show only necessary data
- **Audit Log Integrity**: Verify all sensitive actions properly logged
- **Session Security**: Test session timeout, concurrent session handling

Data Migration Testing

Migration Scenarios:
- **Schema Updates**: Test RLS policy changes without breaking existing access
- **Data Archival**: Validate 90-day chat archival to Cloudflare R2
- **Prescription Retention**: Test 24-month prescription archival with restore capability
- **User Data Export**: Validate GDPR-style data export functionality

Definition of Ready/Done Checklists

Definition of Ready (Before Sprint Planning)

User Story Ready Criteria:
- [ ] **Business Value Clear**: Story explains why feature needed and success criteria
- [ ] **Acceptance Criteria Complete**: Specific, testable criteria defined
- [ ] **Dependencies Identified**: Technical and business dependencies documented
- [ ] **Design Available**: UI/UX mockups available for user-facing features
- [ ] **Security Requirements**: Healthcare data handling requirements specified
- [ ] **Performance Criteria**: Response time and load requirements defined
- [ ] **Test Scenarios**: Key test cases identified for QA planning
- [ ] **Story Points Estimated**: Team consensus on effort required

Definition of Done (Before Story Completion)

Code Quality Standards:
- [ ] **Code Review Completed**: Minimum 2 reviewer approval for healthcare features
- [ ] **Unit Tests Written**: Coverage meets threshold (70% overall, 90% business logic)
- [ ] **Security Review**: SAST scanning passed, no high/critical vulnerabilities
- [ ] **Performance Validated**: API responses within SLA requirements
- [ ] **Accessibility Checked**: WCAG 2.2 AA compliance for patient-facing features

Healthcare Compliance:
- [ ] **RLS Policies Tested**: Data access properly restricted by user role
- [ ] **PII Handling Verified**: Partner interfaces show only necessary data
- [ ] **Audit Events Logged**: All data access and modifications logged
- [ ] **Error Handling**: Graceful failure modes that don’t expose sensitive data

Integration Requirements:
- [ ] **API Integration Tested**: External service integration working reliably
- [ ] **Cross-Platform Verified**: Feature works on target Android devices
- [ ] **Localization Complete**: English/Swahili text implemented correctly
- [ ] **Documentation Updated**: API docs and user guides reflect changes

Release Criteria & Go/No-Go Decision Framework

Pre-Release Validation Gates

Gate 1: Feature Complete (Week 10)
- [ ] All “Must Have” features implemented and tested
- [ ] Critical user journeys working end-to-end
- [ ] Healthcare provider partners trained and ready
- [ ] Performance testing shows system meets SLA requirements

Gate 2: Security Validated (Week 11)
- [ ] Security audit completed with no critical/high vulnerabilities
- [ ] Penetration testing shows no unauthorized data access possible
- [ ] RLS policies tested against all attack scenarios
- [ ] Healthcare compliance requirements validated

Gate 3: Production Ready (Week 12)
- [ ] Load testing validates 10k MAU capacity with performance SLAs
- [ ] Disaster recovery tested with <4 hour recovery time

- [ ] Monitoring and alerting operational with proper escalation
- [ ] Support team trained and operational procedures documented

Go/No-Go Decision Criteria

Go Criteria (All Must Be Met):
- [ ] **User Safety**: No critical bugs that could impact patient care
- [ ] **Data Security**: Healthcare data properly protected per regulations
- [ ] **System Stability**: 99.5% uptime demonstrated in staging for 72 hours
- [ ] **Partner Readiness**: Healthcare providers successfully using system
- [ ] **Support Capability**: Team ready to handle user issues 24/7

No-Go Criteria (Any One Triggers Delay):
- [ ] **Critical Security Vulnerability**: High/critical issues in security audit
- [ ] **Performance Failure**: System cannot handle target load within SLA
- [ ] **Data Loss Risk**: Backup/recovery procedures not validated
- [ ] **Regulatory Non-Compliance**: Healthcare regulations not properly addressed
- [ ] **Partner Network Insufficient**: <15 verified pharmacies operational

Launch Decision Authority

- **Final Go/No-Go Decision**: Product Manager + Engineering Lead + Medical Director
- **Escalation Path**: CEO for business impact decisions, CTO for technical risks
- **External Approvals**: Legal (compliance), Healthcare Advisors (clinical safety)

Incident Response Framework

Severity Levels & Response Times

Severity	Description	Response Time	Examples
P0 - Critical	Patient safety risk or complete service outage	15 minutes	Authentication down, prescription data corrupted
P1 - High	Major feature unavailable, high user impact	1 hour	WhatsApp integration failure, GP assignment broken
P2 - Medium	Feature degraded, moderate user impact	4 hours	Pharmacy map loading slowly, notification delays
P3 - Low	Minor issues, low user impact	24 hours	UI inconsistencies, non-critical analytics issues

Incident Response Procedures

P0/P1 Incident Response:
1. **Immediate Response** (0-15 minutes):
- Page on-call engineer and product manager

- Create incident channel in Slack
- Begin preliminary investigation and impact assessment

1. **Escalation & Communication** (15-30 minutes):
   - Notify medical director for patient safety issues
   - Update status page with customer-facing communication
   - Engage additional engineers if needed for resolution

2. **Resolution & Recovery** (30 minutes - 4 hours):
   - Implement fix or activate rollback procedures
   - Validate fix resolves issue without creating new problems
   - Monitor system stability for 2 hours post-resolution

3. **Post-Incident** (Within 48 hours):
   - Conduct blameless post-mortem with timeline reconstruction
   - Document root cause analysis and preventive measures
   - Update runbooks and monitoring based on lessons learned

Logging, Metrics, Tracing & Alert SLOs

Logging Strategy

Application Logs:

```
Healthcare Events:
  - User authentication attempts and outcomes
  - Consultation start/end with participants
  - Prescription creation and fulfillment status
  - Specialist referrals and appointment bookings
  - Diagnostic orders and results delivery

Security Events:
  - Failed authentication attempts (potential attacks)
  - Unauthorized data access attempts
  - PII unmasking events with business justification
  - Admin privilege escalations and access grants

System Events:
  - API response times and error rates
  - Database query performance and connection health
  - External service integration failures (WhatsApp, Maps)
  - Background job processing and queue health
```

Log Retention Policy:
- **Application Logs**: 30 days hot storage, 12 months cold storage
- **Security Logs**: 24 months hot storage for compliance audit
- **System Logs**: 7 days hot storage, 90 days cold storage
- **Healthcare Audit Logs**: 7 years retention per regulatory requirements

Metrics & KPI Dashboard

Technical Health Metrics:

```
Availability SLIs:
  - Overall system uptime: 99.5% monthly target
  - Authentication service uptime: 99.9% (critical path)
  - WhatsApp integration uptime: 99.0% (external dependency)

Performance SLIs:
  - API P75 response time: <500ms
  - API P95 response time: <1000ms
  - Chat message delivery: <2 seconds
  - Mobile app crash rate: <0.1%

Error Rate SLIs:
  - Overall API error rate: <1%
  - Authentication error rate: <0.5%
  - Prescription QR generation errors: <0.1%
```

Business Health Metrics:

```
User Engagement:
  - Daily/Weekly/Monthly Active Users
  - Consultation completion rate (target: >85%)
  - Average consultation duration (target: 15±2 minutes)
  - User retention (30-day: >70%, 90-day: >50%)

Healthcare Quality:
  - GP response time to consultation requests
  - Specialist referral acceptance rate (target: >80%)
  - Prescription fulfillment rate (target: >90% in 7 days)
  - Patient satisfaction rating (target: >4.2/5)

Partner Performance:
  - Pharmacy QR scanning success rate
  - Diagnostic test completion time
  - Partner portal usage and efficiency metrics
```

Alerting Strategy & SLOs

Critical Alerts (P0 - Immediate Page):

- System uptime drops below 99.5%
- Authentication service completely unavailable
- Database connection failures preventing new consultations
- WhatsApp OTP delivery failure rate >5%
- Any security breach or unauthorized data access

Warning Alerts (P1 - 15 Minute Response):

- API P95 response time >1000ms for >5 minutes
- Error rate exceeds 2% for any critical endpoint
- Chat message delivery delay >5 seconds consistently
- Prescription QR generation failure rate >1%

Monitoring Alerts (P2 - Business Hours):

- Unusual traffic patterns or user behavior
- Partner portal performance degradation
- High resource utilization (CPU >80%, Memory >90%)
- Backup job failures or data archival issues

Tracing & Observability

Distributed Tracing:
- **Tools**: Sentry Performance + Custom trace correlation
- **Coverage**: All critical user journeys end-to-end
- **Retention**: 7 days detailed traces, 30 days aggregated metrics

Key Trace Scenarios:
- Patient authentication flow (OTP generation → verification → session creation)
- Complete consultation journey (intake → GP assignment → video call → notes)
- Prescription fulfillment (creation → QR generation → pharmacy verification)
- Specialist referral workflow (GP referral → booking coordination → appointment)

Performance Monitoring:
- **Real User Monitoring**: Mobile app performance on actual devices
- **Synthetic Monitoring**: Automated testing of critical paths every 5 minutes
- **Infrastructure Monitoring**: Database, API gateway, and external service health

---

G) People & Process

RACI Chart

Activity/Decision	**Product Manager**	**Tech Lead**	**Frontend Dev**	**Backend Dev**	**DevOps**	**QA Lead**	**Medical Director**	**Ops Manager**
Requirements & Planning								
Product Requirements Definition	**R/A**	C	C	C	I	C	**C**	C
Technical Architecture Decisions	C	**R/A**	C	C	C	I	I	I
Sprint Planning	**R/A**	C	C	C	I	C	I	I
Feature Prioritization	**R/A**	C	I	I	I	I	**C**	C
Development & Delivery								
Code Review & Approval	I	**R/A**	C	C	I	I	I	I
Database Schema Changes	C	**R/A**	I	C	I	C	I	I

Activity/Decision	**Product Manager**	**Tech Lead**	**Frontend Dev**	**Backend Dev**	**DevOps**	**QA Lead**	**Medical Director**	**Ops Manager**
Security Implementation	C	**R/A**	C	C	C	C	I	I
Healthcare Compliance	C	C	I	I	I	C	**R/A**	I
Testing & Quality								
Test Strategy Definition	C	C	I	I	I	**R/A**	I	I
Security Audit Execution	C	C	I	I	C	**R/A**	C	I
Performance Testing	I	C	C	C	C	**R/A**	I	I
User Acceptance Testing	**R/A**	I	I	I	I	C	C	C
Operations & Support								
Production Deployment	I	C	I	I	**R/A**	C	I	I

Activity/Decision	**Product Manager**	**Tech Lead**	**Frontend Dev**	**Backend Dev**	**DevOps**	**QA Lead**	**Medical Director**	**Ops Manager**
Incident Response	C	**R/A**	C	C	C	C	I	**C**
Partner Onboarding	C	I	I	I	I	I	I	**R/A**
User Support Escalation	I	C	I	I	I	I	C	**R/A**
Governance & Compliance								
Clinical Safety Review	I	I	I	I	I	I	**R/A**	I
Regulatory Compliance	C	I	I	I	I	I	**R/A**	C
Data Privacy Decisions	C	C	I	C	I	I	**R/A**	I
Go/No-Go Launch Decision	**R/A**	**C**	I	I	I	C	**C**	C

Legend: R = Responsible, A = Accountable, C = Consulted, I = Informed

Ceremonies & Workflows

Sprint Ceremonies (2-Week Sprints)

Sprint Planning (4 hours - Start of each sprint)
- **Participants**: Product Manager, Tech Lead, All Developers, QA Lead
- **Inputs**: Prioritized backlog, team velocity, technical constraints
- **Outputs**: Sprint commitment, task breakdown, dependency identification
- **Cadence**: Every 2 weeks (Monday 9 AM - 1 PM)

Daily Standups (15 minutes)
- **Participants**: All development team members
- **Format**: Yesterday’s progress, today’s plan, blockers/dependencies
- **Special Focus**: Healthcare compliance items, patient safety concerns
- **Cadence**: Daily 9 AM (async updates on Fridays)

Sprint Review/Demo (2 hours - End of sprint)
- **Participants**: Development team + stakeholders (Medical Director, Ops Manager)
- **Outputs**: Working software demonstration, stakeholder feedback
- **Healthcare Focus**: Clinical workflow validation, safety feature demos
- **Cadence**: Every 2 weeks (Thursday 2 PM - 4 PM)

Sprint Retrospective (1.5 hours - End of sprint)
- **Participants**: Development team only
- **Format**: What went well, what could improve, action items
- **Special Topics**: Security practices, healthcare compliance processes
- **Cadence**: Every 2 weeks (Friday 10 AM - 11:30 AM)

Weekly Ceremonies

Technical Architecture Review (1 hour)
- **Participants**: Tech Lead, Senior Developers, DevOps
- **Purpose**: Review technical decisions, address architectural debt
- **Focus Areas**: Security patterns, performance optimization, scalability
- **Cadence**: Wednesday 3 PM - 4 PM

Healthcare Compliance Check (30 minutes)
- **Participants**: Product Manager, QA Lead, Medical Director
- **Purpose**: Review compliance with healthcare regulations and safety standards
- **Outputs**: Compliance action items, risk assessment updates
- **Cadence**: Tuesday 11 AM - 11:30 AM

Partner Stakeholder Sync (45 minutes)
- **Participants**: Product Manager, Ops Manager, Tech Lead
- **Purpose**: Align on partner integration progress and operational readiness
- **Topics**: Pharmacy onboarding, specialist integration, support processes
- **Cadence**: Thursday 4:30 PM - 5:15 PM

Tooling Infrastructure

Project Management & Communication

- **Project Management**: Jira with healthcare-specific workflow states
- **Communication**: Slack with dedicated channels for incidents and compliance
- **Documentation**: Confluence with access controls for sensitive healthcare information

