High-Level Design - openshift-voting-app Azure Migration

**Document Version**: 1.0

**Date**: 2025-07-17

**Classification**: Confidential

**Status**: Draft

---

Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Changes** |
| 1.0 | 2025-07-17 | Migration Team | Initial version |

---

Table of Contents

1. Introduction
2. Scope
3. Current State Summary
4. Target Architecture
5. Service Mapping
6. Migration Strategy
7. Technical Architecture
8. Security Design
9. Networking Design
10. Data Architecture
11. Monitoring and Observability
12. Migration Phases
13. Risk Assessment and Mitigation
14. Cost Analysis
15. Success Criteria
16. Appendices

1. Introduction

1.1 Purpose

Executive Summary: Migration of Openshift Voting App to Azure

This document outlines the High-Level Design for migrating the Openshift Voting App (currently hosted on Github at https://github.com/end-of-game/openshift-voting-app) to the Azure cloud platform. The primary objective of this migration is to modernize the application infrastructure, leverage Azure's managed services, and improve scalability and reliability while maintaining the application's core functionality. This migration will involve shifting from the existing microservices architecture utilizing Redis and a SQL database to a comparable Azure environment using Azure App Service for compute, Azure Database for PostgreSQL for persistent data storage, and Azure Cache for Redis to maintain performant data caching.

The proposed architecture leverages Azure's Platform-as-a-Service (PaaS) offerings to minimize operational overhead. The application's components will be re-deployed as web applications within Azure App Service, providing a fully managed hosting environment. The existing SQL database will be migrated to Azure Database for PostgreSQL, a robust and scalable managed database service. Azure Cache for Redis will replace the existing Redis implementation, ensuring consistent caching performance and simplifying management. This approach simplifies the deployment process, automates scaling, and reduces the need for manual infrastructure management.

This migration is expected to yield several key benefits including increased availability and resilience through Azure's global infrastructure, improved scalability to handle fluctuating user demand, and reduced operational overhead due to the utilization of managed services. Furthermore, leveraging Azure's integrated monitoring and logging tools will provide enhanced insights into application performance and health. We anticipate a low complexity migration effort with an estimated timeline of [Insert Estimated Timeline Here - e.g., 4-6 weeks] for completion, contingent upon thorough testing and validation. This migration represents a strategic move towards a more efficient and scalable cloud infrastructure.

1.2 Document Scope

This High-Level Design document provides:

* Target state architecture on Microsoft Azure
* Service mapping from current to target state
* Migration strategy and phasing
* Technical design decisions
* Risk mitigation approaches
* Cost estimates and optimization strategies

1.3 Audience

This document is intended for:

* Technical Architecture Team
* Development Teams
* Infrastructure Teams
* Project Management
* Security and Compliance Teams
* Executive Stakeholders

1.4 Related Documents

* AS-IS State Analysis Document
* Azure Well-Architected Framework
* Organization's Cloud Adoption Framework
* Security and Compliance Requirements

2. Scope

2.1 In Scope

Applications and Services

* voting-app

Data Stores

* Redis Cache
* MongoDB
* SQL Database

Infrastructure Components

* Container orchestration
* Load balancing
* Networking
* Security controls
* Monitoring

2.2 Out of Scope

The following items are explicitly out of scope for this migration:

* Third-party SaaS migrations
* End-user training
* Legacy system decommissioning
* Data archival beyond 2 years

2.3 Assumptions

* Current application architecture is stable
* No major feature changes during migration
* Azure subscription and landing zone ready
* Team has basic Azure knowledge

2.4 Constraints

* Migration must be completed within 3 months
* Minimal downtime allowed for critical services
* Budget constraints as per approved proposal
* Compliance requirements must be maintained

3. Current State Summary

3.1 Architecture Overview

The current application follows a **microservices architecture** with the following characteristics:

* \*\*Total Services\*\*: 1
* \*\*Critical Services\*\*: 0
* \*\*Architecture Pattern\*\*: Event-driven microservices
* \*\*Container Platform\*\*: Docker
* \*\*Orchestration\*\*: Kubernetes/OpenShift

3.2 Technology Stack

|  |  |  |
| --- | --- | --- |
| **Component** | **Technology** | **Purpose** |
| voting-app | Unknown | Service |

3.3 Current Challenges

Based on the AS-IS analysis, the following challenges have been identified:

1. \*\*Scalability\*\*: Manual scaling of services
2. \*\*Operations\*\*: High operational overhead for infrastructure management
3. \*\*Security\*\*: Limited security controls and monitoring
4. \*\*Cost\*\*: Unpredictable infrastructure costs
5. \*\*Agility\*\*: Slow deployment and update cycles

