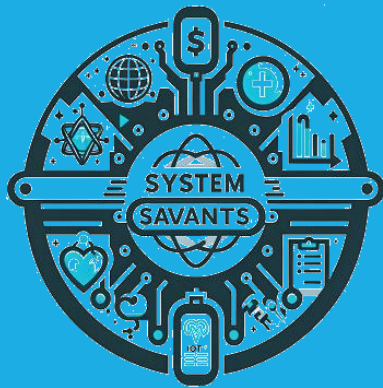


O'REILLY
ARCHITECTURE KATAS
WINTER 2024



THE TEAM



Vishal Gamji



Gibran Castillo



Harshada
Kandalgaonkar



Subodh Gupta



Ch1. Introduction

[Problem Background & Business Goals]



Ch2. Navigating Challenges

[Architecture Analysis]



Ch3. Dueling Architectural & Operational Obstacles [Architecture Decision Records]



Ch4. Eureka!

[The Proposed Solution]



Ch5. Curtains Close

[The End]

OUR STORY ...

CHAPTER 1: INTRODUCTION

PROBLEM BACKGROUND & BUSINESS GOALS



StayHealthy, Inc.: A leading medical software company in San Francisco with two major products: MonitorThem and MyMedicalData.



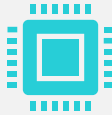
MonitorMe: StayHealthy, Inc. is exploring growth opportunities with MonitorMe. A real-time patient monitoring solution for hospitals, integrating with existing products for enhanced patient care.



The Vision: Disrupt patient monitoring industry with a reliable solution for hospitals that incorporates real-time data analysis, insights, and flexible EHR integration.



Technical Complexity: Developing MonitorMe to handle data from multiple patient-monitoring devices with varying transmit rates and data volumes, ensuring data integrity and low latency.



On-Premises Hosting & Deployment: Packaging and deploying this solution surfaces operational obstacles and introduces a risk element for StayHealthy Inc. as they haven't operated in this business model.



Integration Hurdles: Seamlessly integrating MonitorMe with MyMedicalData for EHR updates, while ensuring data security and patient confidentiality.

