Software & Security Architecture, Deployment Infrastructure and

Architecture Execution Plan

For

Team Eagles

**Contents**

[1.](#_heading=h.wcc4ax4nx9or) OVERVIEW 3

[2.](#_heading=h.u6ld7dy3pir8) SOFTWARE ARCHITECTURE 3

[2.1.](#_heading=h.ktp76dvalkkk) High Level logical components and their interactions 3

[2.2.](#_heading=h.1vym51ge7nuy) Description of the Modules or system components of the system 4

[2.3.](#_heading=h.64cpofbw4el1) Outline the Type of connections and integration patterns 6

[2.4 Architecture styles used (E.G., tiers, event driven …) 7](#_heading=h.sknq9kap4uv6)

[3.](#_heading=h.ugrf1k9l1ugo) SECURITY ARCHITECTURE 9

[3.1.](#_heading=h.omuhlpz69suz) Security architecture diagrams 9

[3.2.](#_heading=h.3gh9982nx26) Components Description: 9

[4.](#_heading=h.uwx7htlfwpj9) DEPLOYMENT INFRASTRUCTURE 11

[5.](#_heading=h.f4e0vooswi48) ARCHITECTURE EXECUTION PLAN 14

[5.1.](#_heading=h.gank0z9g5ese) TIMELINE 14

[6.](#_heading=h.pqpp8lsozhne) OPERATIONALIZATION (SUPPORT, MONITORING, MAINTENANCE) 15

[7.](#_heading=h.ex566jlpfifq) GLOSSARY (example) 15

[8.](#_heading=h.xrv7bu47efgp) REFERENCES 15

# OVERVIEW

This document outlines the foundational facets of architecture required for the solution. These facets, or “views” of the architecture, are constructed to support the requirements and architecture defined in Deliverable 1. Deliverable 1 focused primarily on capturing details of the external view of the system as the user will experience and use it. In this document, the focus is on the following internal architectural views of the system as well as an implementation plan:

* **Software Architecture**
* **Security Architecture**
* **Deployment Infrastructure**
* **Execution Plan**

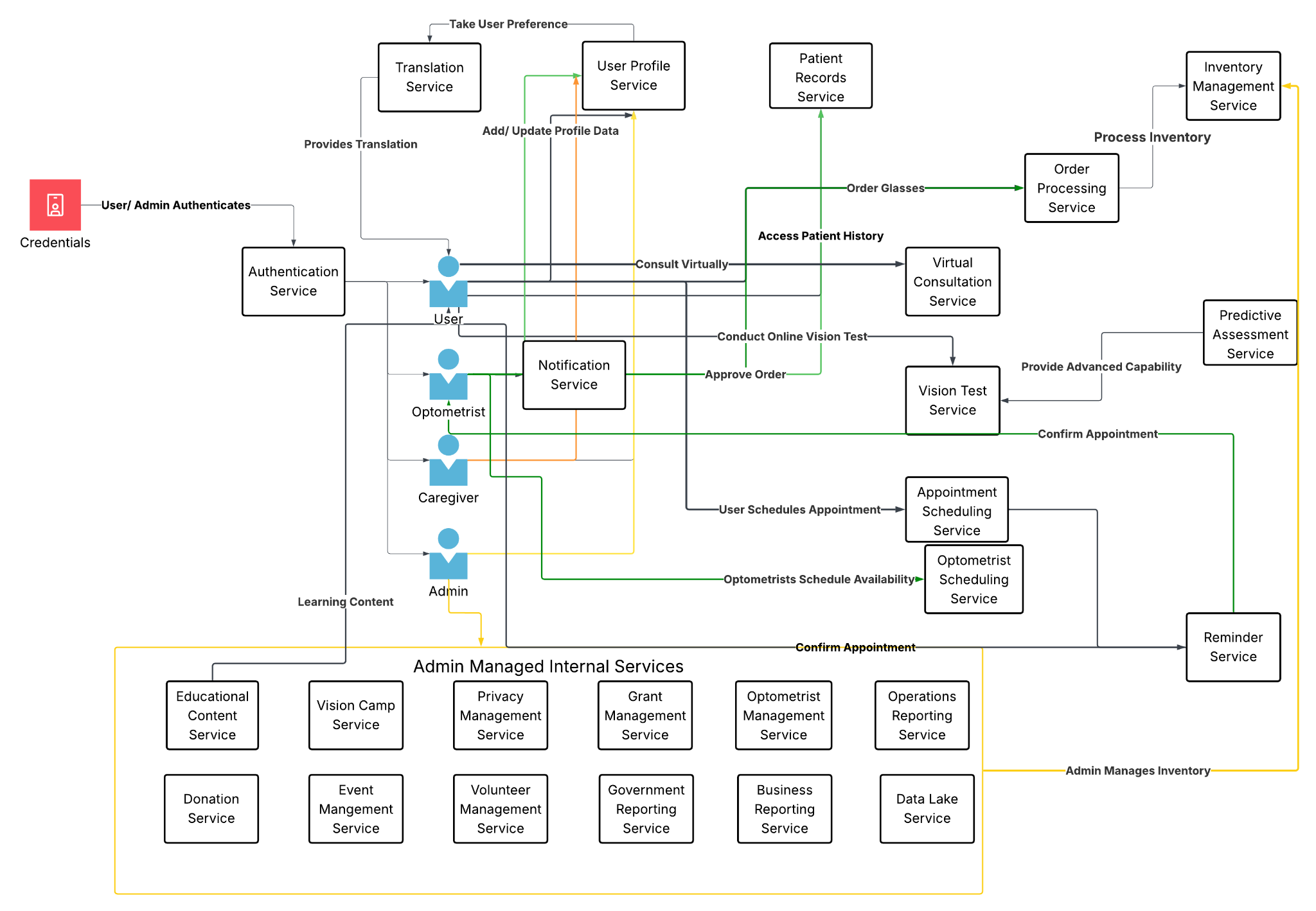
# SOFTWARE ARCHITECTURE

The iCare platform follows a microservices architecture designed to support vision screening and correction for 100M people in India. Our architecture emphasizes offline capabilities, privacy-first design, and AI-powered vision assessment.

