HealthSync – Real-Time Patient Monitoring System

1. Project Overview

HealthSync is a fictional healthcare technology platform designed to securely collect, process, and analyze real-time patient biometric data from wearable devices. Built entirely on AWS, it is a serverless, scalable, and secure system that prioritizes HIPAA-aligned compliance, proactive alerting, disaster recovery, and AI-driven decision support.

2. Problem Statement

Hospitals and healthcare systems need a cloud-based infrastructure to support continuous patient monitoring using IoT wearables. The architecture must provide real-time alerting, seamless scale, role-based access control, auditability, and advanced analytics capabilities—all while maintaining robust security and compliance standards.

3. Core Requirements

The solution must support real-time ingestion, secure storage, event-driven alerting, role-based access, auditing, and the ability to perform data analysis and predictive modeling on patient vitals.

4. Architecture Overview

The HealthSync platform is composed of the following architecture layers:

- - Ingestion Layer
- - Processing Layer
- - Notification Layer
- - Storage Layer
- - Analytics & Dashboard Layer
- - Global Scale / Multi-Region Disaster Recovery Layer
- - AI/ML Extension Layer
- - CI/CD & Infrastructure as Code Layer

5. Detailed Architecture Documentation

Business Scenario:

HealthSync is a fictional healthcare technology company working with hospitals to develop a cloud-native remote patient monitoring system. The system collects real-time data from wearable devices (e.g.,, heart rate, oxygen saturation, body temperature), transmits it securely to the cloud, and:

- Analyzes the data in real time for abnormal readings
- Notifies clinicians instantly if vitals cross risk thresholds
- Stores data historically for trend analysis and compliance
- Allows authorized users (nurses, physicians, admins) to access dashboards and reports

HealthSync must ensure HIPAA-grade security, handle thousands of devices streaming concurrently, and scale effortlessly without manual infrastructure management.

Core Requirements:

- Real-time ingestion of vital sign data from wearable devices
- Event-driven alerting when thresholds are breached (e.g.,, HR > 160 bpm)
- Secure storage of all data with long-term retention
- Role-based access for clinicians (nurses vs doctors vs admins)
- Monitoring and audit logs for compliance
- Data lake + analytics layer for historical trends (e.g.,, average vitals per week)

The architecture will include:

Ingestion Layer

The ingestion layer is responsible for securely receiving and streaming biometric data (heart rate, oxygen levels, temp, etc.) from thousands of wearable IoT devices to the backend, in near real-time.

This layer must:

- Handle high throughput from distributed devices
- Accept data in various formats (JSON via MQTT or REST)
- Ensure secure transmission and identity verification
- Route the data into a processing pipeline without loss

To meet these requirements, we use AWS IoT Core as the primary entry point for wearable devices, which communicate using the lightweight MQTT protocol. IoT Core provides device identity management, secure TLS communication, and rule-based routing. From there, the data is streamed into Amazon Kinesis Data Streams, which acts as a highly scalable, durable, and low-latency streaming pipeline for ingesting and buffering biometric data. For devices or apps that do not support MQTT, we offer an alternate ingestion path using Amazon API

Gateway combined with AWS Lambda, enabling HTTPS clients to send JSON payloads that are then validated and forwarded to the same Kinesis stream. This hybrid approach ensures compatibility across different device types while maintaining a unified backend stream.

We evaluated Amazon SQS, but it is better suited for message decoupling than continuous real-time telemetry. Amazon MSK (Kafka) was also considered for its robust streaming capabilities, but it requires more operational overhead and is unnecessary for our serverless-first approach. Relying solely on API Gateway and Lambda for ingestion was rejected due to scalability concerns and higher execution costs for continuous streaming. Finally, AWS IoT Events offers event automation, but was deemed more useful for industrial workflows, making it excessive for our simpler patient telemetry ingestion use case.

All device connections to AWS IoT Core are secured using either X.509 certificates or Amazon Cognito identities, enforcing strict authentication for every device. Data streamed into Kinesis is encrypted at rest using AWS KMS, and all interactions are governed by least-privilege IAM roles. The API Gateway + Lambda path also includes request validation and IAM-authenticated execution. This layered security approach ensures all inbound data is protected from origin to stream.