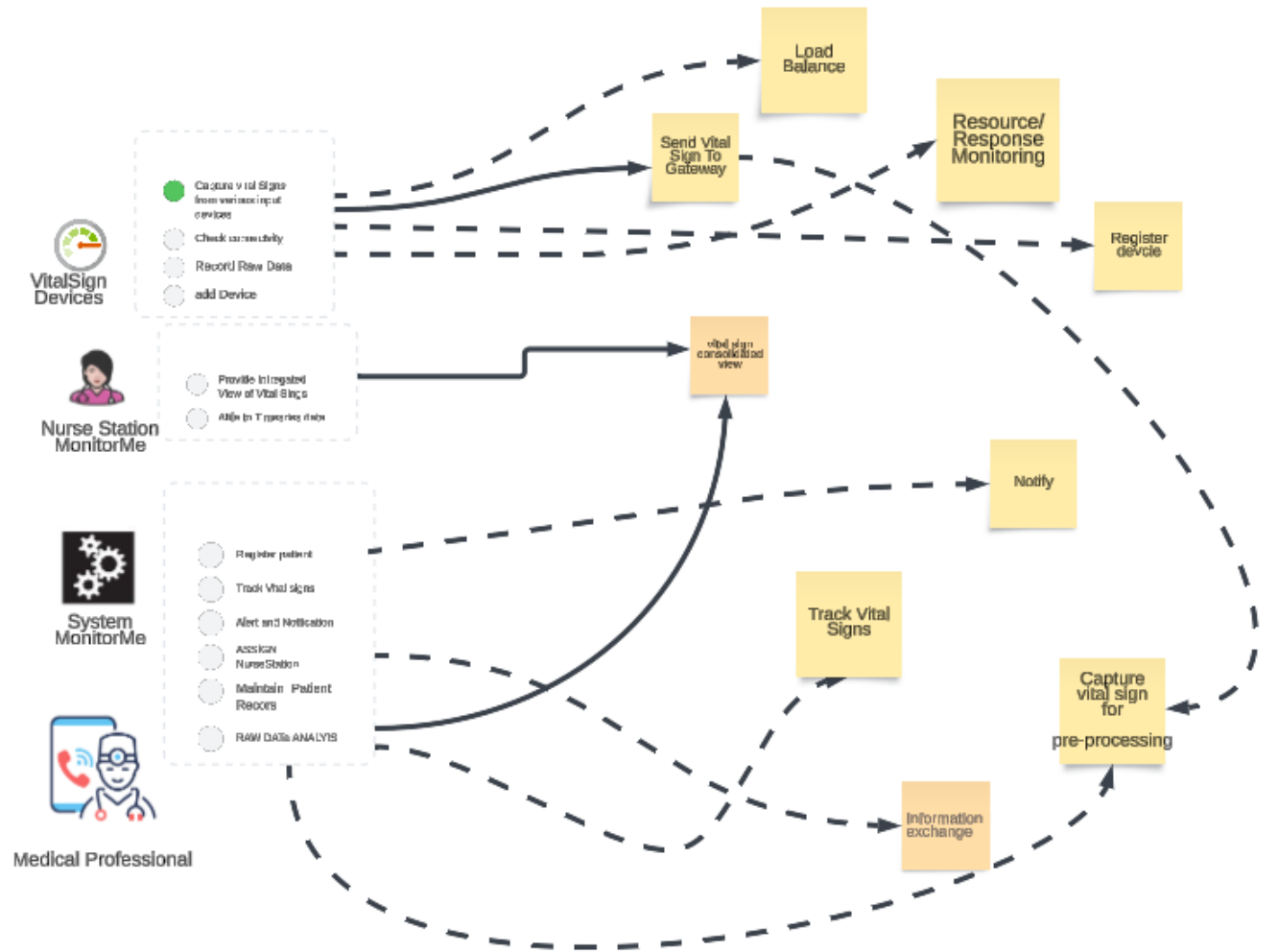
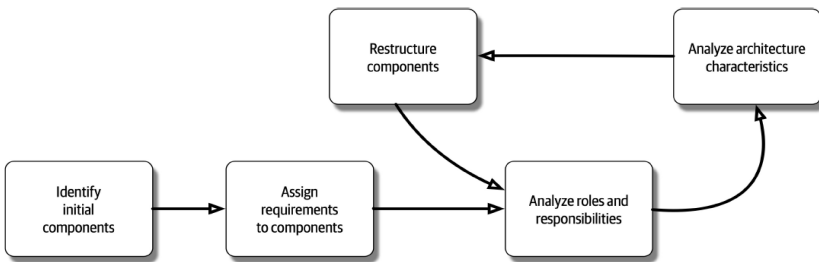


Scalability and Future Expansion: Building a system capable of expanding to accommodate more devices and patients without compromising performance.

CHAPTER 2: NAVIGATING CHALLENGES

ARCHITECTURE ANALYSIS

2.1 ACTORS, ACTIONS, AND COMPONENTS



2.2 ARCHITECTURE CHARACTERISTICS

Architecture Characteristics Worksheet

System/Project: MonitorMe Domain: Health Care
Architect/Team: Systems Savants Date: February 17th, 2024

Candidate Architecture Characteristics

performance	data integrity	deployability
responsiveness	data consistency	testability
availability	adaptability	abstraction
fault tolerance	extensibility	workflow
scalability	interoperability	configurability
elasticity	concurrency	recoverability

others: _____

- ^a denotes characteristics that are related; some systems
^b only need one of these, other systems may need both

Top 3 Driving Characteristics

- ☒ [Real-Time] Performance
- ☒ High Availability
- ☒ Interoperability
- ☐ Scalability
- ☐ Deployability
- ☐ Data Integrity
- ☐

Instructions

- Identify no more than 7 driving characteristics.
- Pick the top 3 characteristics (in any order).
- Implicit characteristics can become driving characteristics if they are *critical* concerns.
- Add additional characteristics identified that weren't deemed as important as the list of 7 to the *Others Considered* list.