The core components include:

1. **Vision Test Service**
   * Self-administered vision tests via mobile app
   * AI-powered risk assessment using smartphone camera
   * Offline-mode with data synchronization
2. **Appointment Management Service**
   * Scheduling with nearby optometrists
   * Automated reminders via SMS/push notifications
   * Provider availability tracking
3. **Prescription & Eyewear Management Service**
   * Digital prescription storage and tracking
   * Eyewear ordering with subsidy verification
   * Order tracking and delivery management
4. **User Profile Service**
   * Multi-language user registration and preferences
   * Medical history and vision test records
   * Privacy controls and consent management
5. **Vision Risk Model Service**
   * Front camera-based eye screening
   * AI/ML analysis for vision risk assessment
   * Privacy-focused image processing

## High Level logical components and their interactions



The iCare platform implements a microservices architecture with five key service components that interact through well-defined APIs to deliver the complete vision screening and care solution.

1. **Vision Test Service**
   * Provides self-administered vision tests via mobile app
   * Implements AI-powered risk assessment using smartphone camera
   * Supports offline-mode with data synchronization for rural areas
2. **Appointment Management Service**
   * Enables scheduling with nearby optometrists
   * Delivers automated reminders via SMS/push notifications
   * Tracks provider availability for efficient appointment allocation
3. **Prescription & Eyewear Management Service**
   * Manages digital prescription storage and tracking
   * Handles eyewear ordering with subsidy verification
   * Provides order tracking and delivery management
4. **User Profile Service**
   * Supports multi-language user registration and preferences
   * Maintains medical history and vision test records
   * Implements privacy controls and consent management
5. **Vision Risk Model Service**
   * Enables front camera-based eye screening
   * Performs AI/ML analysis for vision risk assessment
   * Implements privacy-focused image processing

These services interact through secure APIs and event-driven messaging to provide a seamless user experience while maintaining separation of concerns for scalability and maintenance.

The system is designed with a mobile-first approach, emphasizing offline capabilities for rural areas with limited connectivity, while ensuring data privacy through consent-based data collection and processing.

## Description of the Modules or system components of the system

The following table provides a detailed description of each system component with their associated user stories:

| **Component Name** | **Description** | **User Stories** |
| --- | --- | --- |
| Vision Test Service | Enables users to perform self-guided vision tests directly through the mobile application. Includes standard visual acuity tests, color vision tests, and astigmatism tests presented in an intuitive interface. Test results are stored securely and can be accessed by authorized healthcare providers. | 1.1, 1.2, 1.4, 1.5, 1.6 |
| AI Risk Assessment | Analyzes vision test results using AI algorithms to identify potential vision issues and their severity. Provides immediate feedback to users with clear recommendations for next steps. | 1.1, 1.2, 5.2 |
| Offline Mode Manager | Enables critical functionality to continue working in areas with limited connectivity. Stores test results locally and synchronizes with the central database when connectivity is restored. | 1.3, 1.6 |
| Appointment Management Service | Facilitates scheduling appointments with nearby optometrists based on location, availability, and specialty. Includes calendar integration and automated booking confirmation. | 2.1, 2.2, 2.3, 2.4 |
| Notification System | Delivers timely reminders about upcoming appointments, nearby eye camps, and follow-up care recommendations via SMS, push notifications, or in-app alerts based on user preferences. | 2.2, 1.6, 6.1 |
| Provider Availability Tracker | Manages optometrist schedules and availability slots to optimize appointment booking and reduce wait times. Provides real-time calendar updates to healthcare providers. | 2.3, 2.4, 7.1 |
| Prescription Management | Securely stores and manages digital prescriptions with version control and expiration tracking. Enables optometrists to create, update, and share prescriptions digitally. | 3.1, 3.4, 5.1 |
| Eyewear Order Processing | Handles the complete order lifecycle from prescription validation to delivery tracking. Includes subsidy eligibility verification for low-income patients. | 3.2, 3.3 |
| User Profile Service | Manages user registration, authentication, and profile information with support for multiple languages and accessibility preferences. Maintains comprehensive medical history and test records. | 4.1, 4.2, 4.3, 4.4, 1.4 |
| Consent Management | Implements privacy controls for data sharing and storage with clear opt-in/opt-out mechanisms and consent tracking. Enforces data retention policies based on user preferences. | 11.4 |
| Vision Risk Model Service | Processes eye selfies taken with a smartphone's front camera to detect vision risks. Uses computer vision techniques to analyze eye features and identify potential issues before professional screening. | 1.1, 1.2, 1.5 |
| Image Preprocessing | Detects and isolates eye regions from selfie images, enhances image quality, and normalizes lighting and contrast to improve analysis accuracy. Can operate on-device or server-side depending on device capabilities. | 1.1, 1.2, 1.5 |
| Multilingual Support | Provides interface translations, instructions, and content in multiple Indian languages to ensure accessibility across diverse linguistic communities. Includes text-to-speech capabilities for users with limited literacy. | 4.1, 4.2, 4.3, 4.4, 4.5 |
| Analytics Engine | Collects and analyzes platform usage data, test results, and appointment trends to generate insights for service improvement and impact measurement. Supports reporting for donors and stakeholders. | 5.2, 5.4, 7.2, 7.4 |

Each component is designed with modularity in mind, allowing for independent development, testing, and scaling while maintaining interoperability through standardized APIs.

