Azure Environment Audit

Engagement Deliverable Pack

Northwind Health
February 2026

Introduction

This document pack represents the complete deliverables from a comprehensive Azure environment audit engagement. The engagement assessed three Azure subscriptions encompassing production, development, and shared services workloads.

The primary objectives of this audit were to:

- Establish governance clarity through consistent naming and tagging standards
- Identify cost optimization opportunities with actionable implementation guidance
- Document architecture dependencies to support operational decision-making
- Provide a foundation for ongoing Azure governance maturity

The materials enclosed are designed to serve as both a reference and a roadmap for improving the security, efficiency, and manageability of your Azure environment.

What This Engagement Covered

The audit encompassed four key areas:

- **Governance** — Resource organization, naming conventions, tagging standards, and ownership accountability
- **Cost Optimization** — Spending analysis, waste identification, and prioritized savings opportunities
- **Architecture Dependencies** — Application relationships, data flows, and single points of failure
- **Operational Maturity** — Change control procedures, safety guardrails, and audit methodology

---

What You Will Find in This Pack

01 — Executive Summary
A leadership-level overview of key findings, risks, and recommended next steps. Designed for stakeholders who need the strategic picture without technical detail.

02 — Naming Standards and Governance
A complete naming convention and tag governance framework ready for immediate adoption. Includes patterns for all common Azure resource types, required tags, and enforcement guidance.

03 — Cost Analysis and Quick Wins
A detailed breakdown of current Azure spending with ten prioritized optimization opportunities. Each recommendation includes estimated savings, effort level, and implementation steps.

04 — Dependency Map Explanation
A visual architecture diagram illustrating how applications, data services, and infrastructure components interconnect. Includes guidance on interpreting the diagram and assessing resilience.

05 — Audit Methodology and SOP
Documentation of how the audit was performed and the change control procedures that ensure safe implementation of recommendations. Provides transparency and a repeatable process for future assessments.

Intended Use

This deliverable pack is designed to support multiple audiences:

Leadership — Use the Executive Summary and Cost Analysis to understand the strategic impact and prioritize investment decisions.

Engineering Teams — Use the Naming Standards and Dependency Map as reference documentation for new deployments and architecture decisions.

Operations Teams — Use the Audit Methodology and SOP to guide safe execution of changes and establish governance processes for ongoing maintenance.

These documents are living references. As your Azure environment evolves, they should be updated to reflect architectural changes, new services, and lessons learned from implementation.

Closing

This engagement deliverable pack provides a clear picture of the current state of your Azure environment, along with practical guidance for improving governance, reducing cost, and managing risk.

The recommendations are designed to be implemented incrementally, starting with low-effort quick wins and progressing to more substantial optimizations as time and resources allow. Each change follows documented safety procedures to protect production workloads.

We appreciate the opportunity to support Northwind Health in strengthening its Azure foundation.

Prepared by Azure Governance Consulting

Executive Summary

Azure Environment Audit — Northwind Health

Engagement Overview

This document summarizes the findings from a comprehensive audit of Northwind Health's Azure environment, conducted in January–February 2026. The assessment covered three subscriptions (Production, Development, and Shared Services) containing approximately 45 resources across compute, data, networking, security, and monitoring services.

The audit focused on four key areas:

- Resource organization and governance
- Cost efficiency and optimization opportunities
- Architecture dependencies and risk posture
- Operational maturity and change management

Key Findings

Environment Complexity

The Northwind Health Azure environment supports a patient-facing portal, internal APIs, notification services, and a container-based microservices platform. The architecture follows a reasonable tiered design with web, API, and data layers separated by subnets and secured via private endpoints.

Strengths identified:

- Private endpoints in place for SQL, Storage, and Key Vault
- Managed identities used for application authentication
- Application Insights configured for observability

Areas requiring attention:

- Inconsistent naming conventions across resources
- Incomplete tagging, particularly for legacy resources
- Five orphaned resources with no documented owner
- Dev environment resources running 24/7 unnecessarily

Governance Gaps

The environment lacks a formal naming standard, resulting in inconsistent resource names that complicate management and cost attribution. Several resources are missing required tags (owner, cost center, data classification), making it difficult to track accountability and compliance.

A recommended naming standard and tag governance framework is provided in the accompanying documentation.

Cost Optimization Opportunities

Current monthly Azure spend is approximately **$18,450**. The audit identified optimization opportunities totaling **$4,200–$5,800 per month** (23–31% reduction), with potential annual savings of **$50,400–$69,600**.

Category	Monthly Savings	Effort
Delete orphaned/temporary resources	$575	Low
Right-size overprovisioned compute	$1,540	Medium
Optimize storage tiers	$420	Low
Implement dev auto-shutdown	$1,100	Medium
Service tier adjustments	$1,300	Medium
Reserved Instance pricing	$600	Low

Detailed recommendations with implementation steps are provided in the Cost Analysis document.

Risk Assessment

Risk	Severity	Recommendation
Legacy VM with unknown owner	Medium	Identify owner or decommission
Overprovisioned production compute	Low	Right-size based on usage data
Service Bus Premium underutilized	Low	Evaluate Standard tier feasibility
90-day log retention (default)	Low	Reduce to 30 days, archive if needed
Key Vault redundancy	Low	Consolidate to single vault

No critical security vulnerabilities were identified during this audit. The private endpoint architecture provides strong network isolation for sensitive data services.