Implicit Characteristics

usability
security
maintainability
observability

Others Considered

Extensibility
Reliability
Elasticity
Configurability

2.3 CAPACITY PLANNING

MonitorMe Sytem Storage Capacity Planning

Inputs/Config	
Num Instances	1
NumPatients	500
Num Stations	25
NumDevices	8

		Est. rate of event writes/captures from devices			
Vitals	Rate	Per Sec	Per Min	Per Hr	Per Day (24h)
HeartRate	500ms	2	120	7200	172800
Blood Pressure	1hr	-	-	1	24
Oxygen Level	5s	-	12	720	17280
Blood Sugar	2 min	-	-	30	720
Respiration	1s	1	60	3600	86400
ECG	1s	1	60	3600	86400
Body Temperature	5min	-	-	12	288
Sleep Status	2min	-	-	30	720
Events Per Patient			253.22	15,193	364,632

# Vital Events (24hrs)	182,316,000	Events
# Vital Events Per Station (24hrs)	3,646,320,000	Events
Est. Storage per event	1	KB
Required Storage (KB)	182,316,000	KB
Required Storage (MB)	178,043	MB
Required Storage (GB)	174	GB
Required Storage (TB)	0.2	TB
Est. Misc. Storage (GB)	0.04	TB
Daily Required Storage (GB)	217.3	GB
Daily Required Storage (TB)	0.2	TB

Input/Config

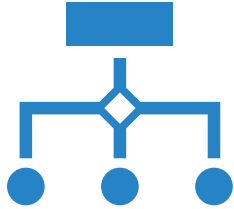
*Event storage

*Other operational storage @ 25% of required storage.

	 layered	 modular monolith	 microkernel	 microservices	 service-based	 service-oriented	 event-driven	 space-based
agility	★	★★	★★★	★★★★★	★★★★★	★	★★★	★★
abstraction	★	★	★★★	★	★	★★★★★	★★★★★	★
configurability	★	★	★★★★★	★★★★	★★	★	★★	★★
cost	★★★★★	★★★★★	★★★★★	★	★★★★★	★	★★★	★★
deployability	★	★★	★★★	★★★★★	★★★★★	★	★★★	★★★★
domain part.	★	★★★★★	★★★★★	★★★★★	★★★★★	★	★	★★★★★
elasticity	★	★	★	★★★★★	★★	★★★	★★★★	★★★★★
evolvability	★	★	★★★	★★★★★	★★★★	★	★★★★★	★★★★
fault-tolerance	★	★	★	★★★★★	★★★★	★★★	★★★★★	★★★★
integration	★	★	★★★	★★★★	★★	★★★★★	★★★	★★
interoperability	★	★	★★★	★★★	★★	★★★★★	★★★	★★
performance	★★★	★★★	★★★	★★	★★★★	★★	★★★★★	★★★★★
scalability	★	★	★	★★★★★	★★★★	★★★	★★★★★	★★★★★
simplicity	★★★★★	★★★★★	★★★★★	★	★★★★	★	★	★
testability	★★	★★	★★★	★★★★★	★★★★★	★	★★	★
workflow	★	★	★★	★	★	★★★★★	★★★★★	★

2.4 ARCHITECTURE STYLE SELECTION

MICROSERVICES
+
EVENT-DRIVEN



Architectural Decisions: Adopting a microservice and event-driven architecture to ensure scalability, fault tolerance, and real-time performance.



Security and Compliance: Implementing layered security measures and achieving compliance with health data regulations without governmental requirements.



Interoperability and Data Integrity: Ensuring the system works flawlessly with existing hospital infrastructure and maintains high data accuracy for life-critical decisions.

CHAPTER 3: DUELING ARCHITECTURAL & OPERATIONAL OBSTACLES

3.1 ARCHITECTURE DECISION RECORD

HIGHLIGHT 1 ADR-001

001 - Use K8s with containerize microservice architecture style

Date: 2024-02-22

Status

Proposed

Context

Microservices architecture goes well with small easily deployable units, which could be individually scaled to handle the load efficiently as process of scaling is quite simple. Each microservice's requirement has a limited scope and could be containerized. All the microservices are not required to deploy together, and changes could be implemented separately and as frequently. This feasibility of developing both separately is something that helps in the future as well for monitoring the performance of all the microservices. then it is going to be pretty easy for you to make the right decision when required.

The architectural decisions made on this project must be recorded in a useful and comprehensible manner.

Decision

We will use the microservice architecture style for backend services in the MonitorMe system, with AWS EKS as the tool of choice in a high availability and auto scale setup.