## Outline the Type of connections and integration patterns

| **Connector Name** | **Description** |
| --- | --- |
| Eye Image Capture | The mobile app uses the front camera to capture eye selfies. Images are preprocessed either on-device using OpenCV.js or sent to the backend for processing with OpenCV-Python. |
| Vision Risk Analysis | Preprocessed images are sent to the ML Inference Engine, which returns risk assessment results. This connection can use either Hugging Face Inference API or AWS SageMaker endpoints. |
| Metadata Enhancement | User metadata (age, region, past test results) is combined with image analysis to improve prediction accuracy and provide contextual recommendations. |
| Results Integration | Analysis results are returned to the mobile app and displayed in the user's test history and dashboard. The app shows risk level, confidence score, and actionable recommendations. |
| Consent-Based Storage | With user permission, images may be encrypted and stored in S3 with a 7-day TTL. If users don't consent, images are processed but not persistently stored. |
| Offline Synchronization | When operating offline, test data is stored locally on the device and synchronized with backend services when connectivity is restored. |

## 2.4 Architecture styles used (E.G., tiers, event driven …)

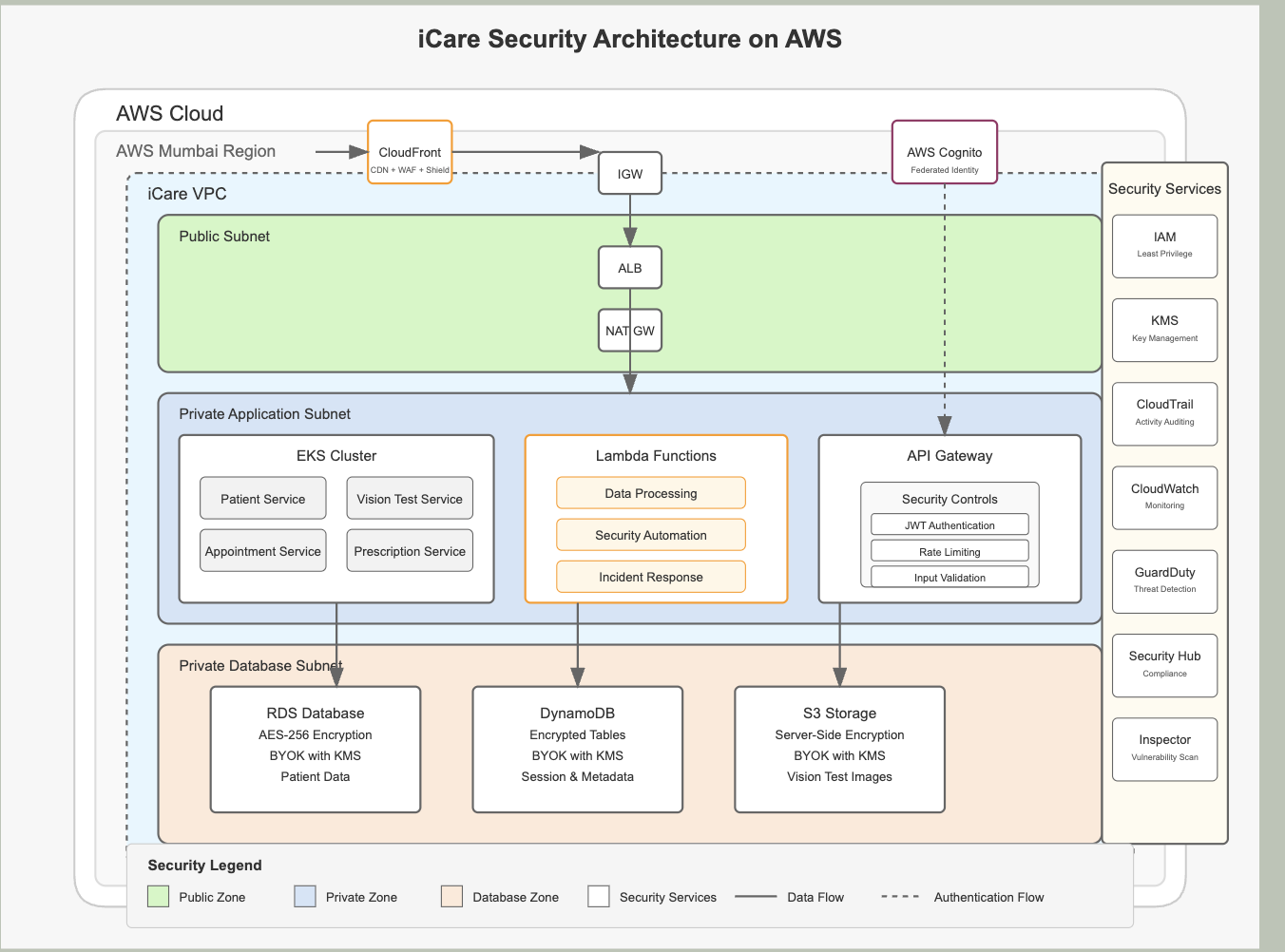
The iCare platform employs multiple architectural patterns tailored to the unique requirements of providing accessible vision care:

1. **Microservices Architecture**
   * Independent, loosely-coupled services organized around business capabilities
   * Each service maintains its own data store and business logic
   * Enables independent deployment and scaling of vision test, appointment, and prescription management services
   * Facilitates team autonomy and parallel development
2. **Event-Driven Architecture**
   * Asynchronous communication between services using events
   * Services publish events when important state changes occur
   * Examples include appointment bookings, test completions, and prescription updates
   * Improves system resilience and reduces inter-service dependencies
3. **Mobile-First Design**
   * Optimized for mobile device capabilities and limitations
   * Responsive UI adapting to various screen sizes and resolutions
   * Offline-first approach for areas with limited connectivity
   * Efficient battery and bandwidth usage
4. **Edge Computing**
   * Image preprocessing can occur on the mobile device using OpenCV.js
   * Reduces bandwidth usage and server load
   * Improves responsiveness in low-connectivity environments
   * Protects user privacy by minimizing data transmission
5. **API Gateway Pattern**
   * Centralized entry point for all client requests
   * Handles cross-cutting concerns like authentication and rate limiting
   * Simplifies client integration with multiple microservices
   * Provides consistent interface while hiding internal architecture
6. **Privacy by Design**
   * Consent-driven data collection and storage
   * Time-limited retention policies (7-day maximum for images)
   * Data minimization and purpose limitation
   * End-to-end encryption for sensitive health data

| Component | Description |
| --- | --- |
| Client Layer | End users can access applications through this layer. Various browsers or thin desktop clients are examples of clients. |
| Presentation Layer | The presentation layer deals with the presentation logic and the page rendering. It contains all the components needed to allow interactions with an end-user. |
| Business Layer | This layer encodes the real-world business rules that determine how data can be created, displayed, stored, and changed. Various business processes and its interaction, governed by various rules are managed at this layer. |
| Data Layer | Rules related to data access and management are implemented at this layer. It abstracts the logic required to access the underlying data stores. |

# SECURITY ARCHITECTURE

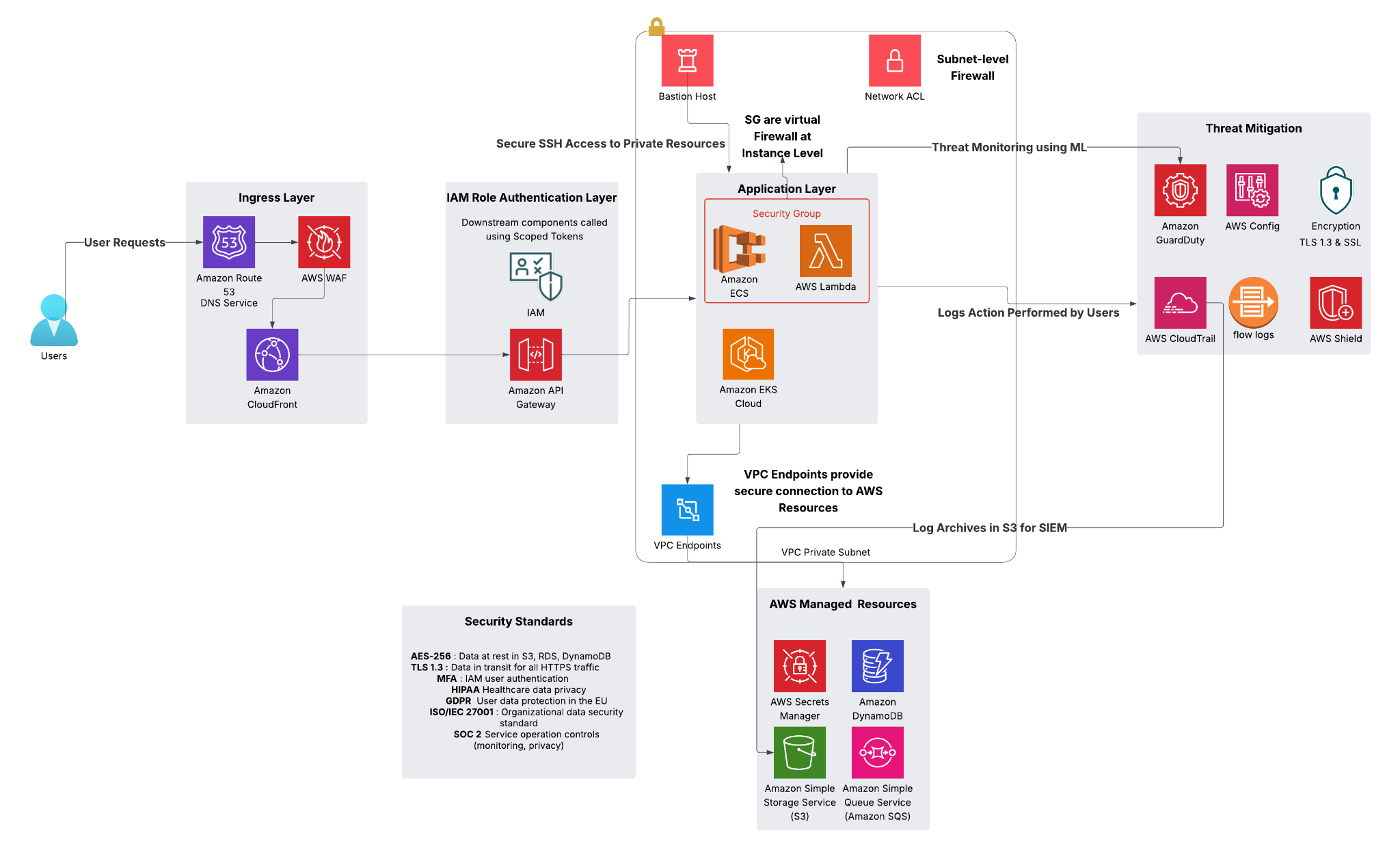
# 



## Components Description

| **Component** | **Description** |
| --- | --- |
| User Consent Framework | Implements privacy by design principles with explicit opt-in for image storage. Uses User\_Preferences table to track consent settings. Ensures compliance with healthcare data privacy regulations. |
| Data Encryption Service | Implements end-to-end encryption for all sensitive data, including eye images and health records. Uses AES-256 encryption for data at rest in S3 and TLS 1.3 for data in transit. |
| Image Storage TTL | Automatically removes eye images after 7 days using S3 lifecycle policies unless explicitly retained by users. Provides an additional privacy safeguard for sensitive biometric data. |
| Authentication Service | Manages user identity and access control with multi-factor authentication options. Supports social login integration and role-based access control for different user types. |
| API Security | Implements rate limiting, request validation, and JWT authentication for all API endpoints. Prevents common API vulnerabilities and abuse. |
| Audit Logging | Tracks all security-relevant actions including consent changes, data access, and administrative operations. Provides tamper-proof security event records. |
| HIPAA/GDPR Controls | Ensures compliance with healthcare privacy regulations through data minimization, access controls, and secure processing practices. |
| Secure Defaults | Implements conservative security defaults throughout the system. Ensures that security features are enabled by default and require explicit action to disable. |