- **Video Calls**: Google Meet for ceremonies and stakeholder meetings

Development & Repository Management

- **Source Control**: GitHub with branch protection rules and required reviews
- **Code Review**: GitHub Pull Requests with mandatory healthcare compliance checklist
- **Package Management**: npm for JavaScript/TypeScript, managed dependency updates
- **Development Environment**: Docker for consistent local development setups

CI/CD & Infrastructure

- **CI/CD Pipeline**: GitHub Actions with healthcare compliance validation steps
- **Infrastructure**: Supabase (managed) + Cloudflare R2 + monitoring tools
- **Secrets Management**: GitHub Secrets with rotation policies for production keys
- **Environment Management**: Separate staging/production with data masking in non-prod

Monitoring & Observability

- **Error Tracking**: Sentry with healthcare-specific error categorization
- **Performance Monitoring**: DataDog with custom healthcare metrics dashboards
- **Log Management**: Supabase logging + external aggregation for compliance
- **Alerting**: PagerDuty integration with escalation policies for patient safety issues

Quality Assurance & Security

- **Testing Framework**: Jest + React Native Testing Library + Cypress for E2E
- **Security Scanning**: Snyk for dependency scanning + GitHub Security Advisories
- **Code Quality**: ESLint + Prettier + SonarQube for maintainability
- **Healthcare Compliance**: Custom linting rules for PII handling and audit logging

Contribution Guidelines & Standards

Code Contribution Workflow

Branch Strategy: GitHub Flow with healthcare-specific naming

```
main (production-ready)
├── feature/patient-authentication-whatsapp
├── feature/prescription-qr-security
├── bugfix/gp-assignment-timeout
└── hotfix/security-audit-findings
```

Pull Request Process:
1. **Development Standards**:
- All code must pass linting and automated tests
- Healthcare-specific code requires additional documentation
- Security-sensitive changes require architecture team review

1. **Review Requirements**:
   - Minimum 2 reviewers for healthcare data handling code
   - Medical Director approval for clinical workflow changes
   - Tech Lead approval for architecture or security changes

2. **Healthcare Compliance Checklist**:
   - [ ] PII handling follows data minimization principles
   - [ ] Audit logging implemented for sensitive operations
   - [ ] RLS policies tested for proper access control

- [ ] Error messages don’t expose sensitive information
- [ ] User input validation prevents security vulnerabilities

Code Quality Standards

TypeScript/JavaScript Standards:

```typescript
// Healthcare-specific interfaces with clear data classification
interface PatientData {
  id: string;
  // PII - requires audit logging for access
  fullName: string;
  phoneNumber: string;

  // Medical data - requires specialist access controls
  allergies?: string[];
  chronicConditions?: MedicalCondition[];
}

// Required audit logging for sensitive operations
async function accessPatientData(patientId: string, accessor: User): Promise<PatientData> {
  await auditLog.log({
    action: 'patient_data_access',
    actorId: accessor.id,
    subjectId: patientId,
    reason: 'clinical_care',
    timestamp: new Date()
  });

  return await database.patients.findById(patientId);
}
```

Database Standards:

```sql
-- All healthcare tables must have RLS enabled
ALTER TABLE patient_records ENABLE ROW LEVEL SECURITY;

-- Example RLS policy with healthcare access controls
CREATE POLICY patient_own_records ON patient_records
  FOR ALL TO authenticated
  USING (patient_id = current_user_id());

-- Audit trigger for sensitive tables
CREATE TRIGGER audit_patient_records_changes
  AFTER INSERT OR UPDATE OR DELETE ON patient_records
  FOR EACH ROW EXECUTE FUNCTION audit_log_changes();
```

Testing Standards

Unit Test Requirements:

```
describe('Prescription QR Code Generation', () => {
  it('should generate unique QR codes for each prescription', async () => {
    // Test healthcare-specific business logic
    const prescription1 = await createPrescription(patientId, gpId, medications);
    const prescription2 = await createPrescription(patientId, gpId, medications);

    expect(prescription1.qrCode).not.toBe(prescription2.qrCode);
    expect(prescription1.qrCode).toMatch(/^[A-Z0-9]{12}$/);
  });

  it('should disable QR code when PDF is downloaded', async () => {
    const prescription = await createPrescription(patientId, gpId, medications);

    await downloadPrescriptionPDF(prescription.id);

    const updated = await getPrescription(prescription.id);
    expect(updated.qrEnabled).toBe(false);
    expect(updated.pdfDownloadedAt).toBeDefined();
  });
});
```

Trunk vs. GitFlow Recommendation

Recommendation: **Modified GitHub Flow** for healthcare development

Rationale:
- **Simplicity**: Reduced complexity important for 3-month timeline
- **Safety**: All changes go through main branch with comprehensive testing
- **Compliance**: Clear audit trail of all changes for healthcare regulations
- **Rapid Iteration**: Shorter feedback cycles important for MVP development

Implementation:

```
main (always production-ready)
├── feature branches (short-lived, <1 week)
├── hotfix branches (for critical production issues)
└── release branches (only for final stabilization)
```

Healthcare-Specific Modifications:
- **Mandatory Reviews**: All healthcare data handling requires medical director sign-off
- **Compliance Gates**: Automated checks for PII handling and audit logging
- **Security Validation**: Required security review for authentication/authorization changes
- **Rollback Capability**: All deployments must include tested rollback procedures

Vendor & Licensing Plan

Software Development Kits (SDKs)

SDK/Service	License Type	Cost	Purpose	Compliance Notes
React Native	MIT	Free	Mobile app development	Open source, no restrictions
Supabase	Apache 2.0 + Commercial	$25/month	Database, auth, real-time	PostgreSQL HIPAA-compliant configuration
WhatsApp Cloud API	Commercial	$0.005-0.02/ message	OTP delivery, video calls	Meta’s terms require healthcare compliance review
Google Maps API	Commercial	$0.50/1000 requests	Pharmacy location services	Standard commercial terms
Sentry	BSD 3-Clause + Commercial	$26/month	Error tracking, performance	Data residency compliance available

Third-Party APIs & Services

Healthcare-Critical Integrations:
- **WhatsApp Cloud API**: Patient communication channel
- **SLA Requirements**: 99.9% uptime, <5 second message delivery
- **Compliance**: Review terms for healthcare data transmission
- **Backup Plan**: SMS gateway as fallback (implementation in Phase 2)

- **Google Maps Platform**: Pharmacy discovery and navigation
- **Usage Estimates**: ~10,000 requests/month for 1000 active users
- **Data Privacy**: Location data handling per Google’s healthcare terms
- **Alternative**: OpenStreetMap + Mapbox for cost optimization if needed

Development & Operations:
- **GitHub**: Source code management and CI/CD
- **Security**: Advanced security features for dependency scanning
- **Compliance**: Audit logs for all code changes and deployments

- **DataDog**: Application monitoring and alerting
- **Healthcare Configuration**: HIPAA-compliant data handling
- **Alerting**: Integration with PagerDuty for incident response

Font & Asset Licensing

Typography:
- **System Fonts**: iOS San Francisco, Android Roboto (license-free)

- **Custom Fonts**: Inter font family (SIL Open Font License)
- **Icon Library**: Heroicons (MIT License) + custom healthcare icons

Media Assets:
- **Stock Photography**: Unsplash/Pexels for marketing materials (free commercial use)
- **Medical Icons**: Custom-designed icons for prescription, consultation, etc.
- **Illustration Library**: Healthcare-specific illustrations (custom or licensed)

Licensing Compliance Management

Dependency Tracking:
- **Tool**: GitHub Dependabot + license scanning
- **Policy**: No GPL or copyleft licenses in production code
- **Review Process**: Legal review required for any commercial license changes

Healthcare Data Licensing:
- **Patient Consent**: Clear licensing of patient data for platform use
- **Partner Agreements**: Data sharing agreements with pharmacies and labs
- **Regulatory Compliance**: Kenya Medical Board licensing requirements

Open Source Contributions:
- **Policy**: Contributions to open source must not expose proprietary healthcare logic
- **Review Process**: Tech Lead approval required for any external contributions
- **Attribution**: Proper attribution for all open source dependencies

---

H) Environments, Tooling, and CI/CD

Environment Strategy

Environment Architecture

```
graph TB
    subgraph "Local Development"
        LocalApp[React Native App]
        LocalWeb[Next.js Portals]
        LocalDB[Local PostgreSQL]
        MockServices[Mock External APIs]
    end

    subgraph "Development Environment"
        DevAPI[Supabase Dev Instance]
        DevStorage[R2 Dev Bucket]
        DevWhatsApp[WhatsApp Sandbox]
    end

    subgraph "Staging Environment"
        StageAPI[Supabase Staging]
        StageStorage[R2 Staging Bucket]
        StageWhatsApp[WhatsApp Test Account]
        StageMonitoring[Sentry Staging]
    end

    subgraph "Production Environment"
        ProdAPI[Supabase Production]
        ProdStorage[R2 Production Bucket]
        ProdWhatsApp[WhatsApp Business API]
        ProdMonitoring[Sentry + DataDog]
    end

    LocalApp --> DevAPI
    LocalWeb --> DevAPI

    DevAPI --> StageAPI
    StageAPI --> ProdAPI

    DevWhatsApp --> StageWhatsApp
    StageWhatsApp --> ProdWhatsApp
```

Environment Configuration

Local Development Environment

- **Purpose**: Individual developer productivity and unit testing
- **Data**: Synthetic test data with realistic healthcare scenarios
- **External Services**: Mocked APIs for offline development
- **Security**: Relaxed authentication for development speed
- **Monitoring**: Console logging only

Development Environment (Shared)

- **Purpose**: Integration testing and feature validation
- **Data**: Shared synthetic dataset refreshed weekly
- **External Services**: Sandbox APIs (WhatsApp, Maps test keys)
- **Security**: Basic authentication, audit logging enabled
- **Monitoring**: Basic error tracking via Sentry

Staging Environment
- **Purpose**: Production-like testing and stakeholder demos
- **Data**: Production-like synthetic data (no real patient information)
- **External Services**: Production API keys with rate limiting
- **Security**: Full security model, RLS policies enforced
- **Monitoring**: Complete monitoring stack (Sentry + DataDog)

Production Environment
- **Purpose**: Live healthcare service delivery
- **Data**: Real patient data with full compliance controls
- **External Services**: Production APIs with SLA monitoring
- **Security**: Maximum security, full audit logging, compliance controls
- **Monitoring**: Real-time alerting, incident response integration

Promotion Policy & Data Strategy

Environment Promotion Pipeline

```
Promotion Flow:
  local → development → staging → production

Promotion Triggers:
  - local → development: Pull request merge to main
  - development → staging: Successful integration tests + manual approval
  - staging → production: Release candidate approval + go/no-go decision

Rollback Policy:
  - Automated rollback on health check failures
  - Manual rollback capability within 5 minutes
  - Database migration rollback procedures tested
```

Seed Data Strategy

Local Development Data:

```sql
-- Synthetic patient data for development
INSERT INTO users (id, role, phone_e164, full_name) VALUES
  ('550e8400-e29b-41d4-a716-446655440001', 'patient', '+254700123001', 'Test Patient One'),
  ('550e8400-e29b-41d4-a716-446655440002', 'gp', '+254700123002', 'Dr. Test GP'),
  ('550e8400-e29b-41d4-a716-446655440003', 'specialist', '+254700123003', 'Dr. Test Specialist');

-- Test consultations with realistic medical scenarios
INSERT INTO consultations (patient_id, gp_id, chief_complaint, status) VALUES
  ('550e8400-e29b-41d4-a716-446655440001', '550e8400-e29b-41d4-a716-446655440002', 'Persistent headaches for 3 days', 'completed');

-- Sample prescriptions for QR code testing
INSERT INTO prescriptions (consultation_id, patient_id, prescriber_id, items_json, qr_code) VALUES
  ((SELECT id FROM consultations LIMIT 1),
   '550e8400-e29b-41d4-a716-446655440001',
   '550e8400-e29b-41d4-a716-446655440002',
   '[{"medication": "Paracetamol 500mg", "quantity": 20, "instructions": "Take 1 tablet every 6 hours"}]',
   'TEST123QR456');
```

Staging Data Refresh:
- **Frequency**: Weekly refresh with new synthetic data
- **Volume**: 1000 patients, 100 GPs, 50 specialists, 10000 historical consultations
- **Realism**: Based on anonymized patterns from healthcare partner feedback
- **Privacy**: Zero real patient data, all synthetic but medically realistic

Production Data Backup:
- **Frequency**: Continuous incremental backup, daily full backup
- **Retention**: 30 days online, 24 months archive for prescriptions
- **Recovery Testing**: Monthly restore tests to staging environment
- **Encryption**: All backups encrypted with managed keys

CI/CD Pipeline Architecture

Pipeline Stages Overview

```
graph LR
    A[Code Push] --> B[Lint & Format]
    B --> C[Unit Tests]
    C --> D[Security Scan]
    D --> E[Build Artifacts]
    E --> F[Integration Tests]
    F --> G[Deploy to Dev]
    G --> H[E2E Tests]
    H --> I[Security Tests]
    I --> J[Performance Tests]
    J --> K[Deploy to Staging]
    K --> L[Staging Validation]
    L --> M{Manual Approval}
    M -->|Approved| N[Deploy to Production]
    M -->|Rejected| O[Rollback]
    N --> P[Production Health Check]
    P --> Q[Monitoring Alert Setup]
```

Detailed Pipeline Configuration

Stage 1: Code Quality & Security (2-3 minutes)

```yaml
name: Code Quality Pipeline
on:
  pull_request:
    branches: [main]
  push:
    branches: [main]

jobs:
  lint-and-format:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Install dependencies
        run: npm ci

      - name: ESLint with healthcare rules
        run: npm run lint:healthcare

      - name: TypeScript type checking
        run: npm run type-check

      - name: Prettier formatting
        run: npm run format:check

  security-scan:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Run Snyk security scan
        uses: snyk/actions/node@master
        env:
          SNYK_TOKEN: ${{ secrets.SNYK_TOKEN }}
        with:
          args: --severity-threshold=high

      - name: Healthcare-specific security checks
        run: npm run security:healthcare-audit
```