Consequences

Microservices will be containerized using AWS EKS. Microservices architecture facilitates

1. Scalability: Each microservice can be scaled independently. Ex: Heart rate service with the most events can be scaled independently.
2. Partition Tolerant: The system requirement of being partition tolerant is satisfied. An outage of one microservice does not impact the overall system.
3. Maintainance and Modularity: Each microservice can be modified, developed, maintained, and upgraded independently.

Positive: Scalability, Resiliency, Maintainability, Flexibility, Security (granular security as needed)

Negative: The challenge with increased flexibility is the need to establish some design guidelines and design standards for microservice development. Since it's new medical patient monitoring system it should to define and enforce (via governance mechanisms) a basic reference architecture (with room for evolution).

3.2 ARCHITECTURE DECISION RECORD

HIGHLIGHT 2 ADR-002

002 - Use API Gateway in self-hosted mode

Date: 2024-02-22

Status

Accepted

Context

Here we were looking for a component that will be the one stop solution for any data coming in or going out of the MonitorMe ecosystem. Data going out of MonitorMe would be snpshot uploads to MyMedicalData. Alerts and push notifications going out to StayHealty Mobile App. Incoming data in future could be from MyMedicalData or from MonitorThem. Options Considered - Ambassador api gateway , Mulesoft, Axway

Decision

We decided to with Ambassador API gateway as it provides a secure and real-time communication option without having to provision or manage any servers to manage connections or large-scale data exchanges. Given the requirement for integrating with EHR systems securely is via secure HTTP ,API Gateway made more sense.

Consequences

Positive:

- Ease of setup and maintainability.
- Security

3.3 ARCHITECTURE DECISION RECORD

HIGHLIGHT 3 ADR-003

003 - Hosting Platform On Prem - AWS Outposts Servers

Date: 2024-02-22

Status

Accepted

Context

The requirement is to host the solution locally on-premises in hospitals.

Decision

The decision is to use AWS Outpost Servers in a high availability mode. AWS Outposts is a family of fully managed solutions delivering AWS infrastructure and services to virtually any on-premises or edge location for a truly consistent hybrid experience. Outposts solutions allow you to extend and run native AWS services on premises, and is available in a variety of form factors, from 1U and 2U Outposts servers to 42U Outposts racks, and multiple rack deployments.

With AWS Outposts, you can run some AWS services locally and connect to a broad range of services available in the local AWS Region. Run applications and workloads on premises using familiar AWS services, tools, and APIs. Outposts supports workloads and devices requiring low latency access to on-premises systems, local data processing, data residency, and application migration with local system interdependencies.

Consequences

The requirement is met along with the ability to run cloud-native technologies on-premises. This allows for local data processing which is a core requirement.

Positive:

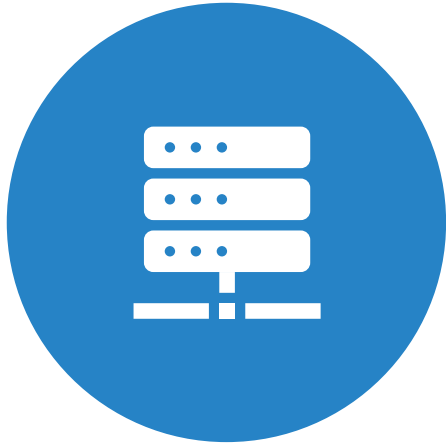
- Access to cloud-native technology
- Faster time to market
- Local data storage and processing
- Low latency data processing

Negative:

- Potential cost
- Disaster recovery limited to hospital infrastructure

Risks:

- Capacity management



ARCHITECTURE & DEPLOYMENT SHOWCASE: ON-PREMISES IMPLEMENTATION AT HOSPITAL LOCATIONS WITH AWS OUTPOSTS SERVERS, ENSURING DATA PRIVACY AND LOCAL PROCESSING NEEDS.



REAL-TIME MONITORING: WITH AN AVERAGE RESPONSE TIME OF LESS THAN A SECOND, MONITORME REVOLUTIONIZES PATIENT CARE, ENABLING HEALTHCARE PROFESSIONALS TO RESPOND SWIFTLY TO PATIENT NEEDS.