Processing Layer

The processing layer in the HealthSync architecture is responsible for executing real-time analysis and decision-making logic based on incoming biometric data. Once patient vitals are ingested via AWS IoT Core and streamed through Amazon Kinesis, this layer validates the data, checks for abnormalities (such as elevated heart rate or low oxygen levels), and triggers the appropriate downstream actions. These actions may include sending immediate clinician alerts, initiating conditional workflows, or simply logging data for audit and trend analysis. This layer ensures the HealthSync platform reacts intelligently and rapidly to changes in patient health in a secure, scalable, and event-driven manner.

The core service in this layer is AWS Lambda, which is triggered by records arriving in Amazon Kinesis Data Streams. Each Lambda function decodes and parses the incoming JSON data, validates the contents, and checks the values against defined medical thresholds (e.g.,, heart rate > 160 bpm or oxygen < 85%). If an anomaly is detected, the Lambda function publishes a structured event to Amazon EventBridge, which acts as a centralized, schema-aware event bus for routing different types of health alerts.

For more complex or stateful alerting logic (e.g.,, waiting 10 minutes and rechecking before notifying a doctor), EventBridge forwards relevant events to AWS Step Functions. Step Functions orchestrate multi-step workflows, allowing the platform to handle conditional logic, timers, retries, and escalations with full visibility. For example, repeated elevated heart rate readings over a fixed time window could result in a more urgent alert.

Events that are confirmed to require immediate attention are then routed (either directly from EventBridge or via Step Functions) to an Amazon SNS topic that pushes out email or

SMS alerts to healthcare staff. This ensures real-time notifications can reach the appropriate responders, such as nurses or physicians, in critical care scenarios.

In parallel, all Lambda executions and their outcomes are logged into Amazon CloudWatch Logs, ensuring full observability for debugging, auditing, and compliance review.

We considered using Kinesis Data Analytics (Apache Flink) to process the streams directly in Kinesis, but its SQL/Java SDK was unnecessarily complex for our lightweight rule-based checks. AWS Glue streaming jobs were also evaluated but are more suitable for ETL use cases feeding data lakes, rather than real-time alerting. Relying solely on SNS instead of Amazon EventBridge was also explored, but SNS lacks event filtering and schema validation — making EventBridge a better fit for modular and scalable workflows. Lastly, we considered centralizing all logic in one large Lambda, but this would have tightly coupled unrelated rules and hurt maintainability and deployment flexibility.

All Lambda functions operate under unique IAM roles following the principle of least privilege — for instance, the processing function can read from Kinesis and publish only to EventBridge. Step Functions and EventBridge are also governed by scoped IAM policies ensuring that only approved publishers and subscribers can access the event bus. Data processed by Lambda functions is encrypted in transit, and environment variables are encrypted using AWS KMS. Each function logs to CloudWatch with metadata tags (e.g.,, patient ID, condition triggered) for better traceability. AWS X-Ray is enabled for distributed tracing of all Lambda and Step Function executions, providing visibility into execution paths and performance bottlenecks.

Together, these measures ensure that the processing layer not only delivers responsive, real-time analysis but also adheres to strong security and compliance practices suitable for handling sensitive healthcare data.

Notification Layer

The Notification Layer in HealthSync serves a mission-critical function: ensuring that when a patient's biometric data crosses a defined danger threshold, the appropriate healthcare staff are notified immediately and reliably. After processing and verifying an alert-worthy event in the previous layer, this component is responsible for delivering that event to the right communication channels — whether it's a nurse on duty, an emergency response team, or a doctor assigned to that patient. It must support both low-latency and high-reliability alerting methods, with delivery guarantees and the flexibility to support various message formats.

The heart of this layer is Amazon Simple Notification Service (SNS). SNS allows HealthSync to publish a single alert to multiple recipients, whether via email, SMS, or application endpoints, with near-instant delivery. SNS topics are created for various alert levels or departments (e.g.,, CriticalVitalsAlertTopic), and subscribers (nurses, physicians, or internal apps) can be dynamically attached to receive messages in real time. This ensures alerts are disseminated broadly and without delay.

For more customized or rich HTML-format emails (e.g.,, including patient name, time of alert, vitals trend charts), we can optionally integrate Amazon Simple Email Service (SES). SES allows us to send branded, structured emails directly from Lambda or Step Functions, particularly useful for non-urgent summaries or detailed notifications to care teams.