4. Target Architecture

4.1 Architecture Principles

The target architecture on Azure follows these principles:

1. \*\*Cloud-Native\*\*: Leverage PaaS services where possible
2. \*\*Microservices\*\*: Maintain service independence and scalability
3. \*\*Security by Design\*\*: Implement defense in depth
4. \*\*Cost Optimized\*\*: Right-size resources with auto-scaling
5. \*\*Highly Available\*\*: Multi-zone deployment for critical services
6. \*\*Observable\*\*: Comprehensive monitoring and logging

4.2 High-Level Architecture Diagram

┌─────────────────────────────────────────────────────────────────┐

│ Azure Subscription │

├─────────────────────────────────────────────────────────────────┤

│ │

│ ┌──────────────┐ ┌─────────────────┐ ┌──────────────┐ │

│ │ Azure Front │────▶│ App Gateway WAF │───▶│ AKS │ │

│ │ Door │ └─────────────────┘ │ │ │

│ └──────────────┘ │ ┌────────┐ │ │

│ │ │ Pods │ │ │

│ │ └────────┘ │ │

│ └───────┬──────┘ │

│ │ │

│ ┌──────────────────────────────┬────────────────────┴──────┐ │

│ │ │ │ │

│ ▼ ▼ ▼ │

│ ┌─────────────┐ ┌──────────────────────┐ ┌──────────────┐ │

│ │Azure Cache │ │ Azure Database for │ │ Azure Key │ │

│ │for Redis │ │ PostgreSQL │ │ Vault │ │

│ └─────────────┘ └──────────────────────┘ └──────────────┘ │

│ │

│ ┌─────────────────────────────────────────────────────────────┐│

│ │ Azure Monitor / App Insights ││

│ └─────────────────────────────────────────────────────────────┘│

└─────────────────────────────────────────────────────────────────┘

4.3 Compute Services

* \*\*voting-app\*\*: Azure App Service

4.4 Data Services

* \*\*redis\*\*: Azure Cache for Redis
* \*\*database\*\*: Azure Database for PostgreSQL

5. Service Mapping

5.1 Application Services

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Current Component** | **Current Tech** | **Target Azure Service** | **Tier** | **Complexity** | **Est. Cost/Month** |
| voting-app | javascript | Azure App Service | B1 (Basic) | Low | $44 - $82 |
| redis | Redis | Azure Cache for Redis | Standard C1 | Low | $50 - $100 |

5.2 Data Services

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Current Component** | **Current Tech** | **Target Azure Service** | **Tier** | **Complexity** | **Est. Cost/Month** |
| redis | Redis | Azure Cache for Redis | Standard C1 | Low | $50 - $100 |
| database | SQL Database | Azure Database for PostgreSQL | General Purpose | Medium | $100 - $200 |

5.3 Service Selection Justification

voting-app

**Selection**: Azure App Service

**Justification**: Azure App Service selected because: Already containerized, enabling cloud-native deployment; Web application with user-facing interface; RESTful API service

redis

**Selection**: Azure Cache for Redis

**Justification**: Direct Redis compatibility for caching and message queue

database

**Selection**: Azure Database for PostgreSQL

**Justification**: SQL database migration with managed service benefits

6. Migration Strategy

6.1 Migration Approach

The migration will follow a **phased approach** to minimize risk and ensure business continuity:

1. \*\*Lift and Shift with Optimization\*\*: Containerized services will be migrated with minimal changes
2. \*\*Gradual Modernization\*\*: Post-migration optimization and cloud-native feature adoption
3. \*\*Blue-Green Deployment\*\*: Zero-downtime migration for critical services

6.2 Migration Principles

* \*\*Risk Mitigation\*\*: Test thoroughly in non-production before production migration
* \*\*Incremental\*\*: Migrate services based on criticality and dependencies
* \*\*Reversible\*\*: Maintain rollback capability throughout the migration
* \*\*Observable\*\*: Comprehensive monitoring during and after migration
* \*\*Automated\*\*: Use Infrastructure as Code and CI/CD pipelines

6.3 Migration Tools

* \*\*Azure Migrate\*\*: Assessment and migration planning
* \*\*Azure Database Migration Service\*\*: Database migration with minimal downtime
* \*\*Azure DevOps\*\*: CI/CD pipeline for automated deployments
* \*\*Terraform\*\*: Infrastructure as Code for Azure resources
* \*\*Azure Container Registry\*\*: Container image management