Recommended Next Steps

Immediate (This Week)

1. **Delete orphaned resources** — Validate and remove the identified orphaned disk, public IP, NIC, and old snapshot after confirmation they are unused.
2. **Remove temporary test resources** — Delete the load testing resource group (`rg-nwh-test-temp-eastus`) following owner confirmation.

3. **Identify legacy resource owners** — Assign ownership to `vm-nwh-legacy-001` and associated resources, or begin decommission planning.

Short-Term (This Month)

4. **Adopt naming standard** — Implement the provided naming convention for all new resources. Plan remediation for existing resources during next maintenance window.
5. **Enforce tag requirements** — Deploy Azure Policy to require mandatory tags on resource creation.
6. **Right-size production compute** — Downgrade `plan-nwh-portal-prod-001` from P2v3 to P1v3 based on utilization data showing 25% average CPU.
7. **Optimize storage tiers** — Migrate `stnwhbackups001` from Hot to Cool tier.

Medium-Term (This Quarter)

8. **Implement dev auto-shutdown** — Configure automation to stop development resources outside business hours.
9. **Evaluate Reserved Instances** — Analyze stable workloads for 1-year reserved capacity pricing on SQL databases.
10. **Review Service Bus tier** — Evaluate whether Standard tier meets requirements given current message volume (~500/day).

Summary

The Northwind Health Azure environment is fundamentally sound with appropriate security controls in place. The primary opportunities lie in governance improvements (naming, tagging) and cost optimization through right-sizing and tier adjustments.

Implementing the recommendations in this report will reduce monthly spend by 23–31%, improve operational clarity through consistent naming and tagging, and reduce risk by addressing orphaned and undocumented resources.

We recommend scheduling a follow-up session to prioritize the implementation roadmap and address any questions regarding specific recommendations.

Prepared by Azure Governance Consulting — February 2026

Azure Naming Standards and Governance

Northwind Health — Recommended Standard
Version 1.0 — February 2026

Purpose

This document establishes the official naming convention and tag governance framework for all Azure resources in the Northwind Health environment. Consistent naming and tagging improves discoverability, enables accurate cost allocation, supports automation, and reduces operational risk.

Adoption of this standard is recommended for all new resources immediately, with a phased remediation plan for existing resources.

Naming Convention

General Pattern

```
{resource-type}-{organization}-{application}-{environment}-{instance}
```

Example: `app-nwh-portal-prod-001`

Pattern Components

Component	Description	Values
`resource-type`	Abbreviated prefix for resource type	See prefix table below
`organization`	Company abbreviation	`nwh`
`application`	Application or workload name	`portal`, `api`, `notifications`
`environment`	Deployment environment	`prod`, `dev`, `test`, `stage`
`instance`	Three-digit instance number	`001`, `002`, etc.

Resource Type Prefixes

Azure Resource	Prefix	Example
Resource Group	`rg-`	`rg-nwh-prod-web-eastus`
App Service	`app-`	`app-nwh-portal-prod-001`

Azure Resource	Prefix	Example
App Service Plan	`plan-`	`plan-nwh-portal-prod-001`
Function App	`func-`	`func-nwh-notifications-prod-001`
Storage Account	`st`	`stnwhpatientdocs001`
SQL Server	`sql-`	`sql-nwh-main-prod-001`
SQL Database	`sqldb-`	`sqldb-nwh-patients-prod`
Key Vault	`kv-`	`kv-nwh-prod-001`
Service Bus	`sb-`	`sb-nwh-prod-001`
Container Registry	`acr`	`acrnwhprod001`
AKS Cluster	`aks-`	`aks-nwh-prod-001`
App Insights	`ai-`	`ai-nwh-prod-001`
Log Analytics	`log-`	`log-nwh-prod-001`
Virtual Network	`vnet-`	`vnet-nwh-prod-eastus-001`
Subnet	`snet-`	`snet-nwh-prod-web`
Private Endpoint	`pep-`	`pep-nwh-sql-prod-001`
Private DNS Zone	`pdnsz-`	`pdnsz-privatelink-sql`
Network Security Group	`nsg-`	`nsg-nwh-prod-web`
Public IP	`pip-`	`pip-nwh-portal-prod-001`
Virtual Machine	`vm-`	`vm-nwh-legacy-001`
Managed Disk	`disk-`	`disk-nwh-vm001-os`
Managed Identity	`id-`	`id-nwh-portal-prod`

Special Naming Rules

Storage Accounts (24-character limit, no hyphens)

```
st{org}{purpose}{instance}
```

Example: `stnwhpatientdocs001`

Container Registries (50-character limit, no hyphens)

```
acr{org}{env}{instance}
```

Example: `acrnwhprod001`

Resource Groups (include region)

```
rg-{org}-{env}-{function}-{region}
```

Example: `rg-nwh-prod-web-eastus`

Environment Codes

Environment	Code	Usage
Production	`prod`	Live customer-facing workloads
Staging	`stage`	Pre-production validation
Test	`test`	Integration and QA testing
Development	`dev`	Developer sandboxes

Region Codes

Region	Code
East US	`eastus`
East US 2	`eastus2`
West US 2	`westus2`
Central US	`centralus`
West Europe	`westeurope`