Stage 2: Testing & Validation (5-8 minutes)

```yaml
  unit-tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Install dependencies
        run: npm ci

      - name: Run unit tests with coverage
        run: npm run test:coverage
        env:
          COVERAGE_THRESHOLD_STATEMENTS: 70
          COVERAGE_THRESHOLD_HEALTHCARE: 90

      - name: Upload coverage to CodeCov
        uses: codecov/codecov-action@v3

  integration-tests:
    runs-on: ubuntu-latest
    needs: unit-tests
    services:
      postgres:
        image: postgres:14
        env:
          POSTGRES_PASSWORD: postgres
          POSTGRES_DB: mediconnect_test
        options: >-
          --health-cmd pg_isready
          --health-interval 10s
          --health-timeout 5s
          --health-retries 5

    steps:
      - uses: actions/checkout@v3
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Run database migrations
        run: npm run db:migrate:test
        env:
          DATABASE_URL: postgresql://postgres:postgres@localhost:5432/mediconnect_test

      - name: Run RLS policy tests
        run: npm run test:rls-policies

      - name: Run API integration tests
        run: npm run test:integration
        env:
          WHATSAPP_API_KEY: ${{ secrets.WHATSAPP_TEST_API_KEY }}
          SUPABASE_ANON_KEY: ${{ secrets.SUPABASE_ANON_KEY_TEST }}
```

Stage 3: Build & Artifact Creation (3-5 minutes)

```yaml
  build-mobile:
    runs-on: ubuntu-latest
    needs: [lint-and-format, security-scan, integration-tests]
    steps:
      - uses: actions/checkout@v3

      - name: Setup React Native environment
        uses: react-native-community/setup-react-native@v1
        with:
          node-version: '18'

      - name: Install dependencies
        run: npm ci

      - name: Build Android APK
        run: |
          cd android
          ./gradlew assembleRelease
        env:
          ANDROID_SIGNING_KEY: ${{ secrets.ANDROID_SIGNING_KEY }}

      - name: Upload APK artifact
        uses: actions/upload-artifact@v3
        with:
          name: android-apk
          path: android/app/build/outputs/apk/release/

  build-web-portals:
    runs-on: ubuntu-latest
    needs: [lint-and-format, security-scan, integration-tests]
    steps:
      - uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Build Next.js portals
        run: |
          npm ci
          npm run build:portals
        env:
          NEXT_PUBLIC_SUPABASE_URL: ${{ secrets.SUPABASE_URL_STAGING }}
          NEXT_PUBLIC_SUPABASE_ANON_KEY: ${{ secrets.SUPABASE_ANON_KEY_STAGING }}

      - name: Upload portal artifacts
        uses: actions/upload-artifact@v3
        with:
          name: web-portals
          path: .next/
```

Stage 4: Deployment & E2E Testing (10-15 minutes)

```
  deploy-staging:
    runs-on: ubuntu-latest
    needs: [build-mobile, build-web-portals]
    environment: staging
    steps:
      - uses: actions/checkout@v3

      - name: Deploy to Vercel (Portals)
        uses: amondnet/vercel-action@v20
        with:
          vercel-token: ${{ secrets.VERCEL_TOKEN }}
          vercel-args: '--prod'
          vercel-org-id: ${{ secrets.VERCEL_ORG_ID }}
          vercel-project-id: ${{ secrets.VERCEL_PROJECT_ID }}
          working-directory: ./

      - name: Update Supabase Edge Functions
        run: |
          npx supabase functions deploy --project-ref $
{{ secrets.SUPABASE_PROJECT_REF }}
        env:
          SUPABASE_ACCESS_TOKEN: ${{ secrets.SUPABASE_ACCESS_TOKEN }}

      - name: Run database migrations
        run: npm run db:migrate:staging
        env:
          DATABASE_URL: ${{ secrets.STAGING_DATABASE_URL }}

      - name: Health check staging deployment
        run: npm run health-check:staging

  e2e-tests:
    runs-on: ubuntu-latest
    needs: deploy-staging
    steps:
      - uses: actions/checkout@v3

      - name: Setup Playwright
        uses: microsoft/playwright-github-action@v1

      - name: Run E2E tests
        run: npm run test:e2e:staging
        env:
          E2E_BASE_URL: ${{ secrets.STAGING_BASE_URL }}
          E2E_TEST_USER_PHONE: ${{ secrets.E2E_TEST_PHONE }}
          E2E_WHATSAPP_API_KEY: ${{ secrets.WHATSAPP_TEST_API_KEY }}

      - name: Upload E2E test results
        uses: actions/upload-artifact@v3
        if: failure()
        with:
          name: e2e-test-results
          path: test-results/
```

Stage 5: Production Deployment (5-10 minutes)

```yaml
  deploy-production:
    runs-on: ubuntu-latest
    needs: [e2e-tests]
    environment: production
    if: github.ref == 'refs/heads/main'
    steps:
      - uses: actions/checkout@v3

      - name: Manual deployment approval
        uses: trstringer/manual-approval@v1
        with:
          secret: ${{ github.TOKEN }}
          approvers: product-manager,tech-lead,medical-director
          minimum-approvals: 2

      - name: Deploy to production
        run: |
          # Deploy web portals
          npx vercel --prod --token ${{ secrets.VERCEL_TOKEN_PROD }}

          # Deploy Supabase functions
          npx supabase functions deploy --project-ref $
{{ secrets.SUPABASE_PROJECT_REF_PROD }}

          # Run production migrations (if any)
          npm run db:migrate:production
        env:
          SUPABASE_ACCESS_TOKEN: ${{ secrets.SUPABASE_ACCESS_TOKEN_PROD }}
          DATABASE_URL: ${{ secrets.PRODUCTION_DATABASE_URL }}

      - name: Production health check
        run: npm run health-check:production
        env:
          PROD_BASE_URL: ${{ secrets.PRODUCTION_BASE_URL }}

      - name: Setup production monitoring
        run: npm run monitoring:setup-alerts
        env:
          SENTRY_DSN: ${{ secrets.SENTRY_DSN_PROD }}
          DATADOG_API_KEY: ${{ secrets.DATADOG_API_KEY }}

      - name: Notify deployment success
        uses: 8398a7/action-slack@v3
        with:
          status: success
          text: 'Mediconnect production deployment successful! 🚀'
        env:
          SLACK_WEBHOOK_URL: ${{ secrets.SLACK_WEBHOOK }}
```

Infrastructure-as-Code Layout

Terraform Configuration Structure

```
infrastructure/
├── environments/
│   ├── development/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   └── terraform.tfvars
│   ├── staging/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   └── terraform.tfvars
│   └── production/
│       ├── main.tf
│       ├── variables.tf
│       └── terraform.tfvars
├── modules/
│   ├── supabase/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   └── outputs.tf
│   ├── cloudflare/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   └── outputs.tf
│   └── monitoring/
│       ├── main.tf
│       ├── variables.tf
│       └── outputs.tf
└── shared/
    ├── backend.tf
    └── providers.tf
```

Supabase Configuration Module

```
# modules/supabase/main.tf
resource "supabase_project" "main" {
  organization_id = var.organization_id
  name            = "${var.project_name}-${var.environment}"
  database_password = var.database_password
  region          = "ap-southeast-1"  # Singapore for Kenya proximity

  # Healthcare-specific configuration
  settings = {
    # Enable Row Level Security by default
    row_level_security = true

    # Enable audit logging
    audit_logging = true

    # Healthcare data retention
    backup_retention_days = var.environment == "production" ? 90 : 7
  }
}

# RLS policies deployment
resource "supabase_sql" "rls_policies" {
  project_ref = supabase_project.main.reference_id

  sql = file("${path.module}/sql/rls_policies.sql")

  depends_on = [supabase_project.main]
}

# Healthcare-specific database functions
resource "supabase_sql" "healthcare_functions" {
  project_ref = supabase_project.main.reference_id

  sql = file("${path.module}/sql/healthcare_functions.sql")

  depends_on = [supabase_sql.rls_policies]
}
```

Cloudflare R2 Configuration

```
# modules/cloudflare/main.tf
resource "cloudflare_r2_bucket" "healthcare_storage" {
  account_id = var.cloudflare_account_id
  name       = "mediconnect-${var.environment}-healthcare-data"
  location   = "auto"
}

# Bucket lifecycle for healthcare data retention
resource "cloudflare_r2_bucket_lifecycle" "healthcare_retention" {
  bucket_name = cloudflare_r2_bucket.healthcare_storage.name
  account_id  = var.cloudflare_account_id

  rules {
    id     = "prescription-retention"
    status = "enabled"

    filter {
      prefix = "prescriptions/"
    }

    transition {
      days          = 90
      storage_class = "INFREQUENT_ACCESS"
    }

    expiration {
      days = 730  # 24 months for prescriptions
    }
  }

  rules {
    id     = "chat-retention"
    status = "enabled"

    filter {
      prefix = "chat-media/"
    }

    expiration {
      days = 90  # 90 days for chat media
    }
  }
}
```

Secrets Management

Secret Categories & Access Patterns

Development Secrets (Low Security)
- Database connection strings for development/staging
- Test API keys for external services
- Non-production encryption keys
- **Storage**: GitHub repository secrets
- **Access**: All developers during development

Production Secrets (High Security)
- Production database credentials
- WhatsApp Business API production keys

- Encryption keys for healthcare data
- Third-party service production credentials
- **Storage**: Separate production secret store
- **Access**: Limited to DevOps and approved deployment processes

Healthcare Compliance Secrets (Maximum Security)
- Patient data encryption keys
- Audit logging access keys
- Compliance reporting credentials
- **Storage**: Hardware Security Module (HSM) or equivalent
- **Access**: Audited access with business justification required

Secret Rotation Policy

```
Secret Rotation Schedule:
  Database Passwords:
    - Frequency: Every 90 days
    - Process: Automated with zero-downtime rotation
    - Validation: Connection health checks after rotation

  API Keys (WhatsApp, Maps):
    - Frequency: Every 180 days
    - Process: Manual rotation with testing
    - Validation: Integration test suite execution

  Encryption Keys:
    - Frequency: Every 365 days
    - Process: Key versioning with backward compatibility
    - Validation: Data decrypt/encrypt verification

  Healthcare Audit Keys:
    - Frequency: Every 30 days
    - Process: Automated with compliance approval
    - Validation: Audit log accessibility verification
```

Secret Access Audit

```sql
-- Example audit logging for secret access
CREATE TABLE secret_access_audit (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  secret_name VARCHAR(255) NOT NULL,
  accessor_user_id UUID REFERENCES users(id),
  access_reason TEXT NOT NULL,
  access_timestamp TIMESTAMPTZ DEFAULT NOW(),
  ip_address INET,
  success BOOLEAN NOT NULL
);

-- Required audit logging for healthcare secret access
CREATE OR REPLACE FUNCTION log_secret_access(
  p_secret_name TEXT,
  p_accessor_id UUID,
  p_reason TEXT,
  p_success BOOLEAN
) RETURNS VOID AS $$
BEGIN
  INSERT INTO secret_access_audit
    (secret_name, accessor_user_id, access_reason, success)
  VALUES
    (p_secret_name, p_accessor_id, p_reason, p_success);
END;
$$ LANGUAGE plpgsql SECURITY DEFINER;
```

Feature Flags & Configuration Strategy

Feature Flag Architecture

```
// Feature flag configuration with healthcare safety controls
interface FeatureFlag {
  name: string;
  enabled: boolean;
  healthcareImpact: 'none' | 'low' | 'medium' | 'high' | 'critical';
  rolloutPercentage: number;
  enabledForRoles: UserRole[];
  safeguards: {
    requiresApproval: boolean;
    rollbackTriggers: string[];
    monitoringAlerts: boolean;
  };
}

const featureFlags: FeatureFlag[] = [
  {
    name: 'ai_intake_v2',
    enabled: false,
    healthcareImpact: 'medium',
    rolloutPercentage: 0,
    enabledForRoles: ['patient'],
    safeguards: {
      requiresApproval: true,
      rollbackTriggers: ['error_rate_>5%', 'completion_rate_<80%'],
      monitoringAlerts: true
    }
  },
  {
    name: 'prescription_qr_v2',
    enabled: false,
    healthcareImpact: 'high',
    rolloutPercentage: 0,
    enabledForRoles: ['patient', 'pharmacy_admin'],
    safeguards: {
      requiresApproval: true,
      rollbackTriggers: ['qr_scan_failure_>2%', 'security_alert'],
      monitoringAlerts: true
    }
  },
  {
    name: 'specialist_video_calls',
    enabled: true,
    healthcareImpact: 'low',
    rolloutPercentage: 100,
    enabledForRoles: ['specialist', 'patient'],
    safeguards: {
      requiresApproval: false,
      rollbackTriggers: ['call_failure_rate_>10%'],
      monitoringAlerts: true
    }
  }
];
```

Configuration Management Strategy

Environment-Specific Configuration:

```typescript
// Configuration with healthcare-specific validation
interface AppConfig {
  database: {
    url: string;
    maxConnections: number;
    auditLogging: boolean;
  };
  healthcare: {
    consultationTimeoutMinutes: number;
    prescriptionExpiryDays: number;
    auditRetentionMonths: number;
    piiMaskingEnabled: boolean;
  };
  externalServices: {
    whatsapp: {
      apiKey: string;
      webhookSecret: string;
      rateLimitPerHour: number;
    };
    maps: {
      apiKey: string;
      maxRequestsPerDay: number;
    };
  };
}

// Configuration validation for healthcare compliance
function validateConfig(config: AppConfig): void {
  if (!config.healthcare.auditRetentionMonths || config.healthcare.auditRetentionMonths < 24) {
    throw new Error('Healthcare audit retention must be at least 24 months');
  }

  if (!config.database.auditLogging) {
    throw new Error('Database audit logging is required for healthcare compliance');
  }

  if (!config.healthcare.piiMaskingEnabled && process.env.NODE_ENV === 'production') {
    throw new Error('PII masking must be enabled in production');
  }
}
```