## Security architecture diagrams



## Components Description

| **Component** | **Description** |
| --- | --- |
| **Authenticator** | Verifies the identity of users and services. Supports password-based login, social login (e.g., Google, Aadhaar eKYC), and multi-factor authentication. Integrates with the API Gateway to validate JWT tokens on each request. |
| **Session Manager** | Maintains user session data securely. Issues session tokens, handles expiration, and supports logout mechanisms. Works with Redis (ElastiCache) for fast token lookups. |
| **Cache Manager** | Temporarily stores session and frequently accessed user data to improve response times and reduce database load. Uses in-memory caching with eviction policies. |
| **Cryptography** | Provides cryptographic services for data encryption/decryption, secure key management, and digital signature verification. Uses AES-256 encryption for stored data and TLS 1.3 for secure transmission. |
| **Consent Manager** | Central module that handles user privacy preferences. Logs all consent events, supports revocation, and enforces TTL policies for biometric data. |
| **Security Event Logger** | Captures logs of critical events including login attempts, data access, and admin actions. Sends events to a secure log store for audit and analysis. |
| **Compliance Controller** | Ensures all processing adheres to HIPAA and GDPR regulations. Controls data minimization, access logs, and lawful basis of processing. |

# DEPLOYMENT INFRASTRUCTURE

The iCare platform requires a scalable, reliable infrastructure capable of serving 100M users across India, with particular attention to areas with limited connectivity.

## Compute Resources

| **Component** | **Description** | **Technology** |
| --- | --- | --- |
| Mobile Application | Cross-platform mobile app with offline capabilities | React Native / Flutter with OpenCV.js |
| API Gateway | Entry point for all client requests | AWS API Gateway |
| Vision Risk Service | Processes eye images and performs risk assessment | AWS Lambda / ECS with OpenCV-Python |
| ML Inference Service | Runs the vision risk prediction model | Hugging Face Inference API or AWS SageMaker |
| Web Application | Administrative interfaces and provider portals | React.js with responsive design |
| Background Processing | Asynchronous tasks like image processing and notifications | AWS Lambda, SQS |