Required Tags

All resources must have the following tags applied at creation:

Tag	Description	Example Values
`env`	Environment	`prod`, `dev`, `test`, `stage`
`app`	Application name	`patient-portal`, `notifications`, `shared`

Tag	Description	Example Values
`owner`	Technical owner (email)	`j.martinez@northwindhealth.fake`
`costcenter`	Financial cost center	`CC-4100`, `CC-4200`
`data_classification`	Data sensitivity	`PHI`, `Confidential`, `Internal`, `Test`
`lifecycle`	Resource lifecycle status	`active`, `review`, `temporary`, `deprecated`

Optional Tags

Tag	Description
`created_by`	Creation method (terraform, manual, pipeline)
`created_date`	Resource creation date
`expiry_date`	For temporary resources
`compliance`	Compliance requirements (HIPAA, SOC2)

Data Classification Values

Classification	Description	Handling
`PHI`	Protected Health Information	Encryption required, access logging, HIPAA BAA
`Confidential`	Sensitive business data	Encryption required, need-to-know access
`Internal`	Internal business use	Standard access controls
`Public`	Publicly shareable	No special controls
`Test`	Test/synthetic data only	No production data allowed

Examples: Good vs. Bad Names

App Services

✓ Good	✗ Bad	Issue
`app-nwh-portal-prod-001`	`PatientPortal`	Missing prefix, environment, instance
`func-nwh-notifications-prod-001`	`myfunction`	Generic, unidentifiable

Storage Accounts

✓ Good	✗ Bad	Issue
`stnwhpatientdocs001`	`st-nwh-docs`	Hyphens not allowed

✓ Good	✗ Bad	Issue
`stnwhbackups001`	`storageaccount1`	Generic, no context

SQL Resources

✓ Good	✗ Bad	Issue
`sql-nwh-main-prod-001`	`SQLServer`	Generic, uppercase
`sqldb-nwh-patients-prod`	`database1`	No context

Resource Groups

✓ Good	✗ Bad	Issue
`rg-nwh-prod-web-eastus`	`ResourceGroup1`	Generic
`rg-nwh-dev-api-eastus`	`rg-portal`	Missing environment, region

Implementation Plan

Phase 1: New Resources (Immediate)

- Apply naming standard to all newly created resources
- Configure Azure Policy to enforce tag requirements
- Document exceptions in governance register

Phase 2: Existing Resources (Next Maintenance Window)

- Inventory resources requiring rename
- Plan remediation during scheduled maintenance
- Update automation scripts and documentation

Phase 3: Policy Enforcement (This Quarter)

- Enable deny policies for non-compliant names in Production
- Implement automated compliance reporting
- Schedule quarterly governance reviews

Governance Enforcement

Azure Policy Recommendations

1. **Require tags on resource creation** — Deny creation without required tags
2. **Enforce naming patterns** — Custom policy to validate naming convention
3. **Audit non-compliant resources** — Monthly compliance report

Exception Process

Deviations from this standard require:

1. Documented technical justification
2. Approval from cloud governance lead
3. Entry in exceptions register with review date

Ownership Reference

Cost Center	Department	Primary Owner
CC-4000	Infrastructure	m.thompson@northwindhealth.fake
CC-4100	Patient Portal	j.martinez@northwindhealth.fake
CC-4200	Messaging	s.chen@northwindhealth.fake
CC-4300	Data Platform	d.patel@northwindhealth.fake
CC-4500	Container Platform	l.garcia@northwindhealth.fake

This standard is based on Microsoft Cloud Adoption Framework recommendations, adapted for Northwind Health requirements.

Cost Analysis and Optimization Opportunities

Northwind Health — Azure Environment
Assessment Period: January 2026

Current State Overview

The Northwind Health Azure environment spans three subscriptions with a combined monthly spend of approximately **$18,450**. This analysis identifies optimization opportunities totaling **$4,200–$5,800 per month**, representing a 23–31% cost reduction with minimal operational risk.

Spend Distribution by Service

Service	Monthly Cost	% of Total
Azure SQL Database	$4,850	26.3%
App Service Plans	$3,920	21.2%
AKS (Kubernetes)	$2,680	14.5%
Storage Accounts	$2,150	11.7%
Service Bus	$1,890	10.2%
App Insights / Log Analytics	$1,420	7.7%
Virtual Machines (Legacy)	$890	4.8%
Other (Network, DNS, etc.)	$650	3.5%
Total	**$18,450**	**100%**

Spend Distribution by Environment

Environment	Monthly Cost	% of Total
Production	$14,200	77.0%
Development	$3,450	18.7%
Shared Services	$800	4.3%

Optimization Opportunities Summary

Priority	Opportunity	Monthly Savings	Effort	Risk
1	Delete orphaned resources	$85	Very Low	Very Low

Priority	Opportunity	Monthly Savings	Effort	Risk
2	Remove temporary load test resources	$490	Very Low	Low
3	Reduce Log Analytics retention	$280	Low	Low
4	Downgrade Production App Service Plan	$650	Low	Medium
5	Right-size AKS cluster	$890	Medium	Medium
6	Migrate backup storage to Cool tier	$420	Low	Low
7	Implement dev environment auto-shutdown	$1,100	Medium	Low
8	Downgrade Service Bus tier	$700	Medium	Medium
9	Reserved Instances for SQL	$600	Low	Low
10	Optimize App Insights sampling	$400	Medium	Low

Total Identified Savings: $4,200–$5,800/month
Potential Annual Savings: $50,400–$69,600

Detailed Recommendations

1. Delete Orphaned Resources

Savings: $85/month
Effort: Very Low (< 1 hour)
Risk: Very Low

The following resources are orphaned (not attached to any active workload):

Resource	Type	Estimated Cost
`disk-nwh-legacy-001`	Managed Disk	$45/month
`pip-nwh-legacy-001`	Public IP	$15/month
`snap-nwh-backup-20250115`	Snapshot	$25/month
`nic-nwh-legacy-001`	Network Interface	$0 (no cost)

Action: Move to quarantine resource group for 7-day observation period, then delete after confirming no dependencies.