Feature Flag Safety Controls:

```
class HealthcareFeatureFlags {
  async enableFeature(flagName: string, userId: string): Promise<boolean> {
    const flag = this.getFeatureFlag(flagName);

    if (flag.healthcareImpact === 'critical') {
      // Critical healthcare features require medical director approval
      const approval = await this.getMedicalDirectorApproval(flagName, userId);
      if (!approval) {
        await this.auditLog.log({
          action: 'feature_flag_denied',
          flagName,
          userId,
          reason: 'missing_medical_director_approval'
        });
        return false;
      }
    }

    if (flag.rolloutPercentage < 100) {
      // Gradual rollout with monitoring
      const userHash = this.hashUserId(userId);
      const shouldEnable = userHash % 100 < flag.rolloutPercentage;

      if (shouldEnable) {
        await this.setupMonitoringAlerts(flagName, userId);
      }

      return shouldEnable;
    }

    return flag.enabled;
  }

  async rollbackFeature(flagName: string, reason: string): Promise<void> {
    const flag = this.getFeatureFlag(flagName);

    // Immediate rollback for healthcare-critical features
    if (flag.healthcareImpact === 'critical' || flag.healthcareImpact === 'high') {
      await this.disableFeatureImmediately(flagName);
      await this.notifyMedicalTeam(flagName, reason);
    }

    await this.auditLog.log({
      action: 'feature_flag_rollback',
      flagName,
      reason,
      timestamp: new Date()
    });
  }
}
```

I) Launch, Rollout & Post-Launch

Rollout Strategy

Phased Launch Approach

Phase 1: Internal Alpha (Week 13, Days 1-3)
- **Scope**: 20 internal team members and healthcare advisors
- **Purpose**: Production system validation and immediate issue detection
- **Duration**: 72 hours with 24/7 monitoring
- **Success Criteria**: Zero critical bugs, all user journeys functional
- **Rollback Trigger**: Any P0 incident or data security issue

Phase 2: Limited Beta (Week 13, Days 4-5)
- **Scope**: 50 external beta users (healthcare workers and their families)
- **Purpose**: Real-world workflow validation and user feedback collection
- **Duration**: 48 hours with close user support
- **Success Criteria**: >80% task completion rate, positive user feedback
- **Rollback Trigger**: User safety concerns or system instability

Phase 3: Controlled Launch (Week 13, Days 6-7)
- **Scope**: 500 general users with invitation-only access
- **Purpose**: Validate system capacity and operational processes
- **Duration**: 48 hours with full operational support
- **Success Criteria**: System performance within SLA, support processes working
- **Rollback Trigger**: Performance degradation or support overwhelm

Canary Deployment Configuration

```
# Canary deployment strategy for healthcare safety
Canary Strategy:
  initial_percentage: 1%
  increment_percentage: 5%
  increment_interval: 2 hours
  max_percentage: 50%

  # Healthcare-specific safety checks
  health_checks:
    - endpoint: /health/database
      timeout: 5s
      interval: 30s
    - endpoint: /health/whatsapp-integration
      timeout: 10s
      interval: 60s
    - endpoint: /health/prescription-system
      timeout: 5s
      interval: 30s

  # Automatic rollback triggers
  rollback_triggers:
    - error_rate > 2%
    - response_time_p95 > 1000ms
    - health_check_failures > 3
    - manual_trigger: true

  # Healthcare compliance checks
  compliance_gates:
    - audit_logging_functional: true
    - rls_policies_enforced: true
    - pii_masking_active: true
```

Blue-Green Deployment Process

```
graph TB
    subgraph "Blue Environment (Current Production)"
        BlueApp[Current App Version]
        BlueDB[Production Database]
        BlueMonitor[Production Monitoring]
    end

    subgraph "Green Environment (New Version)"
        GreenApp[New App Version]
        GreenDB[Database Migrations]
        GreenMonitor[Monitoring Setup]
    end

    subgraph "Load Balancer"
        LB[Traffic Router]
        HealthCheck[Health Checks]
    end

    subgraph "Rollback Safety"
        RollbackTrigger[Automated Rollback]
        ManualSwitch[Manual Override]
    end

    LB --> BlueApp
    LB -.->|Gradual Switch| GreenApp
    HealthCheck --> RollbackTrigger
    ManualSwitch --> LB

    BlueDB <--> GreenDB
```

Blue-Green Deployment Steps:

1. **Green Environment Preparation**: Deploy new version to green environment with database migrations
2. **Health Validation**: Run comprehensive health checks on green environment
3. **Traffic Switching**: Gradually route traffic from blue to green (10% → 25% → 50% → 100%)
4. **Monitoring**: Real-time monitoring of all healthcare metrics and user satisfaction
5. **Rollback Capability**: Instant traffic routing back to blue if issues detected

Staged Percentage Rollout

User Segmentation Strategy

```
interface RolloutSegment {
  name: string;
  percentage: number;
  criteria: UserCriteria;
  healthcareRisk: 'low' | 'medium' | 'high';
  rollbackPlan: string;
}

const rolloutSegments: RolloutSegment[] = [
  {
    name: 'healthcare_professionals',
    percentage: 5,
    criteria: { roles: ['gp', 'specialist'], experience: 'any' },
    healthcareRisk: 'low',
    rollbackPlan: 'immediate_fallback_to_previous_portal'
  },
  {
    name: 'tech_savvy_patients',
    percentage: 10,
    criteria: { role: 'patient', appUsageHigh: true, location: 'nairobi_central' },
    healthcareRisk: 'medium',
    rollbackPlan: 'disable_new_features_keep_basic_access'
  },
  {
    name: 'pharmacy_partners',
    percentage: 15,
    criteria: { roles: ['pharmacy_admin'], verification: 'verified' },
    healthcareRisk: 'medium',
    rollbackPlan: 'fallback_to_manual_prescription_processing'
  },
  {
    name: 'general_population',
    percentage: 70,
    criteria: { role: 'patient', location: 'nairobi', registrationDate: 'after_beta' }
,
    healthcareRisk: 'high',
    rollbackPlan: 'full_system_rollback_with_user_communication'
  }
];
```

Rollout Timeline & Monitoring

Week 1: Foundation Users (5%)

- Healthcare professionals and internal partners
- Intensive monitoring with 15-minute check-ins
- Direct communication channels for immediate feedback
- Success metrics: 100% system availability, <1% error rate

Week 2: Early Adopters (15%)

- Tech-savvy patients and verified pharmacy partners
- Daily monitoring reports and user satisfaction surveys
- Weekly feedback sessions with user representatives
- Success metrics: 99.5% availability, >85% task completion

Week 3: Broad Rollout (50%)
- General patient population in target Nairobi areas
- Standard monitoring with automated alerting
- Community support channels and FAQ resources
- Success metrics: All SLA targets met, positive user feedback

Week 4: Full Launch (100%)
- Complete user base access with waitlist clearing
- Production monitoring with business-as-usual processes
- Ongoing user support and continuous improvement
- Success metrics: Sustained performance, user growth targets

Rollback Playbook

Automated Rollback Triggers

```
# Automated rollback configuration
Rollback Triggers:
  critical_errors:
    - prescription_qr_generation_failure_rate > 5%
    - authentication_failure_rate > 10%
    - database_connection_loss > 30_seconds
    - whatsapp_integration_down > 5_minutes

  performance_degradation:
    - api_response_time_p95 > 2000ms for 5_minutes
    - chat_message_delivery_delay > 30_seconds
    - prescription_pdf_generation_timeout > 60_seconds

  healthcare_safety:
    - audit_logging_failure detected
    - rls_policy_violation detected
    - pii_exposure_incident triggered
    - medical_director_manual_override

  user_experience:
    - user_completion_rate < 60% for 1_hour
    - support_ticket_spike > 300% normal_volume
    - user_satisfaction_score < 3.0/5.0
```

Manual Rollback Procedures

Immediate Rollback (P0 - Execute within 5 minutes):

```bash
#!/bin/bash
# Emergency rollback script for critical healthcare issues

# 1. Immediate traffic routing to previous version
kubectl patch service mediconnect-api -p '{"spec":{"selector":{"version":"previous"}}}'

# 2. Database connection switch to read replica (if needed)
export DATABASE_URL=$PREVIOUS_DATABASE_URL
kubectl set env deployment/mediconnect-api DATABASE_URL=$DATABASE_URL

# 3. Disable new feature flags immediately
curl -X POST "$FEATURE_FLAG_API/disable-all-new-features" \
  -H "Authorization: Bearer $EMERGENCY_TOKEN"

# 4. Activate incident response
curl -X POST "$PAGERDUTY_API/incidents" \
  -H "Authorization: Token token=$PAGERDUTY_TOKEN" \
  -d '{"incident":{"type":"incident","title":"Mediconnect Emergency Rollback","service":{"id":"'"$SERVICE_ID"'","type":"service_reference"}}}'

# 5. Send user communication
python scripts/emergency_user_notification.py --template="rollback" --urgency="high"

echo "Emergency rollback completed. Incident response activated."
```

Planned Rollback (P1-P2 - Execute within 1 hour):
1. **Assessment Phase** (15 minutes):
- Gather metrics on issue impact and user feedback
- Consult with medical director on patient safety implications
- Evaluate whether partial rollback (feature flags) sufficient

1. **Communication Phase** (15 minutes):
   - Notify users via in-app notification about planned maintenance
   - Alert healthcare partners about potential service changes
   - Prepare customer service team with FAQ responses

2. **Rollback Execution** (15 minutes):
   - Execute graduated rollback starting with most affected user segments
   - Monitor system stability at each rollback step
   - Validate that previous version functionality restored completely

3. **Validation Phase** (15 minutes):
   - Run automated test suite against rolled-back system
   - Verify all critical healthcare workflows functioning
   - Confirm user satisfaction metrics return to baseline

Rollback Communication Templates

Critical Rollback User Notification:

```
🚨 MEDICONNECT SERVICE UPDATE

We've temporarily switched back to our previous system version to ensure your
healthcare experience remains safe and reliable.

✅ All consultations, prescriptions, and appointments remain secure
✅ Your health data is protected and accessible
✅ Healthcare partners can continue providing services

We're working to resolve the issue quickly. Updates: mediconnect.help/status

Questions? Contact support at +254-XXX-XXXX

- The Mediconnect Team
```

Analytics & Instrumentation Plan

Events Schema Design

```
// Healthcare-specific analytics events with privacy protection
interface AnalyticsEvent {
  event_id: string;
  event_name: string;
  user_id_hash: string;  // Hashed for privacy
  user_role: UserRole;
  timestamp: Date;
  properties: Record<string, any>;
  healthcare_context?: {
    consultation_type?: 'gp' | 'specialist';
    prescription_involved?: boolean;
    emergency_flag?: boolean;
  };
  privacy_level: 'public' | 'internal' | 'healthcare_only';
}

// Key events for healthcare analytics
const healthcareEvents = {
  // Authentication & Onboarding
  'user_registration_started': {
    properties: ['language_selected', 'referral_source'],
    privacy_level: 'internal'
  },
  'whatsapp_otp_sent': {
    properties: ['delivery_success', 'attempt_count'],
    privacy_level: 'internal'
  },
  'authentication_completed': {
    properties: ['time_to_verify', 'device_type'],
    privacy_level: 'internal'
  },

  // Healthcare Journey Events
  'ai_intake_started': {
    properties: ['symptoms_category', 'urgency_level'],
    privacy_level: 'healthcare_only'
  },
  'gp_consultation_requested': {
    properties: ['wait_time_minutes', 'gp_assigned_id_hash'],
    healthcare_context: { consultation_type: 'gp' },
    privacy_level: 'healthcare_only'
  },
  'consultation_completed': {
    properties: ['duration_minutes', 'prescription_issued', 'referral_made'],
    healthcare_context: { consultation_type: 'gp', prescription_involved: true },
    privacy_level: 'healthcare_only'
  },
  'prescription_fulfilled': {
    properties: ['fulfillment_method', 'pharmacy_id_hash',
'time_to_fulfillment_hours'],
    healthcare_context: { prescription_involved: true },
    privacy_level: 'healthcare_only'
  }
};
```

Dashboard Questions & KPIs

Executive Dashboard Questions:
1. **How many patients completed their healthcare journey this week?**
- Metric: End-to-end completion rate (intake → consultation → resolution)
- Target: >85% completion rate
- Alert: <80% for 24 hours triggers investigation

1. **Are our healthcare providers efficiently serving patients?**
   - Metric: Average GP consultation time and patient satisfaction
   - Target: 15±2 minutes, >4.2/5 satisfaction
   - Alert: >20 minutes average or <4.0 satisfaction

2. **Is our prescription system secure and reliable?**
   - Metric: QR code success rate and security incident count
   - Target: >98% QR success, 0 security incidents
   - Alert: <95% success rate or any security incident

Operational Dashboard Questions:
1. **Which parts of our system need immediate attention?**
- Metrics: API error rates, database performance, external service uptime
- Targets: <1% error rate, <500ms P75 response time, >99% uptime
- Alerts: Real-time notifications for threshold breaches

1. **How are our healthcare partners performing?**
   - Metrics: Pharmacy claim processing time, specialist response rates
   - Targets: <30 min claim processing, >90% specialist acceptance
   - Alerts: Performance degradation affecting patient care

2. **What's the user experience quality across different touchpoints?**
   - Metrics: App crash rates, chat delivery times, video call success
   - Targets: <0.1% crash rate, <2s message delivery, >95% call success
   - Alerts: User experience degradation patterns