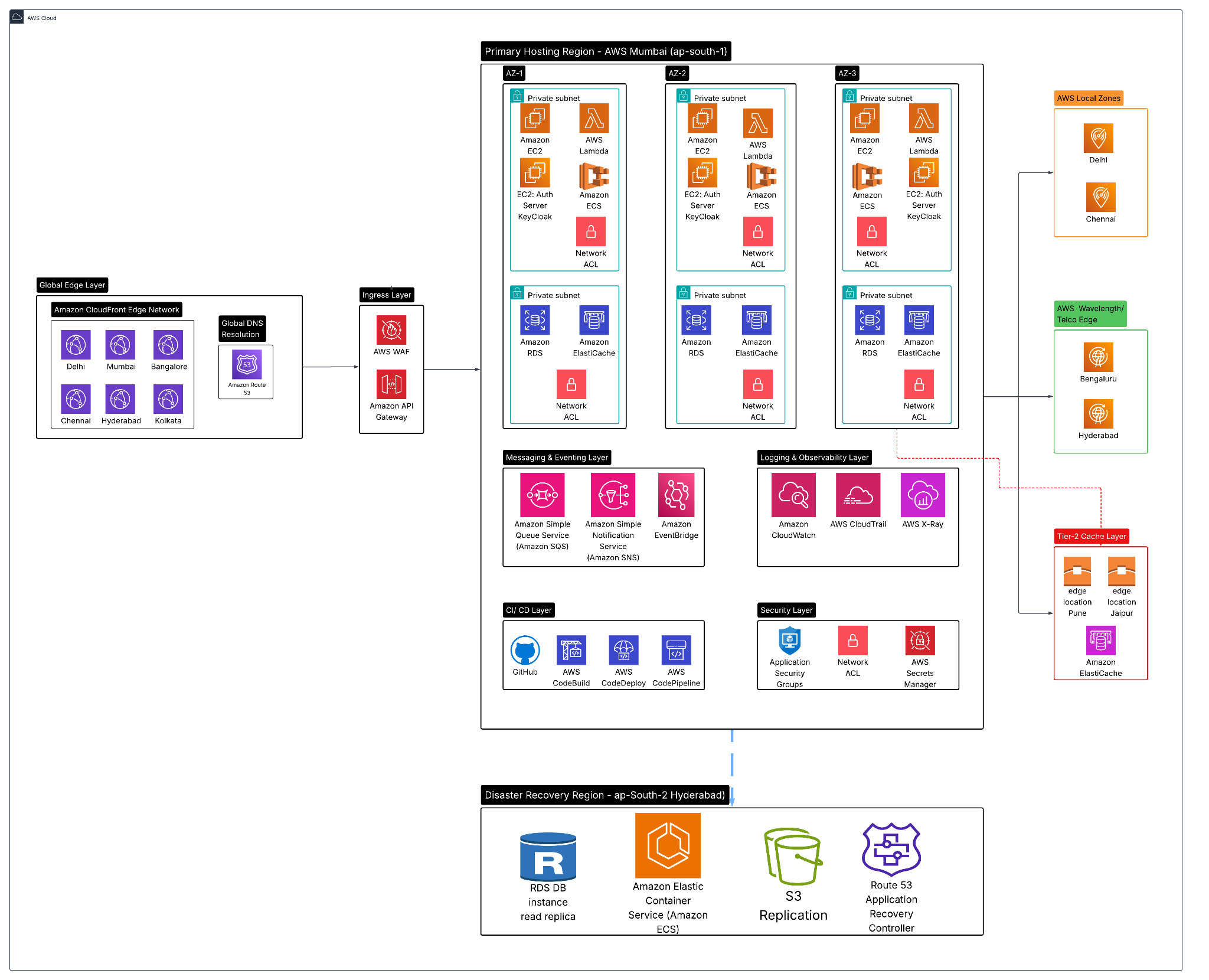
## Storage Resources

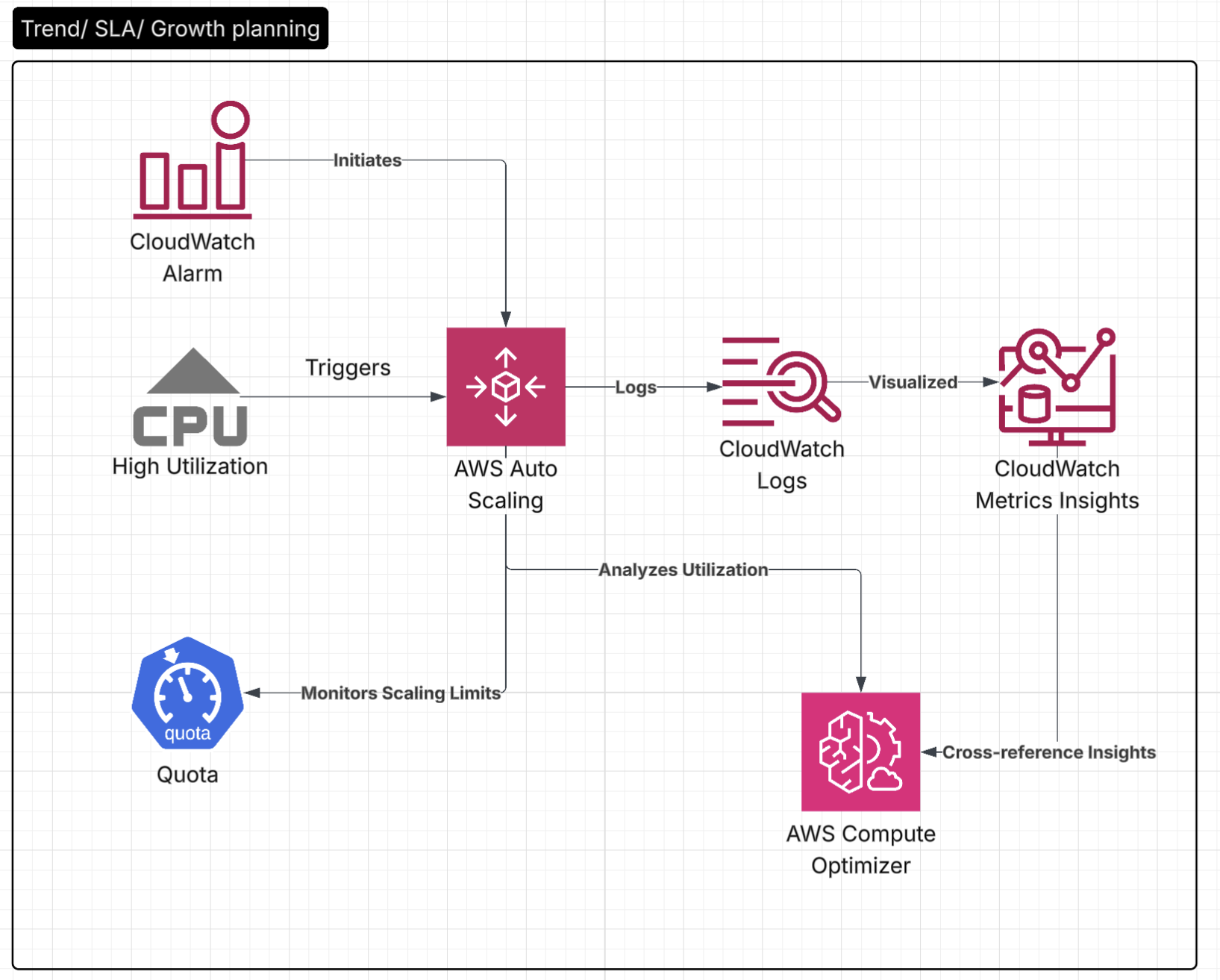
| **Component** | **Description** | **Technology** |
| --- | --- | --- |
| User Data Store | Stores user profiles, preferences, and medical records | Amazon RDS (PostgreSQL) |
| Image Storage | Temporary storage for eye images (with 7-day TTL) | Amazon S3 with server-side encryption |
| Session Cache | Improves performance for frequent operations | Amazon ElastiCache (Redis) |
| Offline Data | Local storage for app data when offline | SQLite / React Native AsyncStorage |