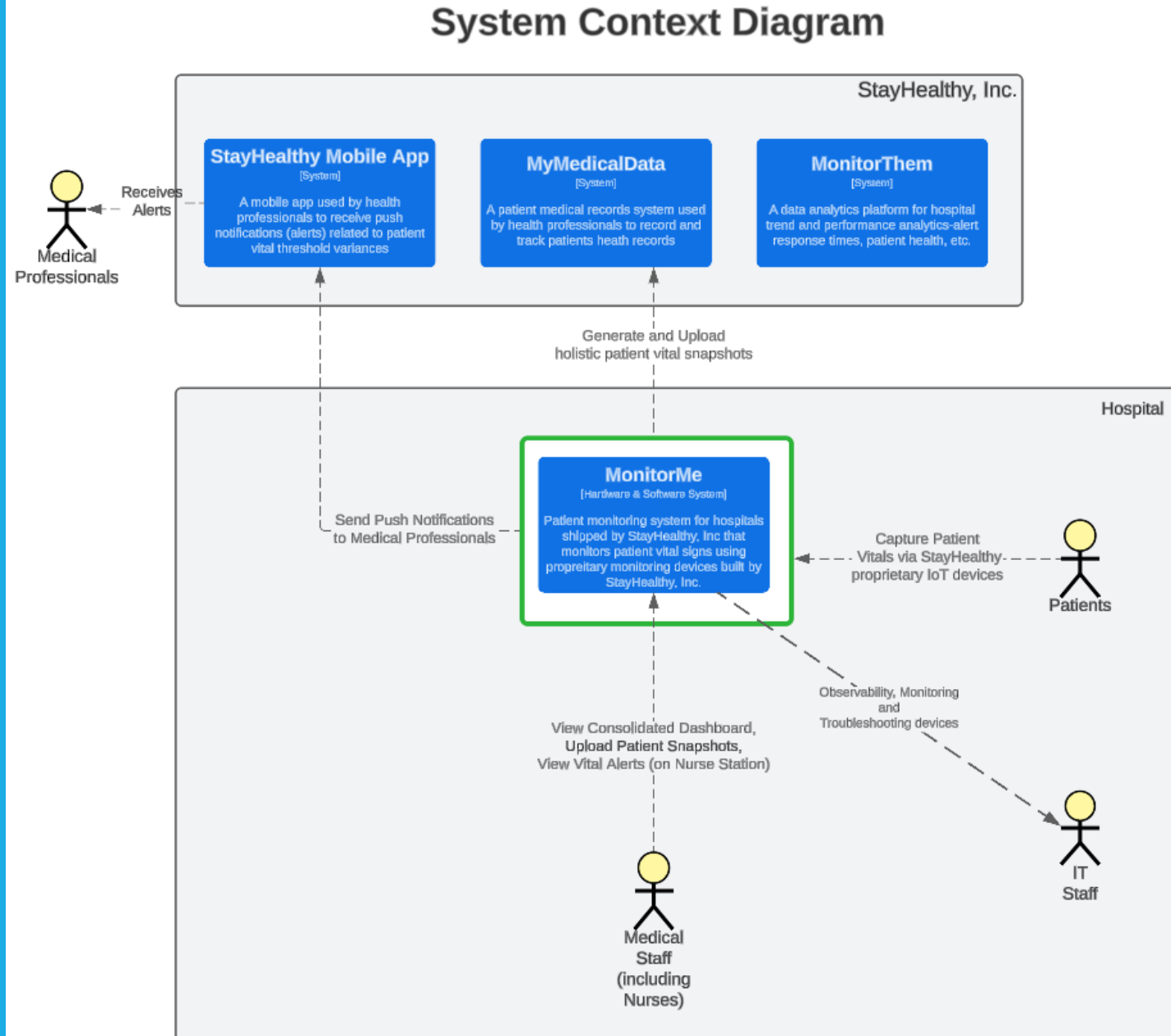
FUTURE-PROOF AND SCALABLE: DESIGNED TO ACCOMMODATE ADDITIONAL MONITORING DEVICES AND INTEGRATE WITH EVOLVING MEDICAL SOFTWARE LANDSCAPES, ENSURING LONG-TERM VIABILITY.