Analytics Implementation Architecture

```
graph TB
    subgraph "Data Collection"
        MobileApp[Mobile App Events]
        WebPortal[Web Portal Events]
        Backend[Backend System Events]
        Healthcare[Healthcare Workflow Events]
    end

    subgraph "Event Processing"
        EventStream[Real-time Event Stream]
        DataValidation[Healthcare Privacy Validation]
        EventEnrichment[Context Enrichment]
    end

    subgraph "Storage & Analysis"
        AnalyticsDB[(Analytics Database)]
        DataWarehouse[(Data Warehouse)]
        ComplianceStore[(Compliance Audit Store)]
    end

    subgraph "Visualization & Alerting"
        ExecutiveDashboard[Executive Dashboard]
        OperationalDashboard[Operational Dashboard]
        HealthcareDashboard[Healthcare Quality Dashboard]
        RealTimeAlerts[Real-time Alerts]
    end

    MobileApp --> EventStream
    WebPortal --> EventStream
    Backend --> EventStream
    Healthcare --> EventStream

    EventStream --> DataValidation
    DataValidation --> EventEnrichment
    EventEnrichment --> AnalyticsDB
    EventEnrichment --> ComplianceStore

    AnalyticsDB --> DataWarehouse
    DataWarehouse --> ExecutiveDashboard
    DataWarehouse --> OperationalDashboard
    DataWarehouse --> HealthcareDashboard

    EventStream --> RealTimeAlerts
```

Privacy-Safe Analytics Implementation:

```
class HealthcareAnalytics {
  async trackEvent(event: AnalyticsEvent): Promise<void> {
    // Healthcare privacy validation
    if (event.privacy_level === 'healthcare_only') {
      event.user_id_hash = this.hashUserId(event.user_id_hash);
      event.properties = this.sanitizeHealthcareData(event.properties);
    }

    // Remove PII from all analytics events
    const sanitizedEvent = this.removePII(event);

    // Route to appropriate storage based on sensitivity
    if (event.healthcare_context) {
      await this.complianceStore.store(sanitizedEvent);
    }

    await this.analyticsStore.store(sanitizedEvent);

    // Real-time alerting for critical events
    if (this.isCriticalHealthcareEvent(event)) {
      await this.alertingService.notify(event);
    }
  }

  private sanitizeHealthcareData(properties: Record<string, any>): Record<string,
any> {
    const sanitized = { ...properties };

    // Remove potential PII patterns
    Object.keys(sanitized).forEach(key => {
      if (this.containsPII(sanitized[key])) {
        sanitized[key] = '[REDACTED]';
      }
    });

    return sanitized;
  }

  private isCriticalHealthcareEvent(event: AnalyticsEvent): boolean {
    const criticalEvents = [
      'prescription_security_violation',
      'unauthorized_pii_access',
      'consultation_emergency_escalation',
      'system_downtime_healthcare_impact'
    ];

    return criticalEvents.includes(event.event_name) ||
           event.healthcare_context?.emergency_flag === true;
  }
}
```

J) Appendices & Templates

ADR Template

```markdown
# ADR-XXX: [Title]

**Status**: [Proposed | Accepted | Deprecated | Superseded]
**Date**: YYYY-MM-DD
**Deciders**: [List of decision makers]
**Technical Story**: [GitHub issue/ticket reference]

## Context

[Describe the problem space and the factors that influenced the decision. Include any
constraints, assumptions, or business requirements specific to healthcare.]

### Healthcare-Specific Context
- **Patient Safety Impact**: [None | Low | Medium | High | Critical]
- **Data Privacy Considerations**: [Description of PII/PHI handling requirements]
- **Regulatory Implications**: [Any healthcare compliance factors]
- **Clinical Workflow Impact**: [How this affects healthcare providers]

## Decision

[Clearly state the architectural decision made]

### Implementation Details
- **Technical Approach**: [Specific implementation approach]
- **Integration Points**: [How this integrates with existing systems]
- **Security Measures**: [Security considerations and implementations]
- **Healthcare Safeguards**: [Any specific healthcare safety measures]

## Alternatives Considered

| Alternative | Pros | Cons | Healthcare Impact | Decision Rationale |
|-------------|------|------|-------------------|-------------------|
| Option A | [Pros] | [Cons] | [Impact] | [Why not chosen] |
| Option B | [Pros] | [Cons] | [Impact] | [Why not chosen] |

## Consequences

### Positive Consequences
- [List positive outcomes expected from this decision]
- [Include any improvements to patient care or provider workflow]

### Negative Consequences
- [List potential drawbacks or challenges]
- [Include any risks to healthcare delivery]

### Risks & Mitigations
- **Risk**: [Identified risk]
  - **Mitigation**: [How risk is addressed]
  - **Owner**: [Who is responsible for mitigation]

### Rollback Strategy
- **Rollback Triggers**: [Conditions that would trigger rollback]
- **Rollback Process**: [Steps to reverse this decision]
- **Data Migration**: [How to handle data if rollback needed]

## Compliance & Audit Considerations

- **Audit Logging Requirements**: [What must be logged]
- **Data Retention Impact**: [How this affects data retention policies]
- **Third-Party Dependencies**: [Any new external dependencies]
- **Regulatory Documentation**: [Documentation requirements]
```

```
## Follow-up Actions

- [ ] [Action item 1 with owner and due date]
- [ ] [Action item 2 with owner and due date]
- [ ] [Monitoring/alerting setup with owner]
- [ ] [Documentation updates with owner]

## Notes

[Any additional notes, links to related ADRs, or follow-up decisions needed]
```

Risk Register Template

```markdown
# Mediconnect Risk Register

**Document Version**: 1.0
**Last Updated**: YYYY-MM-DD
**Next Review**: YYYY-MM-DD
**Owner**: Project Manager

## Risk Assessment Scale

### Probability Scale (1-5)
1. **Rare** (0-5%): May occur only in exceptional circumstances
2. **Unlikely** (5-25%): Could occur at some time
3. **Possible** (25-50%): Might occur at some time
4. **Likely** (50-75%): Will probably occur in most circumstances
5. **Almost Certain** (75-100%): Expected to occur in most circumstances

### Impact Scale (1-5)
1. **Insignificant**: Minor inconvenience, no patient safety impact
2. **Minor**: Small impact on operations, minimal patient experience degradation
3. **Moderate**: Noticeable impact on service delivery, patient experience affected
4. **Major**: Significant impact on healthcare delivery, potential patient safety concerns
5. **Catastrophic**: Severe impact on patient safety, regulatory violations, system failure

### Risk Matrix
| Impact/Probability | Rare (1) | Unlikely (2) | Possible (3) | Likely (4) | Almost Certain (5) |
|-------------------|----------|-------------|-------------|------------|-------------------|
| **Catastrophic (5)** | 🟡 5 | 🟡 10 | 🔴 15 | 🔴 20 | 🔴 25 |
| **Major (4)** | 🟢 4 | 🟡 8 | 🟡 12 | 🔴 16 | 🔴 20 |
| **Moderate (3)** | 🟢 3 | 🟢 6 | 🟡 9 | 🟡 12 | 🔴 15 |
| **Minor (2)** | 🟢 2 | 🟢 4 | 🟢 6 | 🟡 8 | 🟡 10 |
| **Insignificant (1)** | 🟢 1 | 🟢 2 | 🟢 3 | 🟢 4 | 🟡 5 |

🔴 **High Risk (15-25)**: Immediate action required
🟡 **Medium Risk (5-12)**: Action plan required
🟢 **Low Risk (1-4)**: Monitor and review

## Current Risks

### HIGH RISKS (Immediate Action Required)

#### RISK-001: WhatsApp API Service Disruption
- **Category**: External Dependency
- **Probability**: 3 (Possible)
- **Impact**: 5 (Catastrophic)
- **Risk Score**: 15 🔴
- **Description**: WhatsApp Cloud API outage or rate limiting affecting patient authentication and video consultations
- **Patient Safety Impact**: HIGH - Patients cannot access healthcare services
- **Early Warning Indicators**:
  - WhatsApp message delivery rates drop below 95%
  - Authentication failure rates increase above 5%
  - Support tickets mentioning WhatsApp issues spike
- **Mitigation Strategies**:
  - **Primary**: Implement SMS fallback for authentication
  - **Secondary**: Partner with WhatsApp Business API partner for SLA guarantees
  - **Tertiary**: Develop in-app video calling capability for future release
- **Contingency Plan**:
  - Activate SMS gateway within 30 minutes of detection
  - Communicate service changes to users via in-app notifications
  - Escalate to WhatsApp Business support through enterprise channel
- **Owner**: Technical Lead
- **Review Date**: Weekly
- **Last Updated**: YYYY-MM-DD

#### RISK-002: Insufficient Healthcare Partner Capacity
- **Category**: Operational Capacity
- **Probability**: 4 (Likely)
- **Impact**: 4 (Major)
- **Risk Score**: 16 🔴
- **Description**: Inadequate number of GPs, specialists, or verified pharmacies to
meet user demand during scaling
- **Patient Safety Impact**: HIGH - Delayed healthcare access
- **Early Warning Indicators**:
  - Average GP response time exceeds 45 minutes
  - Pharmacy claim processing time exceeds 2 hours
  - User completion rate drops below 70%
  - Partner utilization rates exceed 90%
- **Mitigation Strategies**:
  - **Primary**: Accelerate partner onboarding with dedicated recruitment team
  - **Secondary**: Implement dynamic pricing to incentivize partner participation
  - **Tertiary**: Establish overflow partnerships with telemedicine providers
- **Contingency Plan**:
  - Activate waitlist system for new users
  - Extend GP consultation hours through shift scheduling
  - Partner with existing telehealth services for overflow capacity
- **Owner**: Operations Manager
- **Review Date**: Daily during launch phase
- **Last Updated**: YYYY-MM-DD

### MEDIUM RISKS (Action Plan Required)

#### RISK-003: Database Performance Under Load
- **Category**: Technical Infrastructure
- **Probability**: 3 (Possible)
- **Impact**: 3 (Moderate)
- **Risk Score**: 9 🟡
- **Description**: PostgreSQL database performance degradation affecting API response
times and user experience
- **Patient Safety Impact**: MEDIUM - Delayed access to health records
- **Early Warning Indicators**:
  - Database query P95 latency exceeds 200ms
  - API response times consistently above 800ms
  - Database connection pool exhaustion alerts
- **Mitigation Strategies**:
  - **Primary**: Implement read replicas for query distribution
  - **Secondary**: Optimize database indexes for healthcare queries
  - **Tertiary**: Database sharding strategy for user data
- **Contingency Plan**:
  - Scale up database resources immediately
  - Enable query caching for read-heavy operations
  - Activate read-only mode for non-critical features
- **Owner**: Database Administrator
- **Review Date**: Weekly
- **Last Updated**: YYYY-MM-DD

#### RISK-004: Regulatory Compliance Gaps
- **Category**: Legal/Regulatory
- **Probability**: 2 (Unlikely)
- **Impact**: 5 (Catastrophic)
- **Risk Score**: 10 🟡
- **Description**: Failure to meet Kenya Medical Board or data protection requirements
- **Patient Safety Impact**: HIGH - Platform shutdown risk
- **Early Warning Indicators**:
  - Audit findings indicating non-compliance
  - Changes in Kenya healthcare regulations
  - Data handling practices questioned by regulators
- **Mitigation Strategies**:
  - **Primary**: Engage healthcare law expertise for compliance review
  - **Secondary**: Implement comprehensive audit logging
  - **Tertiary**: Establish relationship with Kenya Medical Board
- **Contingency Plan**:
  - Immediate compliance remediation with legal counsel
  - Suspend operations if required pending compliance
  - Engage regulatory affairs consultant
- **Owner**: Legal Counsel / Medical Director
- **Review Date**: Monthly
- **Last Updated**: YYYY-MM-DD

### LOW RISKS (Monitor and Review)

#### RISK-005: Team Knowledge Gaps
- **Category**: Human Resources
- **Probability**: 3 (Possible)
- **Impact**: 2 (Minor)
- **Risk Score**: 6 🟢
- **Description**: Limited healthcare domain expertise within development team
- **Patient Safety Impact**: LOW - Potential workflow inefficiencies
- **Mitigation Strategies**:
  - **Primary**: Healthcare domain training for development team
  - **Secondary**: Medical advisor embedded in development process
  - **Tertiary**: Healthcare workflow documentation and review processes
- **Owner**: Engineering Manager
- **Review Date**: Monthly

## Risk Monitoring & Review Process

### Weekly Risk Review (High & Medium Risks)
- **Participants**: Project Manager, Technical Lead, Medical Director
- **Duration**: 30 minutes
- **Agenda**: Review risk indicators, assess mitigation effectiveness, identify new risks
- **Outputs**: Updated risk scores, new action items, escalation decisions

### Monthly Comprehensive Review (All Risks)
- **Participants**: Full project team and stakeholders
- **Duration**: 60 minutes
- **Agenda**: Complete risk register review, lessons learned, risk appetite assessment
- **Outputs**: Updated risk register, new risk identification, mitigation plan updates

### Emergency Risk Review (Triggered by Events)
- **Triggers**: Any high-risk event occurrence, major system incident, regulatory changes
- **Response Time**: Within 2 hours of trigger event
- **Participants**: Project Manager, affected risk owners, escalation stakeholders
- **Outputs**: Immediate action plan, communication strategy, risk score updates

## Risk Escalation Matrix

| Risk Score | Escalation Level | Notification Timeline | Approval Required |
|------------|-----------------|----------------------|-------------------|
| 1-4 (Low) | Team Level | Weekly reporting | None |
| 5-12 (Medium) | Management Level | 24 hours | Project Manager |
| 13-25 (High) | Executive Level | 2 hours | CEO/CTO |
| Catastrophic Event | Board Level | Immediate | Board Chairman |
---

## Change Log

| Date | Version | Changes | Author |
|------|---------|---------|---------|
| YYYY-MM-DD | 1.0 | Initial risk register creation | Project Manager |
| | | | |
```