---

2. Remove Temporary Load Test Resources

Savings: $490/month
Effort: Very Low (< 1 hour)
Risk: Low

The resource group `rg-nwh-test-temp-eastus` contains load testing resources that are no longer needed:

Resource	Type	Estimated Cost
`app-nwh-loadtest-001`	App Service	—
`plan-nwh-loadtest-001`	App Service Plan (P1v2)	$490/month

These resources are tagged with `lifecycle=temporary` and testing was completed over 30 days ago.

Action: Confirm with owner (a.wong@northwindhealth.fake) that testing is complete, then delete the entire resource group.

---

3. Reduce Log Analytics Retention

Savings: $280/month
Effort: Low (< 2 hours)
Risk: Low

Current State	Recommended
`log-nwh-prod-001` retention: 90 days	30 days operational
Cost: ~$680/month	Cost: ~$400/month

Action:

1. Reduce retention to 30 days
2. Configure export to Storage Account (Cool tier) for long-term archival if compliance requires
3. Use Archive tier for data older than 90 days

---

4. Downgrade Production App Service Plan

Savings: $650/month
Effort: Low (1–2 hours)
Risk: Medium (requires validation)

Current	Recommended
`plan-nwh-portal-prod-001`	Same plan
SKU: P2v3 (4 vCPU, 16 GB)	SKU: P1v3 (2 vCPU, 8 GB)
Cost: ~$1,500/month	Cost: ~$850/month
Utilization: 25% CPU, 30% memory	Adequate headroom at P1v3

Action:

1. Review 30-day performance metrics to confirm utilization
2. Test workload on P1v3 in staging environment
3. Schedule scale-down during low-traffic window
4. Monitor for 48 hours post-change

Alternative: Implement autoscaling rules to scale between P1v3 (baseline) and P2v3 (peak).

---

5. Right-Size AKS Cluster

Savings: $890/month
Effort: Medium (4–8 hours)
Risk: Medium

Current	Recommended
`aks-nwh-prod-001`	Same cluster
Nodes: 3 × Standard_D4s_v3	Nodes: 2 × Standard_D4s_v3
Utilization: ~15%	Enable autoscaler (min: 2, max: 4)
Cost: ~$2,680/month	Cost: ~$1,790/month

Action:

1. Review pod resource requests and limits
2. Analyze peak utilization patterns over past 30 days
3. Enable cluster autoscaler
4. Scale node pool to 2 nodes
5. Test pod scheduling and failover scenarios

---

6. Migrate Backup Storage to Cool Tier

Savings: $420/month
Effort: Low (2–4 hours)
Risk: Low

Current	Recommended
`stnwhbackups001` tier: Hot	Tier: Cool
Access pattern: Monthly or less	Cool is optimal for infrequent access
Cost: ~$650/month	Cost: ~$230/month

Action:

1. Analyze blob access patterns via Storage Analytics
2. Create lifecycle management policy to automatically tier blobs:

    - Move to Cool after 30 days
    - Move to Archive after 180 days
3. Set default tier to Cool for new uploads

---

7. Implement Dev Environment Auto-Shutdown

Savings: $1,100/month
Effort: Medium (4–6 hours)
Risk: Low

Development resources currently run 24/7 (168 hours/week) despite actual usage of ~45 hours/week during business hours.

Affected resources:

- `plan-nwh-portal-dev-001` (B2)
- `sql-nwh-main-dev-001` and databases
- Associated services

Action:

1. Document required uptime hours with development team
2. Implement Azure Automation runbook for scheduled stop/start
3. Configure SQL auto-pause (if using Serverless) or scheduled scale-down
4. Target schedule: 7 AM – 7 PM weekdays only

---

8. Evaluate Service Bus Tier

Savings: $700/month
Effort: Medium (4–8 hours)
Risk: Medium

Current	Potential
`sb-nwh-prod-001` tier: Premium	Tier: Standard
Message volume: ~500 messages/day	Standard handles this easily
Cost: ~$1,890/month	Cost: ~$1,190/month

Important consideration: Premium tier is required for private endpoints. If network isolation is mandatory, this recommendation may not apply.

Action:

1. Confirm whether private endpoint requirement is firm
2. If not, test Standard tier in non-production
3. Plan migration during maintenance window

---

9. Reserved Instances for SQL Databases

Savings: $600/month
Effort: Low (1–2 hours)
Risk: Low (financial commitment only)

Production SQL databases are stable, long-term workloads suitable for Reserved Capacity pricing.