CHAPTER 4: THE EUREKA MOMENT

[THE PROPOSED SOLUTION]

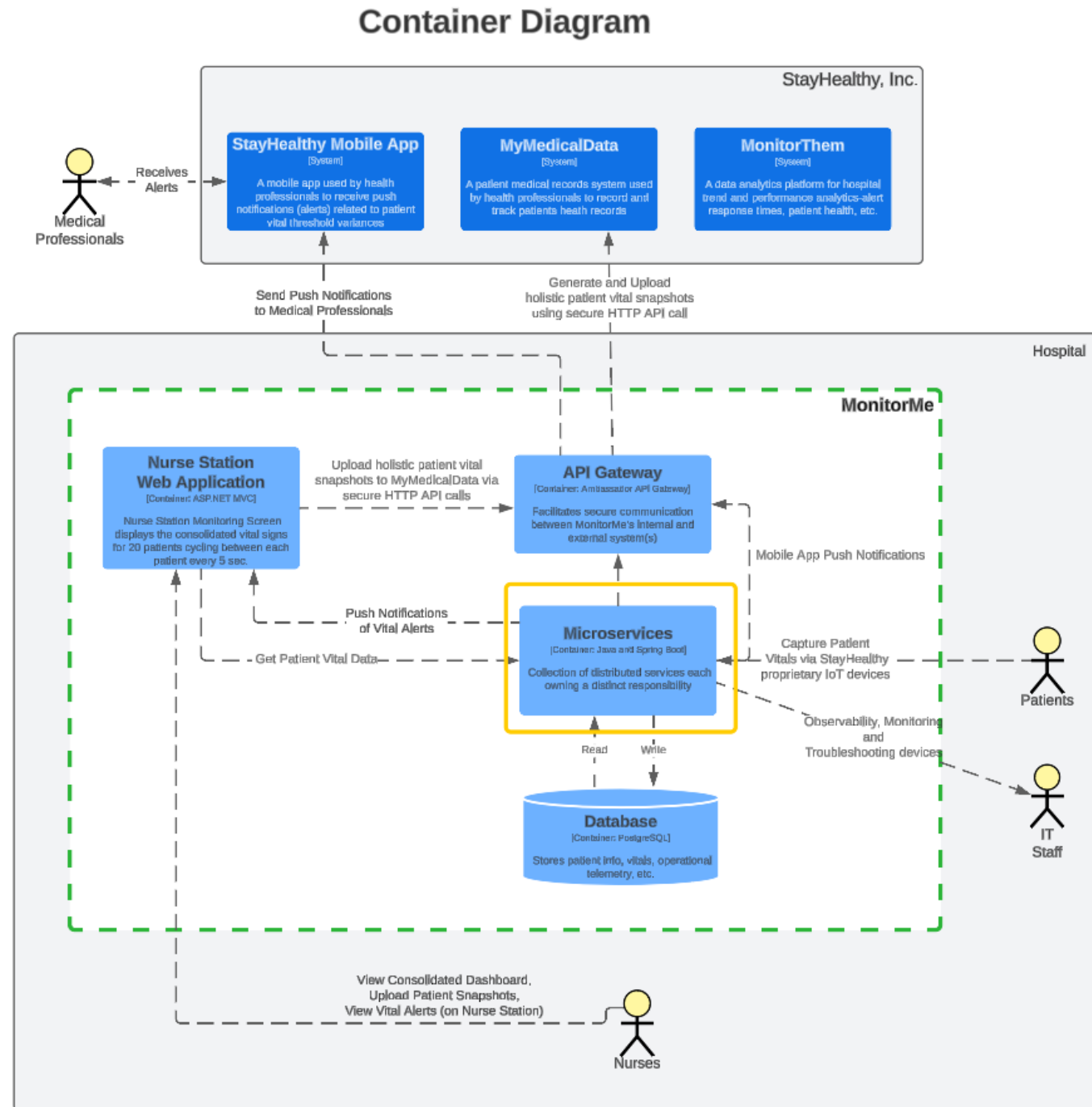
4.1 HIGH LEVEL ARCHITECTURE

C4 MODEL SYSTEM CONTEXT DIAGRAM (C1)