RAID Log Template

```
# RAID Log - Mediconnect MVP

**Document Version**: 1.0
**Project**: Mediconnect Healthcare Platform MVP
**Last Updated**: YYYY-MM-DD
**Owner**: Project Manager

## How to Use This RAID Log

**RAID** stands for:
- **R**isks: Uncertain events that could impact the project
- **A**ssumptions: Things we believe to be true but haven't verified
- **I**ssues: Problems that are currently happening
- **D**ependencies: Things the project relies on to be successful

Each item should be:
- **Specific**: Clear description of what it is
- **Actionable**: Clear next steps identified
- **Owned**: Someone responsible for managing it
- **Tracked**: Regular status updates

---

## 🚨 RISKS

### RISK-01: WhatsApp API Service Disruption
- **Status**: 🔴 **Active**
- **Impact**: High - Could prevent patient access to healthcare services
- **Probability**: Medium (25-50%)
- **Description**: WhatsApp Cloud API outage or rate limiting affecting authentication and video calls
- **Mitigation Actions**:
  - [ ] Implement SMS fallback authentication (Owner: Backend Dev, Due: Week 8)
  - [ ] Establish direct support channel with WhatsApp Business (Owner: Ops Manager, Due: Week 6)
  - [ ] Create monitoring dashboards for WhatsApp integration health (Owner: DevOps, Due: Week 4)
- **Contingency Plan**: SMS gateway activation within 30 minutes
- **Owner**: Technical Lead
- **Last Updated**: YYYY-MM-DD

### RISK-02: Insufficient Partner Network Capacity
- **Status**: 🟡 **Monitoring**
- **Impact**: High - Delayed healthcare access for patients
- **Probability**: High (50-75%)
- **Description**: Not enough GPs, specialists, or pharmacies to handle user scale-up
- **Mitigation Actions**:
  - [ ] Partner recruitment campaign (Owner: Ops Manager, Due: Week 2)
  - [ ] Capacity monitoring dashboard (Owner: Data Analyst, Due: Week 4)
  - [ ] Overflow partner agreements (Owner: Business Dev, Due: Week 6)
- **Owner**: Operations Manager
- **Last Updated**: YYYY-MM-DD

---

## 📋 ASSUMPTIONS

### ASSUMP-01: WhatsApp Usage in Target Market
- **Status**: ✅ **Validated**
- **Assumption**: 90%+ of target users have WhatsApp and are comfortable with video calls
- **Validation Method**: User surveys and market research
- **Result**: 94% of surveyed users have WhatsApp, 87% comfortable with video calls
- **Impact if False**: Would require alternative communication channels
- **Owner**: Product Manager
- **Validation Date**: YYYY-MM-DD

### ASSUMP-02: Kenyan Regulatory Environment
- **Status**: 🟡 **Partially Validated**
- **Assumption**: Current Kenya telehealth regulations allow our proposed service model
- **Validation Method**: Legal review with healthcare law expert
- **Current Status**: Initial review positive, awaiting formal regulatory opinion
- **Risk if False**: Platform might need significant modifications or licensing
- **Actions**:
  - [ ] Complete formal legal review (Owner: Legal Counsel, Due: Week 3)
  - [ ] Engage with Kenya Medical Board (Owner: Medical Director, Due: Week 4)
- **Owner**: Medical Director
- **Last Updated**: YYYY-MM-DD

### ASSUMP-03: Healthcare Partner Willingness
- **Status**: 🟡 **Needs Validation**
- **Assumption**: Pharmacies and labs will join our verified network under proposed terms
- **Validation Method**: Direct outreach and pilot partnerships
- **Current Status**: Initial interest from 5 pharmacies, 2 labs
- **Target**: 20+ pharmacies, 10+ labs by launch
- **Actions**:
  - [ ] Formalize partnership terms (Owner: Legal, Due: Week 2)
  - [ ] Begin partner onboarding process (Owner: Ops Manager, Due: Week 3)
- **Owner**: Operations Manager
- **Last Updated**: YYYY-MM-DD

---

## 🔥 ISSUES

### ISSUE-01: Database Schema Conflicts
- **Status**: 🔴 **Open**
- **Priority**: High
- **Description**: RLS policies from different documents show conflicting access patterns
- **Impact**: Delayed development, potential security gaps
- **Root Cause**: Multiple document versions without proper consolidation
- **Actions**:
  - [ ] Schema reconciliation meeting (Owner: Tech Lead, Due: This Week)
  - [ ] Create single authoritative schema document (Owner: Backend Dev, Due: Next Week)
  - [ ] Update all dependent documentation (Owner: Technical Writer, Due: Week After Next)
- **Blocking**: Backend development, Security testing
- **Owner**: Technical Lead
- **Opened Date**: YYYY-MM-DD
- **Target Resolution**: YYYY-MM-DD

### ISSUE-02: Conflicting Timeline Information
- **Status**: 🟡 **In Progress**
- **Priority**: Medium
- **Description**: Different documents reference different project timelines (3 months vs end of year)
- **Impact**: Resource planning uncertainty, stakeholder confusion
- **Root Cause**: Documentation created at different times without updates
- **Actions**:
  - [ ] Stakeholder alignment meeting (Owner: PM, Due: This Week)
  - [ ] Official timeline confirmation (Owner: PM, Due: This Week)
  - [ ] Update all project documentation (Owner: Technical Writer, Due: Next Week)
- **Owner**: Project Manager
- **Opened Date**: YYYY-MM-DD
- **Target Resolution**: YYYY-MM-DD

---

## 🔗 DEPENDENCIES

### DEP-01: WhatsApp Cloud API Access
- **Status**: ✅ **Resolved**
- **Type**: External
- **Description**: Need WhatsApp Business API access for OTP and video functionality
- **Dependent On**: Meta/WhatsApp approval process
- **Impact**: Core authentication and consultation features
- **Actions Completed**:
  - ✅ WhatsApp Business account setup
  - ✅ API access approval received
  - ✅ Initial integration testing completed
- **Owner**: Technical Lead
- **Resolution Date**: YYYY-MM-DD

### DEP-02: Healthcare Partner Contracts
- **Status**: 🟡 **In Progress**
- **Type**: Legal/Business
- **Description**: Partnership agreements with pharmacies, labs, and specialists
- **Dependent On**: Legal review and partner negotiations
- **Impact**: Core service delivery capabilities
- **Current Status**: 5 of 20 pharmacy agreements in progress
- **Actions**:
  - [ ] Complete template partnership agreement (Owner: Legal, Due: Week 2)
  - [ ] Execute agreements with priority partners (Owner: Ops Manager, Due: Week 4)
- **Owner**: Operations Manager
- **Target Date**: YYYY-MM-DD

### DEP-03: Kenya Medical Board Compliance
- **Status**: 🟡 **Pending**
- **Type**: Regulatory
- **Description**: Confirmation of compliance with Kenyan healthcare regulations
- **Dependent On**: Medical board review and approval
- **Impact**: Platform launch approval
- **Actions**:
  - [ ] Submit compliance documentation (Owner: Medical Director, Due: Week 3)
  - [ ] Schedule review meeting with medical board (Owner: Medical Director, Due: Week
4)
- **Owner**: Medical Director
- **Target Date**: YYYY-MM-DD

### DEP-04: Development Team Onboarding
- **Status**: 🟡 **In Progress**
- **Type**: Internal Resource
- **Description**: Full development team hired and productive
- **Dependent On**: HR recruitment and technical onboarding
- **Impact**: Development velocity and timeline
- **Current Status**: 3 of 5 developers onboarded
- **Actions**:
  - [ ] Complete final 2 developer hires (Owner: Engineering Manager, Due: Week 1)
  - [ ] Technical onboarding completion (Owner: Tech Lead, Due: Week 2)
- **Owner**: Engineering Manager
- **Target Date**: YYYY-MM-DD

---
```

```
## 📊 RAID Summary Dashboard

### Current Status Overview
| Category | Total | 🔴 High Priority | 🟡 Medium Priority | 🟢 Low Priority | ✅ Resolved |
|----------|-------|-----------------|------------------|---------------|------------|
| **Risks** | 5 | 2 | 2 | 1 | 0 |
| **Assumptions** | 6 | 0 | 3 | 2 | 1 |
| **Issues** | 4 | 1 | 2 | 0 | 1 |
| **Dependencies** | 8 | 0 | 3 | 2 | 3 |

### Action Items Due This Week
- [ ] Schema reconciliation meeting (ISSUE-01) - Tech Lead
- [ ] Stakeholder timeline alignment (ISSUE-02) - Project Manager
- [ ] Complete SMS fallback design (RISK-01) - Backend Dev
- [ ] Partner contract templates (DEP-02) - Legal Counsel

### Escalation Required
- **ISSUE-01**: Database schema conflicts blocking development
- **RISK-01**: WhatsApp dependency needs executive attention for business continuity

---

## 🔄 Review Schedule

### Daily Standup (RAID Updates)
- **When**: Every morning at 9 AM
- **Duration**: 5 minutes for RAID updates
- **Who**: All team members
- **Focus**: New issues, resolved items, blocker escalation

### Weekly RAID Review
- **When**: Friday 3 PM - 4 PM
- **Who**: Project Manager, Team Leads, Key Stakeholders
- **Agenda**: Full RAID review, priority updates, risk assessment
- **Outputs**: Updated RAID log, escalation decisions, action assignments

### Monthly Executive RAID Report
- **When**: Last Friday of each month
- **Who**: Project Manager presents to Executive Team
- **Format**: Summary dashboard + top 5 critical items
- **Outputs**: Strategic decisions, resource allocation, risk appetite changes

---

**Next Review Date**: YYYY-MM-DD
**Review Owner**: Project Manager
**Distribution**: All project stakeholders, executive team
```

Stakeholder Map Template

```
# Stakeholder Map - Mediconnect Healthcare Platform

**Project**: Mediconnect MVP Development and Launch
**Document Version**: 1.0
**Last Updated**: YYYY-MM-DD
**Owner**: Project Manager

## Stakeholder Analysis Framework

### Power/Interest Grid
```