Current	With Reserved Capacity
Pay-As-You-Go pricing	1-year Reserved Capacity
SQL spend: ~$4,850/month	~40% discount on vCore costs
	Estimated savings: ~$600/month

Action:

1. Confirm SQL databases will remain in use for 12+ months
2. Calculate exact reserved capacity requirements
3. Purchase via Azure Portal or Enterprise Agreement

---

10. Optimize App Insights Sampling

Savings: $400/month
Effort: Medium (4–6 hours)
Risk: Low

Current	Recommended
`ai-nwh-prod-001` sampling: 100%	Adaptive sampling: 25–50%
Ingestion: ~15 GB/day	Target: ~5–8 GB/day
Cost: ~$680/month	Cost: ~$280/month

Action:

1. Review telemetry data to identify high-volume, low-value events
2. Enable adaptive sampling in Application Insights SDK
3. Add filtering for common dependency calls (e.g., health checks)
4. Adjust log levels in application code

Implementation Roadmap

Week 1 (Immediate)

- [ ] Delete orphaned resources (after validation)

- [ ] Remove load test resources
- [ ] Reduce Log Analytics retention

Quick win savings: $855/month

Week 2–4 (This Month)

- [ ] Downgrade App Service Plan
- [ ] Migrate backup storage to Cool tier
- [ ] Evaluate Reserved Instance pricing

Month 1 savings: $1,925/month

Month 2–3 (This Quarter)

- [ ] Right-size AKS cluster
- [ ] Implement dev auto-shutdown
- [ ] Evaluate Service Bus tier
- [ ] Optimize App Insights sampling

Full implementation savings: $4,200–$5,800/month

Ongoing Cost Governance

To maintain cost efficiency, we recommend:

1. **Monthly cost reviews** — Review Azure Cost Management reports monthly
2. **Azure Advisor** — Review and action recommendations quarterly
3. **Budget alerts** — Configure budget alerts at 80% and 100% thresholds
4. **Tag enforcement** — Ensure cost center tags are applied for accurate allocation
5. **Reserved Instance review** — Re-evaluate RI coverage annually

---

All cost estimates are based on Azure Pay-As-You-Go pricing as of January 2026. Actual costs may vary based on usage patterns and regional pricing.

Architecture and Dependency Map

Northwind Health — Production Environment
February 2026

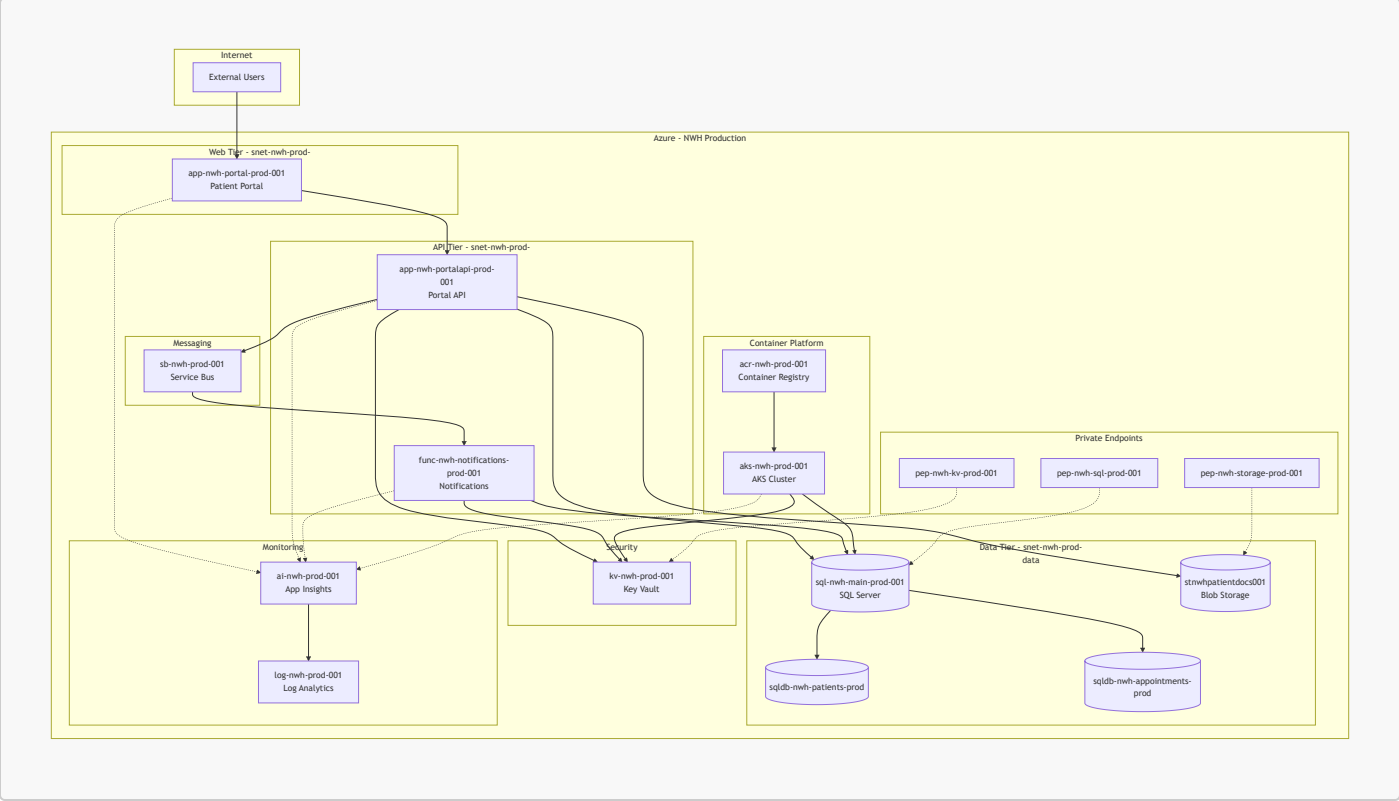
Purpose

This document provides a visual representation of the Northwind Health production Azure environment, illustrating how applications, data services, and infrastructure components interconnect. Understanding these dependencies is essential for:

- **Impact analysis** before making configuration changes
- **Troubleshooting** production issues efficiently
- **Disaster recovery** planning and testing
- **Identifying single points of failure** and resilience gaps

---

Architecture Diagram



How to Read This Diagram

Notation Guide

Symbol	Meaning
Solid arrow (→)	Direct dependency or data flow
Dashed arrow (⇢)	Monitoring/telemetry or private network connection
Rectangle	Compute resource (App Service, Function, AKS)
Cylinder	Data store (SQL Database, Storage Account)
Grouped box	Logical tier or subnet boundary

Tiers and Layers

The architecture follows a three-tier design:

1. **Web Tier** (`snet-nwh-prod-web`) — External-facing applications
2. **API Tier** (`snet-nwh-prod-api`) — Backend services and APIs
3. **Data Tier** (`snet-nwh-prod-data`) — Databases, storage, and private endpoints

---

Critical Data Flows

Patient Portal Flow (Primary Path)

```
External Users → app-nwh-portal-prod-001 → app-nwh-portalapi-prod-001 → sql-nwh-main-prod-001
```

This is the primary user-facing path. Disruption to any component in this chain results in portal unavailability.

Notification Flow (Async Processing)

```
app-nwh-portalapi-prod-001 → sb-nwh-prod-001 → func-nwh-notifications-prod-001
```

Notifications are processed asynchronously via Service Bus. If this path is disrupted, notifications are delayed but do not impact portal availability.

Document Storage Flow

```
app-nwh-portalapi-prod-001 → stnwhpatientdocs001 (via pep-nwh-storage-prod-001)
```

Patient documents are stored in blob storage, accessed via private endpoint for security.

Security Architecture

Private Endpoint Topology

All data-tier services are accessed via private endpoints, ensuring traffic stays within the Azure backbone and never traverses the public internet:

Service	Private Endpoint	Private DNS Zone
SQL Server	`pep-nwh-sql-prod-001`	`pdnsz-privatelink-sql`
Blob Storage	`pep-nwh-storage-prod-001`	`pdnsz-privatelink-blob`
Key Vault	`pep-nwh-kv-prod-001`	`pdnsz-privatelink-vault`

Secrets Management

All applications retrieve secrets from Key Vault (`kv-nwh-prod-001`) at runtime using managed identities. No connection strings or API keys are stored in application configuration.

Network Topology

Virtual Network

VNet: `vnet-nwh-prod-eastus-001`
Address Space: 10.1.0.0/16

Subnet	CIDR	Purpose
`snet-nwh-prod-web`	10.1.1.0/24	Web tier, App Service integration
`snet-nwh-prod-api`	10.1.2.0/24	API tier, Function App integration
`snet-nwh-prod-data`	10.1.3.0/24	Private endpoints, data services

Single Points of Failure Analysis

Component	Risk Level	Impact if Unavailable	Mitigation Options
SQL Server (`sql-nwh-main-prod-001`)	High	All applications fail	Geo-redundant backup, read replica
Key Vault (`kv-nwh-prod-001`)	High	All applications fail to authenticate	Soft-delete enabled, consider geo-redundancy
Service Bus (`sb-nwh-prod-001`)	Medium	Notifications delayed	Geo-DR pairing available

Component	Risk Level	Impact if Unavailable	Mitigation Options
Storage (`stnwhpatientdocs001`)	Medium	Document access unavailable	GRS replication enabled
App Service Plan	Medium	Portal unavailable	Autoscaling configured

Recommended Resilience Improvements

1. **SQL Database** — Consider adding a read replica in a secondary region for disaster recovery scenarios
2. **Key Vault** — Validate soft-delete and purge protection are enabled; consider backup to secondary region
3. **Service Bus** — Evaluate geo-DR pairing if notification SLA is critical

---

Observability

All compute resources send telemetry to Application Insights (`ai-nwh-prod-001`), which exports logs to Log Analytics (`log-nwh-prod-001`).

Monitored Resources

- `app-nwh-portal-prod-001` — Frontend application
- `app-nwh-portalapi-prod-001` — Backend API
- `func-nwh-notifications-prod-001` — Notification processor
- `aks-nwh-prod-001` — Kubernetes cluster

Monitoring Gaps Identified

The legacy VM (`vm-nwh-legacy-001`) does not appear to be sending telemetry to Application Insights. Recommend installing the Azure Monitor agent if this resource is retained.