In the future, HealthSync can also leverage AWS Chatbot via Lambda triggers to push alert messages directly into Slack channels or Microsoft Teams, enabling real-time collaboration for clinical response teams. This integration would allow clinicians to acknowledge alerts, triage incidents, or coordinate actions directly within their messaging tools.

We considered using Amazon Pinpoint for alert delivery, which provides multi-channel messaging, but it is better suited for marketing and engagement use cases rather than operational or emergency alerting. Similarly, using custom Lambda functions to send SMTP email directly was evaluated, but dismissed due to the complexity of managing secure SMTP credentials, monitoring delivery, and scaling. While third-party alerting systems like PagerDuty or Opsgenie offer excellent incident management, they were deemed out of scope for this core AWS-native prototype — though easy to integrate later if needed.

Security and reliability are critical in this layer, as these messages could contain protected health information (PHI). All SNS topics are secured using IAM policies, ensuring only specific services like EventBridge or Step Functions can publish to them. Message payloads can be encrypted at rest using AWS KMS, and access to subscription endpoints (e.g.,, Lambda or HTTPS) can be restricted using token-based authentication or AWS Signature v4. All delivery attempts and successes/failures are tracked and sent to Amazon CloudWatch Logs, allowing for full auditability. Additionally, CloudWatch Alarms can be configured to monitor SNS delivery failures or high error rates, ensuring system reliability and compliance with healthcare uptime expectations.

Storage Layer

The Storage Layer in HealthSync is responsible for persistently storing all patient biometric data, both for immediate retrieval in active monitoring scenarios and for long-term archival and regulatory compliance. It supports rapid lookups for dashboards and clinician access, while also providing scalable, cost-efficient storage for historical data analysis and legal retention. This dual role makes the storage layer a foundational part of HealthSync's architecture, ensuring both real-time visibility and data integrity over time.

For fast and flexible storage of incoming vitals, we use Amazon DynamoDB as the primary operational database. It offers low-latency read/write performance with serverless scalability, making it ideal for recording time-stamped patient data (e.g.,, heart rate, temperature) keyed by patient ID and timestamp. Each record generated from the processing layer is written into DynamoDB tables for real-time querying and trending.

For batch exports, raw telemetry, logs, and archived JSON payloads, we use Amazon S3. This object-based storage service acts as the system's cold storage and data lake foundation, supporting large-scale ingestion and lifecycle management. Data stored in S3 is later

analyzed through AWS analytics tools like Athena or Glue, enabling insights without affecting the production database.

To meet long-term compliance requirements (such as HIPAA's medical record retention policies), Amazon S3 Glacier is used in conjunction with S3 lifecycle rules. After a defined retention window (e.g.,, 90 days), older patient records are automatically transitioned to Glacier for low-cost, long-term archival.

Optionally, we integrate AWS Glue Data Catalog to organize data stored in S3. This makes it queryable through Athena and allows downstream tools to understand schema, metadata, and table formats without manual indexing.

We considered Amazon RDS (e.g.,, Aurora MySQL/Postgres) for structured storage, but DynamoDB was preferred due to its simpler scaling model, lower latency, and cost-effectiveness for high-volume key-value access patterns. While Amazon Timestream is optimized for time-series workloads, it introduces complexity and cost, and doesn't align as well with our need for patient-centric partitioning. Amazon OpenSearch (formerly Elasticsearch) was also evaluated but deemed overkill for structured vitals storage — it's more appropriate for full-text or filtered search use cases. Storing everything directly in S3 was also ruled out for operational reads due to S3's slower lookup times and lack of queryable indexing.

All data at rest in Amazon DynamoDB and Amazon S3 is automatically encrypted using AWS Key Management Service (KMS). DynamoDB tables are configured with Point-in-Time Recovery (PITR) enabled, allowing rollback to any moment within the past 35 days — crucial for operational resilience and data integrity. IAM roles assigned to Lambda functions are scoped strictly to the relevant table or bucket, ensuring least privilege access. S3 buckets have public access blocked by default and can be restricted further using bucket policies and VPC endpoints to prevent exposure to the public internet. Versioning and MFA delete can be enabled on critical buckets to protect against accidental deletion. Furthermore, CloudTrail logs all access events across the storage layer, allowing auditing of data access, mutation, and retention actions. Lifecycle policies in S3 not only optimize cost but also ensure compliance by automatically transitioning or deleting records based on retention schedules.