High Power, High Interest | High Power, Low Interest

MANAGE CLOSELY | KEEP SATISFIED

|

---------------------------+---------------------------

|

Low Power, High Interest | Low Power, Low Interest

KEEP INFORMED | MONITOR

```
### Engagement Levels
- **Unaware**: Don't know about the project
- **Resistant**: Aware but opposed to project
- **Neutral**: Aware but neither supportive nor resistant
- **Supportive**: Aware and supportive of project
- **Leading**: Aware and actively driving project success

---

## PRIMARY STAKEHOLDERS (High Power, High Interest - Manage Closely)

### CEO/Founder
- **Name**: [Name]
- **Role**: Chief Executive Officer
- **Power Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Interest Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Current Engagement**: Leading
- **Key Concerns**:
  - Overall business success and market penetration
  - Regulatory compliance and risk management
  - Fundraising and investor relations
  - Public health impact and company mission
- **Success Criteria**:
  - Platform launch on time and budget
  - Achievement of user growth targets (10k MAU)
  - Positive healthcare outcomes and patient satisfaction
  - Regulatory approval and compliance maintenance
- **Communication Preferences**:
  - Weekly executive briefings (Friday, 30 min)
  - Immediate escalation for high-risk issues
  - Monthly board presentation materials
- **Influence Tactics**:
  - Direct authority over resource allocation
  - Final decision maker on strategic pivots
  - External stakeholder relationships (regulators, investors)
- **Engagement Actions**:
  - Weekly 1:1 with Project Manager
  - Quarterly steering committee meetings
  - Direct communication channel for urgent issues
- **Project Manager Notes**: Requires high-level summaries focused on business impact

---

### Medical Director
- **Name**: [Name]
- **Role**: Chief Medical Officer / Clinical Lead
- **Power Level**: ⭐⭐⭐⭐⭐ (Maximum - Healthcare Decisions)
- **Interest Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Current Engagement**: Leading
- **Key Concerns**:
  - Patient safety and clinical outcomes
  - Healthcare provider workflow efficiency
  - Medical liability and professional standards
  - Clinical feature design and usability
- **Success Criteria**:
  - Zero patient safety incidents
  - High healthcare provider satisfaction (>4.5/5)
  - Efficient clinical workflows (15-min GP consultations)
  - Successful regulatory approvals
- **Communication Preferences**:
  - Bi-weekly clinical review meetings
  - Real-time alerts for safety-related issues
  - Direct input on healthcare features
- **Influence Tactics**:
  - Veto power over clinical features
  - Direct relationships with healthcare partners
  - Regulatory expertise and connections
- **Engagement Actions**:
  - Healthcare feature design reviews
  - Clinical workflow validation sessions
  - Partner relationship management
- **Project Manager Notes**: Must approve all healthcare-related decisions

---

### Technical Lead/CTO
- **Name**: [Name]
- **Role**: Chief Technology Officer / Technical Lead
- **Power Level**: ⭐⭐⭐⭐ (High)
- **Interest Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Current Engagement**: Leading
- **Key Concerns**:
  - Technical architecture scalability and security
  - Development team productivity and morale
  - System reliability and performance
  - Technical debt management
- **Success Criteria**:
  - System meets all SLA requirements (99.5% uptime)
  - Security audit passes with no critical issues
  - Development team velocity is stable
  - Technical architecture supports future scaling
- **Communication Preferences**:
  - Daily technical standups
  - Weekly architecture reviews
  - Immediate escalation for system issues
- **Engagement Actions**:
  - Technical decision review and approval
  - Development team mentoring and support
  - Security and compliance oversight
- **Project Manager Notes**: Key partner for all technical decisions and risk management

---

## SECONDARY STAKEHOLDERS (Keep Satisfied/Keep Informed)

### Product Manager
- **Power Level**: ⭐⭐⭐ (Medium-High)
- **Interest Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Category**: Keep Informed (High Interest, Medium Power)
- **Key Concerns**: User experience, feature prioritization, market fit
- **Engagement**: Weekly product planning sessions, user feedback reviews

### Operations Manager
- **Power Level**: ⭐⭐⭐ (Medium-High)
- **Interest Level**: ⭐⭐⭐⭐ (High)
- **Category**: Keep Satisfied (High Power, Medium Interest)
- **Key Concerns**: Partner onboarding, operational processes, cost management
- **Engagement**: Bi-weekly operational reviews, partner relationship updates

### Legal Counsel
- **Power Level**: ⭐⭐⭐⭐ (High - Compliance Decisions)
- **Interest Level**: ⭐⭐⭐ (Medium)
- **Category**: Keep Satisfied (High Power, Medium Interest)
- **Key Concerns**: Regulatory compliance, contract negotiations, liability management
- **Engagement**: Monthly compliance reviews, contract approval processes

### Healthcare Partners (GPs, Specialists, Pharmacies)
- **Power Level**: ⭐⭐⭐ (Medium-High - Collective)
- **Interest Level**: ⭐⭐⭐⭐ (High)
- **Category**: Keep Informed (High Interest, Medium Individual Power)
- **Key Concerns**: Workflow efficiency, compensation, patient volume
- **Engagement**: Quarterly partner meetings, ongoing support and training

### Development Team
- **Power Level**: ⭐⭐ (Medium - Collective)
- **Interest Level**: ⭐⭐⭐⭐ (High)
- **Category**: Keep Informed (High Interest, Medium Power)
- **Key Concerns**: Technical challenges, work-life balance, career growth
- **Engagement**: Daily standups, sprint reviews, monthly team retrospectives

---

## EXTERNAL STAKEHOLDERS

### Kenya Medical Board
- **Power Level**: ⭐⭐⭐⭐⭐ (Maximum - Regulatory Authority)
- **Interest Level**: ⭐⭐ (Low-Medium)
- **Category**: Keep Satisfied (High Power, Low Interest)
- **Key Concerns**: Patient safety, professional standards, regulatory compliance
- **Success Criteria**: Platform approved for operation, no compliance violations
- **Engagement Strategy**:
  - Formal regulatory submissions
  - Scheduled compliance meetings
  - Proactive communication on policy changes
- **Risk**: Could shut down platform if non-compliant

### Target Patients (Early Adopters)
- **Power Level**: ⭐⭐⭐ (Medium - Collective)
- **Interest Level**: ⭐⭐⭐⭐⭐ (Maximum)
- **Category**: Keep Informed (High Interest, Medium Power)
- **Key Concerns**: Healthcare access, cost, quality of care, data privacy
- **Success Criteria**: Platform meets healthcare needs, positive user experience
- **Engagement Strategy**:
  - User research and feedback sessions
  - Beta testing programs
  - Community engagement via social media

### Investors/Board Members
- **Power Level**: ⭐⭐⭐⭐⭐ (Maximum - Funding Decisions)
- **Interest Level**: ⭐⭐⭐⭐ (High)
- **Category**: Manage Closely (High Power, High Interest)
- **Key Concerns**: Return on investment, growth potential, market penetration
- **Success Criteria**: User growth targets, revenue milestones, market expansion
- **Engagement Strategy**:
  - Monthly investor updates
  - Quarterly board meetings
  - Key milestone communications

---

## STAKEHOLDER COMMUNICATION MATRIX

| Stakeholder Group | Communication Method | Frequency | Key Information | Owner |
|-------------------|---------------------|-----------|-----------------|-------|
| **CEO/Founder** | Executive briefing + 1:1 | Weekly | Business metrics, risks, key decisions | Project Manager |
| **Medical Director** | Clinical review meeting | Bi-weekly | Healthcare outcomes, sa
```

```
fety, compliance | Project Manager |
| **Technical Lead** | Architecture review | Weekly | Technical progress, risks, decisions | Technical Lead |
| **Product Manager** | Product planning session | Weekly | Feature progress, user feedback | Product Manager |
| **Operations Manager** | Operational review | Bi-weekly | Partner status, processes, costs | Ops Manager |
| **Legal Counsel** | Compliance review | Monthly | Regulatory status, contracts, risks | Legal Lead |
| **Healthcare Partners** | Partner newsletter + calls | Monthly | Platform updates, training, feedback | Ops Manager |
| **Development Team** | Sprint ceremonies | Daily/Bi-weekly | Technical progress, blockers, planning | Technical Lead |
| **Kenya Medical Board** | Formal submissions | As required | Compliance documentation, approvals | Medical Director |
| **Investors/Board** | Board report + presentation | Monthly/Quarterly | Business progress, financials, strategy | CEO |

---

## RESISTANCE MANAGEMENT PLAN

### Potential Resistance Sources

#### Healthcare Providers (GPs/Specialists)
- **Potential Concerns**:
  - Technology adoption challenges
  - Workflow disruption during transition
  - Compensation and patient volume concerns
- **Mitigation Strategies**:
  - Comprehensive training and support programs
  - Phased rollout with early adopter incentives
  - Clear compensation structure and volume guarantees
- **Engagement Tactics**:
  - Provider advisory council
  - Regular feedback sessions and workflow improvements
  - Success story sharing and peer networking

#### Traditional Healthcare Facilities
- **Potential Concerns**:
  - Competition from telemedicine platform
  - Patient volume reduction
  - Disruption of existing referral networks
- **Mitigation Strategies**:
  - Position as complement, not replacement
  - Partnership opportunities for complex cases
  - Clear scope limitations (primary care focus)

#### Regulatory Bodies
- **Potential Concerns**:
  - Patient safety in telemedicine delivery
  - Data privacy and security compliance
  - Professional liability and oversight
- **Mitigation Strategies**:
  - Proactive compliance demonstration
  - Regular safety audits and reporting
  - Collaborative approach to regulation development

---

## ENGAGEMENT SUCCESS METRICS

### Primary Stakeholders
```

```
- **CEO Satisfaction**: Weekly briefing attendance, strategic decision speed
- **Medical Director Engagement**: Clinical feature approval rate, safety incident response
- **Technical Lead Partnership**: Architecture decision alignment, development velocity

### Secondary Stakeholders
- **Healthcare Partner Adoption**: Partner onboarding rate, platform utilization
- **Legal Compliance**: Regulatory approval timeline, contract negotiation success
- **Team Morale**: Development velocity, retention rate, satisfaction surveys

### External Stakeholders
- **Regulatory Approval**: Compliance milestones met, no enforcement actions
- **Patient Satisfaction**: User retention rate, NPS score, completion rates
- **Investor Confidence**: Funding milestones, board meeting feedback

---

**Next Review**: YYYY-MM-DD
**Review Process**: Monthly stakeholder analysis update
**Escalation**: Project Manager → CEO for high-power stakeholder issues
```

Estimation Guide Template

```
# Estimation Guide - Mediconnect Healthcare Platform

**Document Version**: 1.0
**Project**: Mediconnect MVP Development
**Last Updated**: YYYY-MM-DD
**Owner**: Technical Lead

## Estimation Philosophy

### Healthcare Development Principles
- **Safety First**: Healthcare features require additional testing and validation time
- **Compliance Overhead**: Regulatory requirements add 20-30% to development estimates
- **Integration Complexity**: Healthcare systems integration is inherently complex
- **Data Security**: Security implementations require thorough testing and review

### Story Point Scale (Modified Fibonacci)
We use a modified Fibonacci sequence: **1, 2, 3, 5, 8, 13, 21, 40**

---

## STORY POINT REFERENCE GUIDE

### 1 Point - Trivial (1-2 hours)
**Characteristics**:
- Very simple changes with no complexity
- No healthcare data involved
- No external integrations
- Well-understood problem with clear solution
- No testing beyond unit tests required

**Healthcare Examples**:
- Update text label on non-clinical screens
- Add new language translation for existing feature
- Simple configuration changes
- Minor CSS/styling adjustments

**Non-Healthcare Examples**:
- Fix typo in documentation
- Update environment variable
- Simple logging addition

---

### 2 Points - Simple (Half day)
**Characteristics**:
- Simple feature with minimal complexity
- Limited healthcare data interaction
- Well-defined acceptance criteria
- Single developer can complete independently
- Standard unit testing sufficient

**Healthcare Examples**:
- Add new field to patient profile (non-clinical)
- Simple validation rule for healthcare form
- Basic error message improvement
- Simple notification text customization

**Technical Examples**:
- Add new API endpoint with CRUD operations
- Simple database migration (add column)
- Basic component styling changes
- Simple utility function creation
```

```
---

### 3 Points - Straightforward (1 day)
**Characteristics**:
- Moderate complexity with some unknowns
- Healthcare data handling with basic privacy requirements
- May require coordination with another team member
- Integration testing required
- Clear path to completion

**Healthcare Examples**:
- Add new non-critical field to prescription display
- Implement basic search functionality for healthcare providers
- Simple reporting feature for operational metrics
- Basic patient communication template

**Technical Examples**:
- API endpoint with complex validation
- Database schema changes with data migration
- Component with multiple states and interactions
- Integration with well-documented external API

---

### 5 Points - Moderate (2-3 days)
**Characteristics**:
- Significant complexity or unknowns
- Healthcare data with moderate privacy/security requirements
- Cross-team coordination required
- Multiple components or systems affected
- Requires design decisions during implementation

**Healthcare Examples**:
- Implement prescription QR code generation
- Add new step to AI intake questionnaire
- Patient data export functionality
- Basic healthcare provider portal screen

**Technical Examples**:
- Complex API integration with authentication
- Database performance optimization
- React Native screen with complex user interactions
- Security feature implementation

---

### 8 Points - Complex (1 week)
**Characteristics**:
- High complexity with significant unknowns
- Critical healthcare data with strict security requirements
- Multiple team coordination required
- Affects core system functionality
- Requires research and design work
- Comprehensive testing including security validation

**Healthcare Examples**:
- Complete AI intake workflow implementation
- WhatsApp video call integration
- Row-level security policy implementation
- Healthcare provider authentication system

**Technical Examples**:
```

```markdown
- Real-time chat system implementation
- Complex data migration with validation
- Performance optimization affecting multiple systems
- Third-party integration with complex authentication

---

### 13 Points - Very Complex (1.5-2 weeks)
**Characteristics**:
- Very high complexity with major unknowns
- Mission-critical healthcare functionality
- Requires extensive research and planning
- Multiple systems integration
- Comprehensive testing including end-to-end scenarios
- May require multiple iterations

**Healthcare Examples**:
- Complete prescription fulfillment workflow
- Healthcare partner portal with role-based access
- Comprehensive audit logging system
- GP consultation workflow with video integration

**Technical Examples**:
- Database architecture overhaul
- Security system implementation
- Complex external system integration
- Performance optimization across entire stack

---

### 21 Points - Epic Level (3-4 weeks)
**Characteristics**:
- Extremely complex with extensive unknowns
- Foundational healthcare system functionality
- Requires significant research, design, and planning
- Affects multiple systems and teams
- Requires phased implementation approach
- Extensive testing and validation required

**Healthcare Examples**:
- Complete patient onboarding and authentication system
- End-to-end specialist referral workflow
- Comprehensive healthcare data model implementation
- Full compliance and audit system

**Technical Examples**:
- Complete authentication and authorization system
- Full CI/CD pipeline implementation
- Comprehensive monitoring and alerting system
- Major architecture migration

---

### 40 Points - Too Large (Break Down Required)
**When to Use**: Never estimate this high - story needs to be broken down into smaller
components

**Action Required**:
- Break into smaller user stories
- Identify dependencies and sequence
- Create epic with multiple stories
- Re-estimate component pieces
```

```
---

## HEALTHCARE-SPECIFIC ESTIMATION FACTORS

### Security & Compliance Multiplier
Add additional time for healthcare-specific requirements:
```

Base Estimate × Compliance Factor = Final Estimate

Compliance Factors:
- No healthcare data: 1.0x
- Basic patient data: 1.2x
- Healthcare records/prescriptions: 1.4x
- Financial healthcare data: 1.6x
- Critical safety systems: 2.0x

```
### Integration Complexity
Healthcare integrations are inherently more complex:
```

Integration Complexity Factors:
- Internal API only: 1.0x
- Simple external API: 1.3x
- Healthcare partner API: 1.5x
- Regulatory system integration: 2.0x
- Multiple healthcare system integration: 2.5x

```
### Testing Requirements
Healthcare features require additional testing:
```

Testing Multipliers:
- Standard feature: 1.0x
- Healthcare workflow: 1.3x
- Patient safety feature: 1.5x
- Security/privacy feature: 1.7x
- Regulatory compliance feature: 2.0x

```
---

## ESTIMATION PROCESS

### Planning Poker Sessions
**Frequency**: During sprint planning and backlog refinement
**Participants**: Product Manager, Technical Lead, 2-3 Developers, Medical Advisor (for healthcare features)
**Duration**: 15 minutes per story maximum

### Estimation Steps
1. **Story Reading**: Product Manager reads story and acceptance criteria
2. **Clarification**: Team asks questions about requirements and scope
3. **Healthcare Impact**: Medical Advisor identifies healthcare-specific considerations
4. **Initial Estimate**: Each developer privately selects story points
5. **Discussion**: Compare estimates and discuss differences
6. **Re-estimation**: Second round after discussion
7. **Consensus**: Agree on final estimate

### Estimation Validation
**Historical Data**: Track actual time vs. estimates to improve accuracy
**Velocity Tracking**: Use team velocity to validate story point estimates
**Healthcare Benchmarking**: Compare similar healthcare features across sprints

---

## ESTIMATION EXAMPLES

### Example 1: Simple Healthcare Feature
**Story**: "As a patient, I want to view my consultation history so I can track my healthcare interactions"

**Analysis**:
- Healthcare data involved (patient consultations)
- Read-only functionality, no complex business logic
- Requires privacy considerations (RLS policies)
- Standard React Native screen with API integration

**Base Estimate**: 3 points (straightforward)
**Healthcare Multiplier**: 1.2x (basic patient data)
**Final Estimate**: 3-5 points → **5 points**

---

### Example 2: Complex Healthcare Integration
**Story**: "As a pharmacy admin, I want to scan prescription QR codes so I can verify and fulfill prescriptions securely"

**Analysis**:
- Critical prescription data handling
- QR code generation and validation logic
- Integration with pharmacy partner systems
- Security requirements (prevent duplicate fulfillment)
- Multiple user roles and permissions

**Base Estimate**: 8 points (complex)
**Healthcare Multiplier**: 1.4x (prescription data)
**Security Multiplier**: 1.3x (QR security)
**Integration Multiplier**: 1.5x (pharmacy systems)
**Final Estimate**: 8 × 1.4 × 1.3 × 1.5 = **22 points → Break down into smaller stories**
```

```
---

### Example 3: Regulatory Compliance Feature
**Story**: "As a system administrator, I want comprehensive audit logging so we
maintain regulatory compliance"

**Analysis**:
- All healthcare data access must be logged
- Complex business rules for what to log
- Performance considerations (high volume)
- Regulatory requirements (data retention, access controls)
- Integration with multiple system components

**Base Estimate**: 13 points (very complex)
**Compliance Multiplier**: 2.0x (critical safety system)
**Final Estimate**: 13 × 2.0 = **26 points → Break down by system component**

---

## VELOCITY CALCULATION

### Team Velocity Tracking
```