Recommendations

1. **Document all dependencies** — This diagram should be kept current as architecture evolves
2. **Test failover scenarios** — Conduct tabletop exercises for Key Vault and SQL outage scenarios
3. **Implement health checks** — Ensure all services have appropriate health endpoints monitored
4. **Review DR strategy** — Current architecture has appropriate backups but limited active-active capability
5. **Address legacy VM** — Identify owner and either integrate monitoring or plan decommission

---

Appendix: Resource Inventory Summary

Tier	Resource Count	Key Resources
Web	2	Portal App Service, App Service Plan

Tier	Resource Count	Key Resources
API	3	API App Service, Function App, App Service Plan
Data	7	SQL Server, 2 Databases, 2 Storage Accounts, 3 Private Endpoints
Security	2	Key Vault (primary), Key Vault (secondary)
Networking	4	VNet, 3 Subnets
Monitoring	2	App Insights, Log Analytics
Containers	2	Container Registry, AKS Cluster

This diagram reflects the production environment as of the audit date. Architecture changes should trigger an update to this documentation.

Audit Methodology and Change Control Procedures

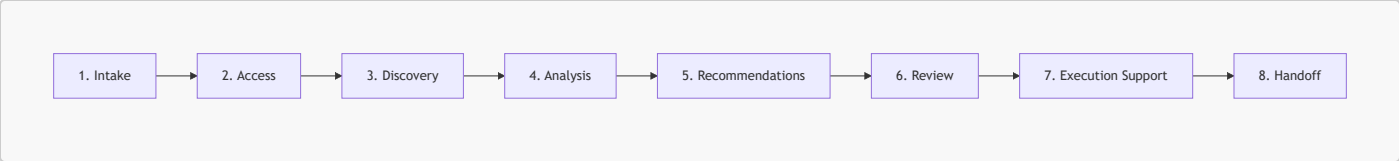
Northwind Health — Azure Environment Audit
February 2026

Part 1: How This Audit Was Performed

This section describes the methodology used to conduct the Azure environment audit, ensuring transparency about our process and the safety measures applied throughout the engagement.

Audit Phases

The audit followed an eight-phase process designed to minimize risk while delivering comprehensive insights:



Phase 1: Intake

Duration: 1–2 meetings

During intake, we gathered context about the business drivers, environment scope, and key stakeholders. Topics covered included:

- Number of subscriptions and approximate resource count
- Primary workloads and applications
- Compliance requirements (HIPAA, SOC2)
- Known issues or specific concerns
- Maintenance windows and change approval processes

Phase 2: Access Provisioning

Duration: 1–3 days

Access was provisioned with the minimum permissions necessary for discovery:

Access Level	Scope	Purpose
Reader	All in-scope subscriptions	Resource inventory, configuration review
Reader	Microsoft Entra ID	Identity and access review

Access Level	Scope	Purpose
Cost Management Reader	Cost Management	Cost data export and analysis
Log Analytics Reader	Relevant workspaces	Diagnostic and logging review

Read-Only First Policy: No write access was provisioned during the discovery phase. This ensures that the audit process cannot accidentally modify production resources.

Phase 3: Discovery

Duration: 2–5 days

Discovery involved systematic inventory and documentation of all Azure resources. Tools and methods used:

Azure Resource Graph Queries

Resource Graph queries were used to extract resource inventory, identify untagged resources, and locate orphaned resources:

```
resources
| project subscriptionId, resourceGroup, name, type, location, tags
```

Azure CLI

The Azure CLI was used to export detailed resource configurations:

```
az resource list --subscription "<subscription-name>" --output json
```

Azure Cost Management

Cost data was exported for the assessment period to identify spending patterns and optimization opportunities.

Discovery Outputs:

- Complete resource inventory (CSV/JSON)
- Resource group organization map
- Tag usage analysis
- Network topology documentation
- Cost breakdown by service and environment

Phase 4: Analysis

Duration: 3–5 days

During analysis, the discovery data was evaluated against best practices and client requirements:

Analysis Area	Focus
Naming & Organization	Consistency, discoverability, alignment with standards
Tagging	Completeness, accuracy, cost attribution capability
Cost	Top drivers, waste identification, optimization opportunities
Security	Private endpoints, Key Vault usage, RBAC configuration
Monitoring	App Insights coverage, log retention, alerting
Architecture	Dependencies, resilience, single points of failure

---

Phase 5: Recommendations

Duration: 2–3 days

Recommendations were developed with the following structure:

1. **Finding** — What was observed
2. **Risk** — Why it matters to the business
3. **Recommendation** — What action to take
4. **Effort** — Implementation complexity
5. **Priority** — Critical / High / Medium / Low

Recommendations were categorized as:

- **Quick Wins** — Low effort, immediate value
- **Short-Term** — Requires planning, achievable this month
- **Medium-Term** — Requires coordination, this quarter
- **Strategic** — Major initiative, ongoing effort

Phase 6: Client Review

Duration: 1–2 meetings

Findings and recommendations were presented to stakeholders for validation and prioritization. The review session covered:

1. Executive summary of findings
2. Key risks and concerns
3. Cost analysis and savings opportunities
4. Recommended quick wins
5. Prioritization discussion
6. Next steps and execution planning

Client feedback was incorporated into final deliverables.