Analytics Layer

The Analytics Layer in HealthSync is designed to turn stored biometric data into actionable insights for clinical staff, data analysts, and healthcare administrators. While the ingestion and processing layers focus on real-time detection and alerts, the analytics layer provides tools to analyze historical trends, identify patterns, and generate reports and visual dashboards. For instance, a doctor could view a patient's average heart rate over the past 30 days, or a hospital administrator could analyze region-wide oxygen level trends to allocate emergency response resources. This layer supports both ad-hoc querying and scheduled reporting, bridging the gap between raw data and decision-making.

At the core of this layer is Amazon Athena, a serverless, pay-per-query SQL engine that allows analysts and dashboards to query historical vitals data directly from Amazon S3. Athena uses AWS Glue Data Catalog to understand the structure and schema of the JSON data stored in S3, enabling fast, schema-on-read querying without loading data into a database. This makes Athena ideal for cost-effective querying across years of patient telemetry without maintaining infrastructure.

For interactive data visualization and sharing, we integrate Amazon QuickSight, which connects directly to Athena as a data source. QuickSight enables the creation of real-time dashboards, charts, and reports that are easily shared with medical staff and hospital administrators. Dashboards can show KPIs like "average vitals by region," "patients with repeated high heart rates," or "daily alert volumes."

Behind the scenes, AWS Glue provides optional ETL capabilities. If HealthSync needs to transform or enrich data (e.g.,, aggregate vitals by time interval, join datasets from multiple sources, clean malformed records), Glue jobs can run on a schedule or trigger based on new S3 data. Glue ensures data pipelines remain automated and scalable.

We considered Amazon Redshift for the analytics backend, but its cluster-based model introduces ongoing cost and administrative overhead, which is unnecessary for the current volume and query patterns. Amazon Timestream is optimized for time-series analysis but lacks the flexibility for wide ad-hoc querying across multiple data types and is less cost-effective for batch analytics over S3 data. Tools like OpenSearch (formerly Elasticsearch) were also evaluated but are better suited for log aggregation and search, not structured analytics and dashboarding. Athena and QuickSight provide a more modular, serverless, and cost-optimized approach for HealthSync's analytical needs.

All Athena queries are executed using IAM roles that define which users or applications can access specific tables, columns, and S3 buckets. The S3 data queried by Athena is encrypted at rest using KMS, and QuickSight access can be restricted to specific dashboards or data sources using row-level and column-level security. AWS Glue Data Catalog is protected via resource-based IAM policies to ensure only authorized services can access or modify schema definitions. All analytics-related activity (e.g.,, query runs, dashboard views, schema changes) is logged via CloudTrail and can be monitored through CloudWatch Logs and Metrics. Additionally, data egress from QuickSight or Athena can be monitored and restricted using VPC endpoints or S3 bucket policies, ensuring PHI doesn't leave the AWS environment unintentionally.

Global Scale / Multi-Region Disaster Recovery Layer

The Global Scale and Disaster Recovery layer enables HealthSync to operate as a highly available, globally distributed platform. In healthcare environments, even short outages can lead to missed alerts or delayed interventions, so maintaining zero-downtime access is critical. This layer ensures the platform can automatically route users to the nearest or healthiest AWS region based on latency or system health, and that all patient data is actively replicated between geographically separated regions. It also lays the foundation for

supporting global expansion, enabling the platform to serve users across different regions with minimal latency and maximum resiliency.

To route users efficiently, Amazon Route 53 is used with latency-based routing and health checks. This setup ensures that end-user requests are sent to the closest AWS region (e.g.,, North Virginia or Oregon), and in the event of a failure, automatically rerouted to a secondary region. This provides both performance optimization and automated failover.

For data replication, Amazon DynamoDB Global Tables provides multi-region, active-active replication of patient vitals data. It ensures that any update in one region is immediately available in another, supporting seamless continuation of patient monitoring in a failover scenario. Similarly, Amazon S3 Cross-Region Replication (CRR) is enabled on vitals archives, logs, and batch files to replicate data from the primary S3 bucket to a backup in a different region.