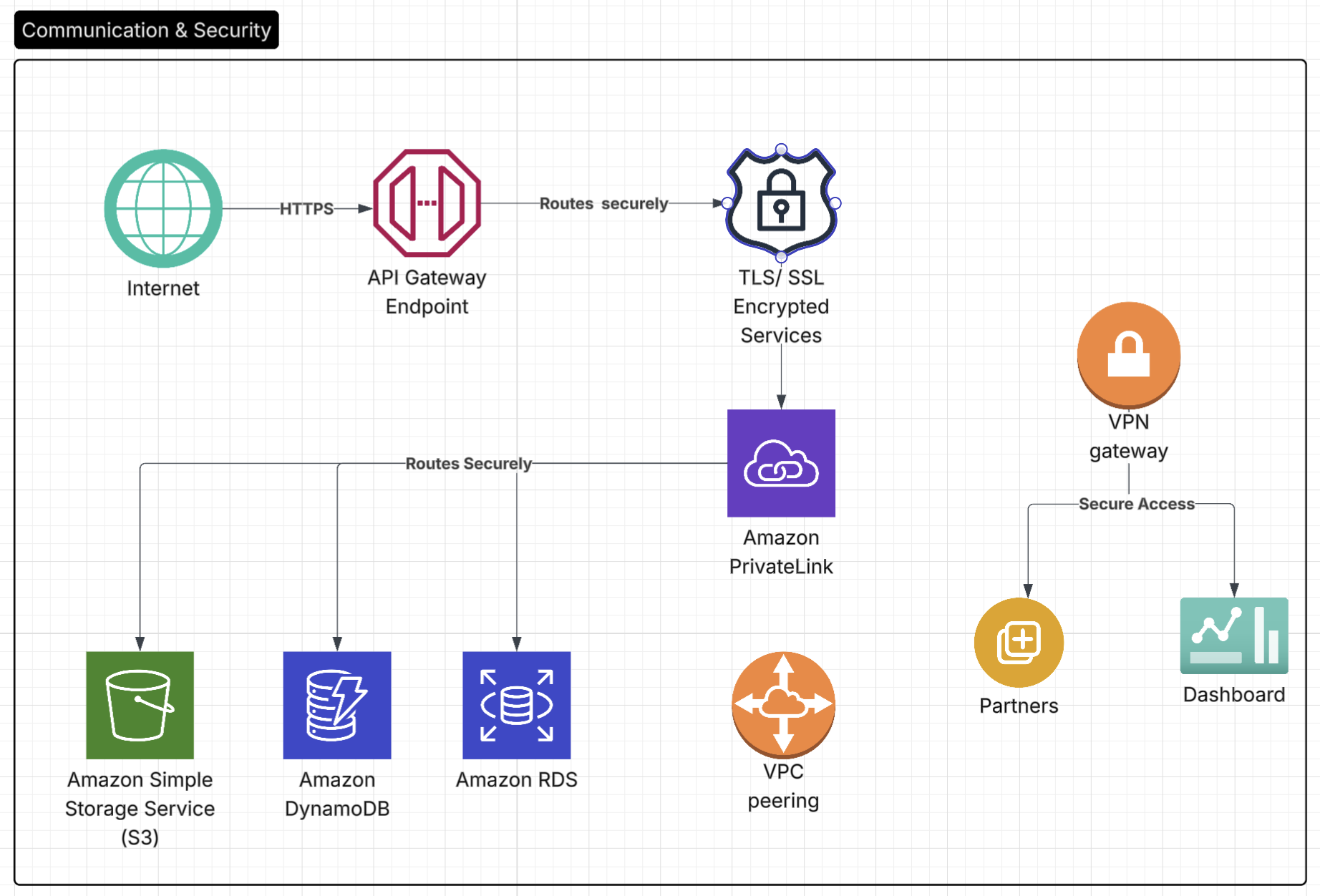
## Network Resources

| **Component** | **Description** | **Technology** |
| --- | --- | --- |
| Content Delivery | Delivers static assets to users across India | Amazon CloudFront |
| API Connectivity | Secures and manages API traffic | AWS API Gateway, Lambda |
| Regional Distribution | Ensures low latency across India | Multi-region deployment (Mumbai, Hyderabad) |
| Offline Support | Handles intermittent connectivity | Progressive Web App techniques, background sync |

AWS Deployment Diagram







* 1. **TECHNOLOGY STACK**

| **Component** | **Technology Choice** | **Justification** |
| --- | --- | --- |
| Mobile Frontend | React Native / Flutter | Cross-platform capability, good performance on low-end devices, offline support |
| Image Processing | OpenCV.js (client-side) / OpenCV-Python (server-side) | Industry-standard image processing library, available for both client and server |
| ML Model | Hugging Face AutoTrain Vision / Custom ResNet18+MLP | Rapid development, good performance with limited training data (100-300 images) |
| Model Hosting | Hugging Face Inference API / AWS SageMaker | Simplified deployment, managed scaling, cost-effective for MVP |
| API Gateway | AWS API Gateway | Managed service with security, throttling, and monitoring capabilities |
| Serverless Functions | AWS Lambda | Cost-effective, auto-scaling, supports Python for image processing |
| Object Storage | Amazon S3 | Secure, durable storage with lifecycle policies for TTL enforcement |
| Database | Amazon RDS (PostgreSQL) | Relational database for structured user and prescription data |
| Caching | Amazon ElastiCache (Redis) | In-memory cache for session data and frequent queries |

**Component to Technology Mapping**

| **Architecture Component** | **Technology Choice** | **Justification** |
| --- | --- | --- |
| Vision Risk Model Service | AWS Lambda + OpenCV-Python + Hugging Face | Serverless architecture for cost-effective scaling, OpenCV for powerful image processing, Hugging Face for simplified ML deployment |
| Mobile Camera Interface | React Native camera module | Cross-platform camera access with alignment guides and quality feedback |
| Image Preprocessing | OpenCV.js (mobile) / OpenCV-Python (server) | Flexibility to process on device when possible or server-side for complex cases |
| User Consent Manager | User\_Preferences table in RDS | Centralized consent tracking integrated with user profile |
| Secure Storage | S3 + Server-side encryption + Lifecycle policy | Encrypted storage with automatic deletion after 7 days |
| Offline Mode | SQLite + Background sync | Local database with synchronization when connectivity is restored |

# ARCHITECTURE EXECUTION PLAN

Our implementation approach follows a phased rollout strategy, focusing first on core vision testing capabilities, then expanding to broader healthcare integration.