4.2 HIGH LEVEL ARCHITECTURE

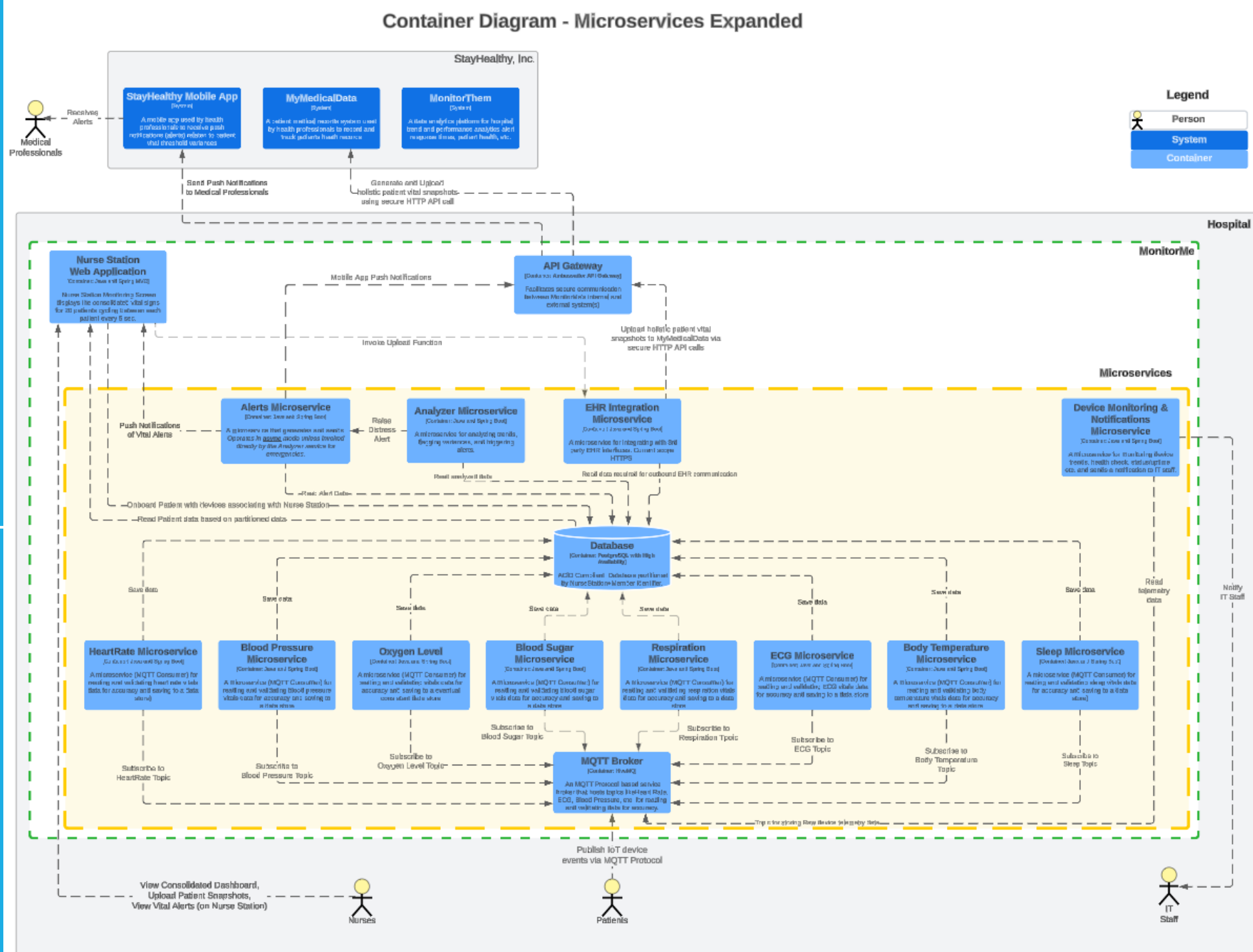
C4 MODEL CONTAINER DIAGRAM (C2)