To maintain consistent infrastructure across all regions, we use AWS CloudFormation StackSets, which allow centralized deployment and updates of infrastructure templates (e.g.,, Lambda functions, IAM roles, API Gateway configurations) into multiple regions from a single control point.

We evaluated several approaches before choosing this fully automated multi-region design. A manual failover model — where engineers would trigger region switchovers and data syncs during a failure — was rejected due to its operational risk, lack of real-time response, and poor user experience. We also considered a warm standby setup, where a secondary region remains mostly idle and gets activated only during emergencies. While this is more cost-efficient, it introduces risks around data freshness and recovery time, which is unacceptable in a healthcare monitoring context.

Finally, we considered implementing multi-region workloads via Kubernetes clusters spanning multiple regions. While Kubernetes provides great control over orchestration, it introduces complexity and infrastructure management responsibilities that conflict with HealthSync's serverless-first design principles. It would also require maintaining container runtimes, node groups, and networking — all of which increase operational overhead.

All cross-region communications and data replications are encrypted in transit and at rest using AWS Key Management Service (KMS). Separate KMS keys are used per region, and replica keys are set up to support S3 CRR and DynamoDB encryption. Route 53 health checks are configured to monitor secure endpoints (e.g.,, HTTPS APIs) and trigger failover only if validated failures are observed. Each region enforces least privilege IAM roles, and security logs (via AWS CloudTrail) are collected separately per region to ensure independent auditability. AWS Config can also be enabled in both regions to track configuration drifts and enforce compliance standards across infrastructure stacks.

AI/ML Extension Layer

The AI/ML Extension Layer adds predictive intelligence to HealthSync by analyzing historical biometric data to identify early warning signs of patient deterioration before critical thresholds are crossed. Rather than waiting for vitals to spike and then triggering alerts, this layer uses machine learning models to detect patterns of concern — such as a patient's gradual decline over several hours — and notifies clinicians earlier, potentially improving intervention times and outcomes. This layer also enables trend analysis, personalized risk scoring, and can even support recommendations for resource allocation across hospital units based on aggregated data trends. Integrating AI transforms HealthSync from a responsive platform into a proactive clinical decision support system.

At the center of this layer is Amazon SageMaker, which provides a fully managed environment for training, deploying, and managing ML models. HealthSync data scientists or ML engineers can train models (e.g.,, XGBoost or LSTM) using patient vital history from Amazon S3 and Athena queries. Once trained, these models are deployed as SageMaker endpoints that can be invoked from AWS Lambda functions within the processing layer or through EventBridge rules. For example, every time a new batch of vitals is received, a Lambda function could send a payload of recent trends to a SageMaker endpoint and receive a prediction (e.g.,, "high likelihood of cardiac event in 30 minutes").

For low-latency inference and model simplicity, Lambda functions can preprocess data and invoke real-time inference endpoints in SageMaker directly. Amazon CloudWatch is used to monitor model invocation success, latency, and error rates. In future stages, HealthSync could also use SageMaker Model Monitor to detect data drift, ensuring deployed models stay accurate over time.

We considered using AWS Comprehend Medical for health-specific predictions, but it's better suited for unstructured clinical text like doctor notes or EMRs, not structured telemetry data from wearable devices. Amazon Forecast was also evaluated but focuses on time-series demand planning and is less suitable for patient-level anomaly detection. AWS Bedrock and generative AI tools were ruled out for now due to their focus on language models rather than time-series physiological data. For teams looking to avoid direct model development, SageMaker JumpStart could be used to launch prebuilt healthcare-related models — but we opted for full SageMaker control for flexibility and model customization.

Invocations of SageMaker endpoints are restricted using IAM roles assigned to Lambda functions with only sagemaker: InvokeEndpoint permission, scoped to specific endpoints. Model artifacts (stored in S3) are encrypted with AWS KMS, and inference payloads can optionally be encrypted in transit. SageMaker supports VPC endpoint configurations, meaning models can be deployed within a private subnet and not exposed to the public internet. Audit logging of inference requests, errors, and performance metrics is captured via CloudWatch Logs, and AWS CloudTrail records who deployed or modified the model. Fine-grained access to training data in S3 is enforced using bucket policies, and SageMaker

Notebooks (if used) should be isolated to private subnets with multi-factor authentication for model creators.