Phase 7: Execution Support (if applicable)

For recommendations requiring implementation, we provided advisory support following strict change control procedures. All changes were:

- Executed by client personnel (not consultants)
- Tested in non-production first
- Scheduled during approved maintenance windows
- Documented with rollback procedures

Phase 8: Handoff

Final deliverables were compiled and handed over, including:

- Executive summary
- Detailed findings and recommendations
- Resource inventory data
- Naming standard documentation
- Dependency diagrams
- Reusable Resource Graph queries

Part 2: Change Control and Safety Procedures

This section documents the safety procedures for implementing any changes arising from audit recommendations. These procedures protect the production environment and ensure changes can be reversed if issues arise.

Core Safety Principles

1. **Read-Only First** — Discover and document before making any changes
2. **Document Before Doing** — Every change is documented with expected outcome and rollback procedure before execution
3. **Non-Production First** — Test all changes in development/test before production
4. **Staged Deletion** — Never delete resources immediately; use quarantine process
5. **Client Ownership** — Client personnel execute changes; consultants advise

---

Change Categories

Category	Risk Level	Examples	Approval	Timing
1	Low	Tags, diagnostic settings, log retention	Technical lead verbal	Anytime
2	Medium	Scale down (non-prod), storage tier change, delete orphaned (after quarantine)	Technical lead written	Business hours
3	Higher	Scale down (prod), network config, Service Bus tier	Change board	Maintenance window
4	High	Private endpoints, Key Vault policies, SQL tier (prod), NSG rules	Change board + Security	Maintenance window

Quarantine Process for Resource Deletion

Resources identified for deletion must go through a quarantine period before permanent removal:

Step 1: Create Quarantine Resource Group

```
az group create \
  --name "rg-nwh-quarantine-eastus" \
  --location "eastus" \
  --tags "purpose=quarantine" "created_date=2026-02-01"
```

Step 2: Move Resources to Quarantine

```
az resource move \
  --destination-group "rg-nwh-quarantine-eastus" \
  --ids "<resource-id>"
```

Step 3: Observation Period

Resource Type	Minimum Wait
Orphaned disk/snapshot	7 days
Unused public IP	7 days
Test/temporary resources	3 days
Any production-tagged resource	14 days

Step 4: Validation Before Deletion

- [ ] No error logs referencing resource

- [ ] No failed deployments looking for resource
- [ ] Owner confirms no longer needed
- [ ] Observation period complete

Step 5: Delete After Validation

```
az group delete --name "rg-nwh-quarantine-eastus" --yes
```

Backup Requirements Before Production Changes

Resource Type	Backup Method	Verification
SQL Database	Point-in-time restore	Verify backup exists
Storage Account	Soft delete enabled	Verify retention period
Key Vault	Soft delete + purge protection	Verify settings
VM	Snapshot OS + data disks	Verify snapshot completed
App Service	Deployment slots	Verify slot can swap
AKS	etcd backup / namespace export	Verify backup accessible

Backup Checklist:

- [ ] Backup taken < 24 hours before change
- [ ] Backup verified (can we restore?)
- [ ] Retention sufficient for rollback window
- [ ] Restore procedure documented

Rollback Procedures

App Service Plan Scale Rollback:

```
az appservice plan update \
  --name "plan-nwh-portal-prod-001" \
  --resource-group "rg-nwh-prod-web-eastus" \
  --sku P2v3  # Original SKU
```

SQL Database Tier Rollback:

```
az sql db update \
  --name "sqldb-nwh-patients-prod" \
```

```
  --server "sql-nwh-main-prod-001" \
  --resource-group "rg-nwh-prod-data-eastus" \
  --service-objective S3  # Original tier
```

Storage Account Tier Rollback:

- Access tier changes may take up to 24 hours to complete
- For urgent rollback, copy data to new account with correct tier

Key Vault Secret Recovery:

```
az keyvault secret recover \
  --vault-name "kv-nwh-prod-001" \
  --name "<secret-name>"
```

Pre-Change Checklist

Use before executing any change:

- [ ] Change documented and approved
- [ ] Backups verified
- [ ] Rollback procedure defined and tested
- [ ] Team notified
- [ ] Monitoring in place
- [ ] Within approved maintenance window (if required)
- [ ] Tested in non-production (if applicable)

Post-Change Checklist

Use after executing any change:

- [ ] Change completed successfully
- [ ] Functionality verified
- [ ] No errors in logs
- [ ] Performance within expected range
- [ ] Change log updated
- [ ] Stakeholders notified

Change Log Template

Date	Resource	Change	Performed By	Approved By	Rollback Available

---

Emergency Procedures

If Something Goes Wrong:

1. **Stop** — Do not make additional changes
2. **Assess** — What changed? What's the impact?
3. **Communicate** — Notify stakeholders immediately
4. **Rollback** — Execute documented rollback procedure
5. **Verify** — Confirm system is restored
6. **Document** — Record incident details for post-mortem

Part 3: Engagement Deliverables Checklist

The following items were delivered as part of this engagement:

Item	Status
Executive Summary	✓ Delivered
Naming Standards and Governance	✓ Delivered
Cost Analysis and Quick Wins	✓ Delivered
Architecture Dependency Map	✓ Delivered
Audit Methodology and SOP	✓ Delivered
Resource Inventory (CSV/JSON)	✓ Available upon request
Tag and Ownership Matrix	✓ Available upon request

Following these procedures ensures that changes are implemented safely, with appropriate oversight and the ability to recover if issues arise.