Sprint Velocity = Story Points Completed / Sprint Duration

Historical Velocity (Example):
- Sprint 1: 25 points completed
- Sprint 2: 28 points completed
- Sprint 3: 22 points completed (had production issues)
- Sprint 4: 30 points completed

Average Velocity: 26.25 points per sprint
Planning Velocity: 25 points per sprint (conservative)

```
### Capacity Planning
```

Team Capacity Factors:
- Planned vacation/holidays: -10% to -20%
- New team member onboarding: -15% first sprint, -10% second sprint
- Production support: -5% to -15% depending on user base
- Healthcare training requirements: -5% for compliance training sprints

```
---

## COMMON ESTIMATION PITFALLS

### Healthcare-Specific Pitfalls
❌ **Underestimating Compliance Overhead**: Healthcare features always take longer due to compliance requirements
❌ **Ignoring Partner Integration Complexity**: Healthcare partners have complex systems and processes
❌ **Overlooking Security Requirements**: Healthcare data requires additional security validation
❌ **Missing Testing Requirements**: Healthcare features need more comprehensive testing

### General Pitfalls
❌ **Optimistic Estimation**: Always assuming best-case scenario
❌ **Not Accounting for Integration**: External dependencies always add complexity
❌ **Ignoring Technical Debt**: Existing code quality affects development speed
❌ **Missing Edge Cases**: Healthcare has many regulatory and safety edge cases

### Mitigation Strategies
✅ **Use Historical Data**: Track actual vs. estimated to improve accuracy
✅ **Include Buffer Time**: Add 20% buffer for healthcare features
✅ **Break Down Large Stories**: Nothing should be estimated above 13 points
✅ **Cross-Team Validation**: Have medical advisor review healthcare feature estimates

---

## ESTIMATION REFINEMENT

### Weekly Estimation Review
- **When**: Friday after sprint review
- **Duration**: 15 minutes
- **Participants**: Technical Lead, Development Team
- **Purpose**: Compare actual vs. estimated, identify patterns

### Monthly Estimation Calibration
- **When**: Last Friday of each month
- **Duration**: 45 minutes
- **Participants**: Full development team + Product Manager
- **Purpose**: Update estimation guidelines based on learnings

### Continuous Improvement
- Track estimation accuracy over time
- Identify types of stories consistently over/under-estimated
- Update multipliers based on actual healthcare development experience
- Share learnings with other healthcare development teams

---

**Next Review**: YYYY-MM-DD
**Owner**: Technical Lead
**Distribution**: Development team, Product Manager, Project Manager
```

Performance Test Checklist

~~~markdown
# Performance Testing Checklist - Mediconnect Healthcare Platform

**Document Version**: 1.0
**Project**: Mediconnect MVP
**Last Updated**: YYYY-MM-DD
**Owner**: QA Lead / Performance Engineer

## Pre-Testing Preparation

### Test Environment Setup
- [ ] **Staging Environment Identical to Production**
  - Same server specifications and configurations
  - Same database version and configuration
  - Same network topology and CDN setup
  - Same external integrations (WhatsApp, Maps APIs)

- [ ] **Test Data Preparation**
  - 10,000 synthetic patient records
  - 1,000 healthcare provider accounts (GPs, specialists)
  - 10,000 historical consultations with messages
  - 5,000 prescriptions with various statuses
  - Realistic healthcare partner data (pharmacies, labs)

- [ ] **Monitoring Infrastructure**
  - Application performance monitoring (DataDog/New Relic)
  - Database performance monitoring
  - Server resource monitoring (CPU, memory, disk I/O)
  - Network latency and throughput monitoring
  - External service response time monitoring

### Test Tool Configuration
- [ ] **Load Testing Tools**
  - Artillery or JMeter configured for healthcare workflows
  - React Native performance profiler for mobile testing
  - Database query profiling tools
  - WhatsApp API rate limiting simulation

- [ ] **Performance Baselines**
  - Single user response time benchmarks established
  - Database query performance baselines documented
  - Memory usage patterns under normal load recorded
  - External API response time averages established

---

## LOAD TESTING SCENARIOS

### Scenario 1: Authentication Load
**Objective**: Validate WhatsApp OTP authentication under high concurrent load

**Test Parameters**:
- **Concurrent Users**: 100 users simultaneously requesting OTP
- **Ramp-up Period**: 60 seconds
- **Test Duration**: 10 minutes
- **Success Criteria**:
  - OTP delivery rate > 95%
  - Average response time < 3 seconds
  - No authentication service failures

**Test Script**:
```yaml
config:
~~~

```
  target: 'https://staging-api.mediconnect.com'
  phases:
    - duration: 60
      arrivalRate: 2
      name: "Ramp up"
    - duration: 600
      arrivalRate: 10
      name: "Sustained load"

scenarios:
  - name: "WhatsApp OTP Authentication"
    weight: 100
    flow:
      - post:
          url: "/auth/whatsapp/otp"
          json:
            phone_e164: "+254{{ $randomInt(700000000, 799999999) }}"
          expect:
            - statusCode: 202
            - hasHeader: "x-request-id"
      - think: 30  # Wait for WhatsApp delivery
      - post:
          url: "/auth/whatsapp/verify"
          json:
            phone_e164: "{{ phone_e164 }}"
            code: "123456"  # Test code
          expect:
            - statusCode: 200
            - hasHeader: "authorization"
```

Success Metrics:
- [ ] 95% of OTP requests return status 202 within 3 seconds
- [ ] WhatsApp delivery rate tracked and > 95%
- [ ] No database connection pool exhaustion
- [ ] Memory usage remains stable throughout test

Scenario 2: Concurrent Consultations

Objective: Test system capacity for simultaneous GP consultations and chat

Test Parameters:
- **Concurrent Consultations**: 60 simultaneous active consultations
- **Chat Message Rate**: 5 messages per minute per consultation
- **Test Duration**: 30 minutes
- **Success Criteria**:
- Chat message delivery < 2 seconds (P95)
- Database queries maintain < 500ms response time
- Real-time updates functioning correctly

Test Script:

```yaml
scenarios:
  - name: "GP Consultation Simulation"
    weight: 100
    flow:
      # Start consultation
      - post:
          url: "/consultations"
          headers:
            Authorization: "Bearer {{ authToken }}"
          json:
            patient_id: "{{ patientId }}"
            chief_complaint: "Headache and fever"
          expect:
            - statusCode: 201
          capture:
            json: "$.id"
            as: "consultationId"

      # Simulate chat messages during consultation
      - loop:
          count: 10
          over:
            - post:
                url: "/consultations/{{ consultationId }}/messages"
                json:
                  message: "{{ $randomString() }}"
                  type: "text"
                expect:
                  - statusCode: 201
            - think: 60  # 1 minute between messages
```

Success Metrics:
- [ ] 60 concurrent consultations maintain < 500ms API response times
- [ ] Chat messages delivered in < 2 seconds (P95)
- [ ] Real-time notifications functioning under load
- [ ] Database connection pool stable
- [ ] No WebSocket connection drops

Scenario 3: Prescription and Pharmacy Load

Objective: Test prescription creation and pharmacy QR verification under load

Test Parameters:
- **Prescription Creation Rate**: 20 prescriptions per minute
- **QR Verification Rate**: 50 QR scans per minute
- **PDF Generation**: 10 PDF downloads per minute
- **Test Duration**: 20 minutes

Success Criteria:
- QR code generation < 1 second
- PDF generation < 5 seconds
- QR verification < 2 seconds
- No duplicate QR codes generated

Test Script:

```yaml
scenarios:
  - name: "Prescription Workflow"
    weight: 60
    flow:
      # Create prescription
      - post:
          url: "/prescriptions"
          headers:
            Authorization: "Bearer {{ authToken }}"
          json:
            consultation_id: "{{ consultationId }}"
            patient_id: "{{ patientId }}"
            items:
              - medication: "Paracetamol 500mg"
                quantity: 20
                instructions: "Take 1 tablet every 6 hours"
          expect:
            - statusCode: 201
          capture:
            json: "$.qr_code"
            as: "qrCode"

      # Generate PDF (30% of prescriptions)
      - get:
          url: "/prescriptions/{{ prescriptionId }}/pdf"
          ifTrue: "{{ $randomInt(1, 10) <= 3 }}"
          expect:
            - statusCode: 200
            - contentType: "application/pdf"

  - name: "Pharmacy QR Verification"
    weight: 40
    flow:
      - post:
          url: "/pharmacy/claims/verify-qr"
          headers:
            Authorization: "Bearer {{ pharmacyToken }}"
          json:
            qr_payload: "{{ qrCode }}"
            org_id: "{{ pharmacyId }}"
          expect:
            - statusCode: 200
```

Success Metrics:
- [ ] Prescription creation < 1 second (P95)
- [ ] QR code uniqueness verified (no duplicates)
- [ ] PDF generation < 5 seconds (P95)
- [ ] QR verification < 2 seconds (P95)
- [ ] Database integrity maintained under load

STRESS TESTING SCENARIOS

Database Stress Test

Objective: Identify database breaking point and resource limits

Test Configuration:
- Gradually increase concurrent users from 10 to 200
- Monitor database CPU, memory, connection pool
- Identify point where response times degrade significantly
- Test database failover and recovery procedures

Key Metrics:
- [ ] Maximum concurrent users before degradation
- [ ] Database CPU utilization under stress
- [ ] Connection pool exhaustion point identified
- [ ] Query timeout rates under maximum load

Memory Leak Detection

Objective: Ensure application doesn't have memory leaks under sustained load

Test Configuration:
- Run sustained load for 4+ hours
- Monitor application memory usage patterns
- Check for gradual memory increase indicating leaks
- Validate garbage collection effectiveness

Success Criteria:
- [ ] Memory usage stabilizes after initial ramp-up
- [ ] No continuous memory growth over 4-hour period
- [ ] Garbage collection cycles remain consistent
- [ ] No out-of-memory errors during sustained load

MOBILE APP PERFORMANCE TESTING

React Native Performance

Test Scenarios:
- [ ] **App Launch Time**: < 3 seconds on mid-range Android devices
- [ ] **Screen Navigation**: < 200ms transition times
- [ ] **Chat Interface**: Handle 1000+ message history without lag
- [ ] **Offline Performance**: Graceful degradation with poor connectivity
- [ ] **Battery Usage**: No excessive battery drain during normal usage

Device Compatibility Testing

Test Devices:
- [ ] **Low-end Android** (2GB RAM, older processor)
- [ ] **Mid-range Android** (4GB RAM, current-gen processor)
- [ ] **High-end Android** (8GB+ RAM, flagship processor)

Performance Criteria per Device Tier:

```
Low-end devices:
- App launch < 5 seconds
- Screen transitions < 500ms
- Chat loads 50 messages < 3 seconds

Mid-range devices:
- App launch < 3 seconds
- Screen transitions < 300ms
- Chat loads 100 messages < 2 seconds

High-end devices:
- App launch < 2 seconds
- Screen transitions < 200ms
- Chat loads 200+ messages < 1 second
```

API PERFORMANCE BENCHMARKS

Critical Endpoint Performance

Authentication Endpoints:
- [ ] `POST /auth/whatsapp/otp`: < 1 second (P95)
- [ ] `POST /auth/whatsapp/verify`: < 500ms (P95)

Consultation Endpoints:
- [ ] `POST /consultations`: < 800ms (P95)
- [ ] `GET /consultations/{id}/messages`: < 1 second for last 50 messages (P95)
- [ ] `POST /consultations/{id}/messages`: < 300ms (P95)

Healthcare Data Endpoints:
- [ ] `POST /prescriptions`: < 1 second (P95)
- [ ] `POST /pharmacy/claims/verify-qr`: < 500ms (P95)
- [ ] `POST /labs/orders`: < 800ms (P95)

API Rate Limiting Validation

Rate Limit Testing:
- [ ] Verify rate limits prevent abuse without affecting normal usage
- [ ] Test rate limit headers are correctly returned
- [ ] Confirm appropriate HTTP 429 responses with Retry-After headers
- [ ] Validate rate limiting doesn’t affect healthcare emergency scenarios

EXTERNAL SERVICE PERFORMANCE

WhatsApp Cloud API Performance

Test Scenarios:
- [ ] **OTP Delivery Time**: Measure average delivery time to Kenyan numbers
- [ ] **Rate Limit Handling**: Test against WhatsApp’s rate limits
- [ ] **Error Rate Monitoring**: Track failed message delivery rates
- [ ] **Webhook Performance**: Test webhook processing under load

Success Criteria:
- [ ] 95% of WhatsApp messages delivered within 10 seconds
- [ ] Rate limiting gracefully handled with proper backoff
- [ ] Webhook processing