## TIMELINE

Phase 1: Vision Risk Model MVP (Months 1-2)

**Focus:** Core vision risk assessment using smartphone camera

| **Epic** | **User Stories** | **Architecture Components** |
| --- | --- | --- |
| Vision Screening & Diagnosis | 1.1, 1.2, 1.5 | Vision Risk Model Service, Mobile Camera Interface, Image Preprocessing |

**Deliverables:**

* Front camera capture flow with alignment guidance
* OpenCV-based image preprocessing
* Basic ML model for vision risk assessment
* Simple mobile UI for test results

### Phase 2: Enhanced User Experience (Months 3-4)

**Focus:** Improved user experience and data integration

| **Epic** | **User Stories** | **Architecture Components** |
| --- | --- | --- |
| Vision Health Tracking & Reports | 5.1, 5.2 | Metadata Enrichment Service, User Profile Service |
| Multilingual & Inclusive Access | 4.1, 4.2, 4.4 | Accessibility Layer, Translation Service |

**Deliverables:**

* Multi-language support for test instructions
* Vision health history tracking
* Enhanced risk assessment with metadata
* Offline mode support

### Phase 3: Professional Integration (Months 5-6)

**Focus:** Connect self-assessment to professional care

| **Epic** | **User Stories** | **Architecture Components** |
| --- | --- | --- |
| Appointment Scheduling | 2.1, 2.2, 2.3 | Appointment Service, Notification Service |
| Prescription Management | 3.1, 3.4 | Prescription Service, Provider Portal |

**Deliverables:**

* Appointment booking based on test results
* Prescription management integration
* Provider portal for optometrists
* Referral system for detected risks

### Phase 4: Scale & Analytics (Months 7-8)

**Focus:** Platform scaling and data insights

| **Epic** | **User Stories** | **Architecture Components** |
| --- | --- | --- |
| iCare Business Analytics | 7.1, 7.2, 7.4 | Analytics Service, Reporting Dashboard |
| Government & Policy Integration | 10.1, 10.2 | Data Export API, Compliance Framework |

**Deliverables:**

* Enhanced analytics dashboard
* Population health insights
* Government/NGO data integration
* System optimizations for scale

### 

# OPERATIONALIZATION (SUPPORT, MONITORING, MAINTENANCE)

## Monitoring Strategy

* **Application Performance:** Track API response times, error rates, and user experience metrics
* **ML Model Performance:** Monitor prediction accuracy, confidence scores, and false positives/negatives
* **Infrastructure Health:** Monitor compute utilization, storage consumption, and network performance
* **User Behavior:** Track test completion rates, retry attempts, and conversion to appointments
* **Data Privacy:** Audit consent changes, data access patterns, and retention policy compliance

## Support Model

* **User Support:** In-app help, chatbot assistance, and email support for technical issues
* **Provider Support:** Dedicated support channel for optometrists and healthcare providers
* **Operations Team:** 24/7 monitoring for critical infrastructure and security alerts
* **ML Ops Team:** Regular model performance review and retraining schedule

## Maintenance Plan

* **Weekly Updates:** Bug fixes, minor feature improvements, and security updates
* **Monthly Releases:** New features, UI enhancements, and performance optimizations
* **Quarterly Model Updates:** ML model retraining with new data to improve accuracy
* **Biannual Security Audits:** Comprehensive security assessment and vulnerability testing

## Disaster Recovery

* **Data Backup:** Daily backups of all user data and system configurations
* **Failover Strategy:** Automatic failover to secondary region in case of primary region failure
* **Recovery Testing:** Regular drills to validate recovery procedures and timelines
* **Business Continuity:** Offline capabilities ensure basic service availability during outages

## Capacity Planning

* **User Growth:** Infrastructure designed to scale to 100M users over 7 years
* **Storage Growth:** Projected storage needs with time-limited retention policies
* **Compute Scaling:** Auto-scaling policies based on demand patterns
* **Bandwidth Planning:** CDN distribution strategy for static assets

The iCare platform emphasizes a privacy-first, mobile-friendly approach to vision care, particularly focusing on using existing smartphone technology to provide accessible screening tools for Uncorrected Refractive Error (URE). Our technical implementation leverages modern cloud infrastructure while accommodating the unique challenges of the Indian market, including variable connectivity and diverse language requirements.

# GLOSSARY (example)

| Term | Description |
| --- | --- |
| NFR | Non-Functional Requirement(s) |
| OMS | Order Management System |
| ERD | Entity relationship diagram |

# REFERENCES

| **Type** | **Reference** |
| --- | --- |
| **Data Privacy Regulations** | [General Data Protection Regulation (GDPR)](https://gdpr.eu/), Health Insurance Portability and Accountability Act (HIPAA), [India's Digital Personal Data Protection Act (DPDP 2023)] |
| **Cloud Infrastructure** | [AWS Well-Architected Framework](https://aws.amazon.com/architecture/well-architected/), [AWS Shared Responsibility Model](https://aws.amazon.com/compliance/shared-responsibility-model/) |
| **Security Standards** | OWASP API Security Top 10, [NIST Cybersecurity Framework](https://www.nist.gov/cyberframework) |
| **Microservices Architecture** | Martin Fowler – Microservices |
| **Edge Computing** | OpenCV.js Documentation, Progressive Web Apps - Google Developers |
| **ML and AI Ethics** | Google AI Principles, Hugging Face Model Governance |