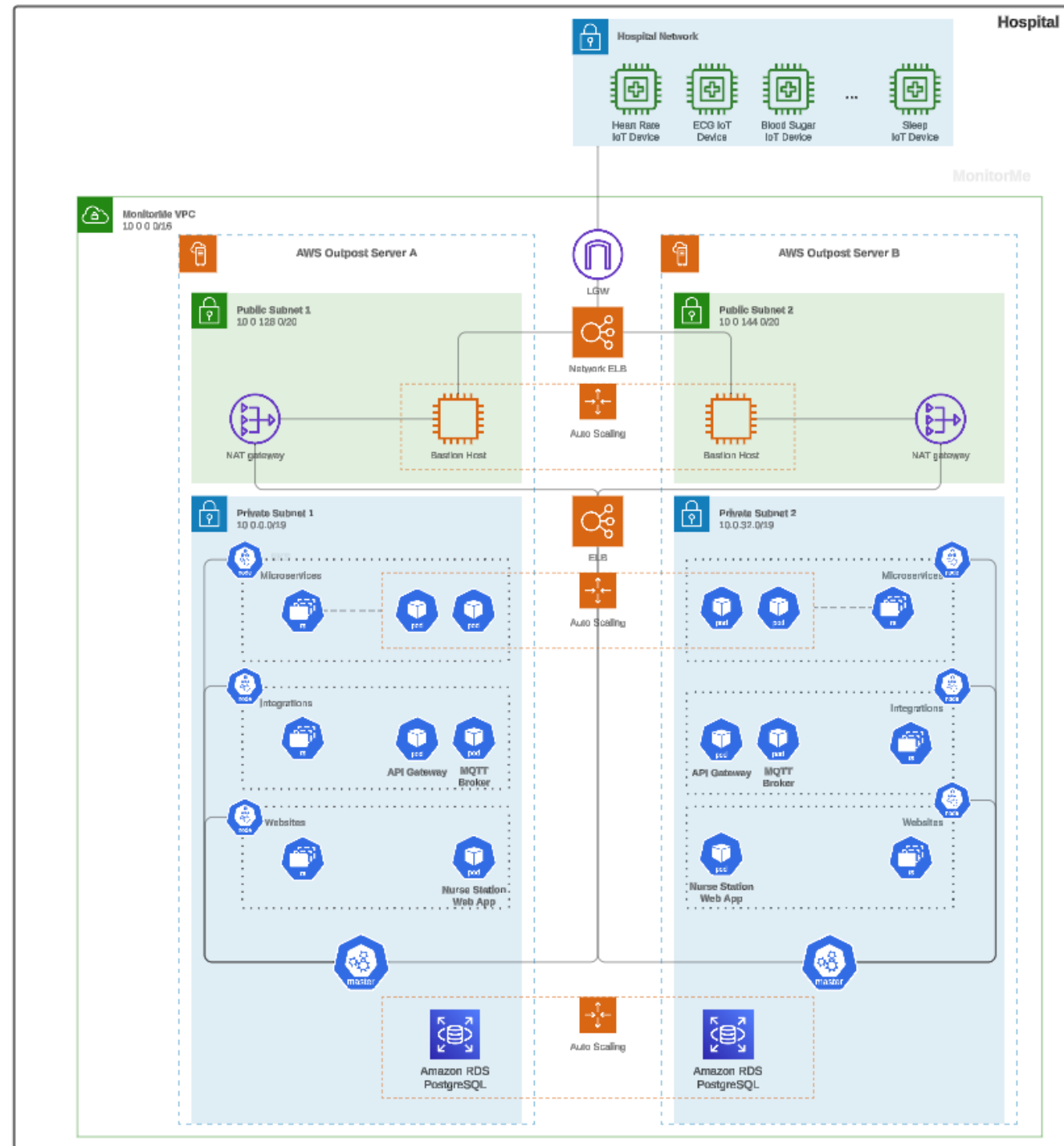
C4 MODEL

CONTAINER DIAGRAM (C2)

MICROSERVICES EXPANDED



4.4 DEPLOYMENT DIAGRAM





INNOVATIVE SOLUTION: MONITORME REPRESENTS A SIGNIFICANT LEAP FORWARD IN PATIENT CARE, COMBINING REAL-TIME MONITORING WITH ROBUST DATA ANALYSIS AND SECURE EHR INTEGRATION.



IMPACT ON HEALTHCARE: ENHANCING PATIENT OUTCOMES, REDUCING HEALTHCARE COSTS, AND PAVING THE WAY FOR DATA-DRIVEN HEALTHCARE SOLUTIONS.



STAYHEALTHY, INC.'S COMMITMENT: CONTINUING TO INNOVATE AND SUPPORT THE HEALTHCARE INDUSTRY WITH SOLUTIONS THAT SAVE LIVES AND IMPROVE PATIENT CARE.

CHAPTER 5: THE END

*Credits: Dall-E generated.



5.1 NURSE STATION DASHBOARD VIEW

*Credits: Dall-E generated.



5.2 MEDICAL PROFESSIONAL MOBILE APP VIEW