7. Technical Architecture

7.1 Container Platform

**Decision**: Azure Kubernetes Service (AKS) for microservices requiring orchestration

**AKS Configuration**:

* \*\*Node Pools\*\*: System (2 nodes) + User (3-10 nodes with auto-scaling)
* \*\*VM Size\*\*: Standard\_D4s\_v3 for production workloads
* \*\*Networking\*\*: Azure CNI with private cluster
* \*\*Ingress\*\*: NGINX Ingress Controller
* \*\*Service Mesh\*\*: Optional Istio/Linkerd for advanced scenarios

7.2 Application Services

**Decision**: Azure App Service for stateless web applications

**App Service Configuration**:

* \*\*Plan\*\*: P1v3 for production, B1 for dev/test
* \*\*Deployment Slots\*\*: Blue-Green deployment support
* \*\*Auto-scale\*\*: CPU/Memory based scaling rules

7.3 Data Platform

**Decision**: Azure Database Migration Service for minimal downtime

**Database Configuration**:

* \*\*PostgreSQL\*\*: General Purpose, Gen5, 4 vCores
* \*\*High Availability\*\*: Zone redundant deployment
* \*\*Backup\*\*: Automated daily backups with 35-day retention
* \*\*Redis Cache\*\*: Standard C1 (1GB) with persistence

7.4 DevOps and CI/CD

**Decision**: Azure DevOps Pipelines with GitOps approach

**Pipeline Architecture**:

trigger:

- main

- develop

stages:

- stage: Build

jobs:

- job: BuildContainers

- job: RunTests

- job: SecurityScan

- stage: DeployDev

jobs:

- deployment: DeployToAKS

- stage: DeployProd

condition: and(succeeded(), eq(variables['Build.SourceBranch'], 'refs/heads/main'))

jobs:

- deployment: DeployToProduction

8. Security Design

8.1 Security Architecture

The security design follows **Defense in Depth** principles:

┌─────────────────────────────────────────────────┐

│ Azure Security Center │

├─────────────────────────────────────────────────┤

│ │

│ ┌───────────┐ ┌──────────┐ ┌──────────┐ │

│ │ WAF │───▶│ NSGs │──▶│ Firewall │ │

│ └───────────┘ └──────────┘ └──────────┘ │

│ │

│ ┌─────────────────────────────────────────┐ │

│ │ Identity & Access (AAD) │ │

│ └─────────────────────────────────────────┘ │

│ │

│ ┌──────────┐ ┌──────────────┐ │

│ │Key Vault │ │Managed Identity│ │

│ └──────────┘ └──────────────┘ │

└─────────────────────────────────────────────────┘

8.2 Identity and Access Management

* \*\*Identity Provider\*\*: Azure Active Directory
* \*\*Service Authentication\*\*: Managed Identity (no passwords)
* \*\*User Authentication\*\*: Azure AD with MFA
* \*\*RBAC\*\*: Role-based access control at all levels

8.3 Secrets Management

* \*\*Secret Store\*\*: Azure Key Vault
* \*\*Certificate Management\*\*: Azure Key Vault
* \*\*Key Rotation\*\*: Automated rotation policies
* \*\*Access Control\*\*: Managed Identity for applications

8.4 Network Security

* \*\*WAF\*\*: Azure Application Gateway WAF
* \*\*DDoS Protection\*\*: Azure DDoS Protection Standard
* \*\*Network Segmentation\*\*: Network Security Groups
* \*\*Private Endpoints\*\*: For all PaaS services

8.5 Data Protection

* \*\*Encryption at Rest\*\*: Azure Storage Encryption
* \*\*Encryption in Transit\*\*: TLS 1.2+
* \*\*Data Classification\*\*: Automated with Azure Purview
* \*\*Backup Encryption\*\*: Encrypted backups with customer-managed keys

9. Networking Design

9.1 Network Topology

**Hub-Spoke Architecture** with centralized services:

Internet

│

┌──────┴──────┐

│ Front Door │

└──────┬──────┘

│

┌──────┴──────┐

│ App GW │

└──────┬──────┘

│

┌───────────────────┴───────────────────┐

│ Hub VNet (10.0.0.0/16) │

│ ┌─────────────┐ ┌──────────────┐ │

│ │ Firewall │ │ Bastion Host │ │

│ └─────────────┘ └──────────────┘ │

└────────────────┬──────────────────────┘

│ Peering

┌───────────┴───────────┐

│ │

┌────┴─────┐ ┌─────┴────┐

│ Spoke 1 │ │ Spoke 2 │

│ AKS │ │ Data │

└──────────┘ └──────────┘

9.2 Virtual Network Design

**VNet**: app-vnet

**Address Space**: 10.0.0.0/16

|  |  |  |
| --- | --- | --- |
| **Subnet** | **Address Range** | **Purpose** |
| aks-subnet | 10.0.1.0/24 | Aks |
| app-subnet | 10.0.2.0/24 | App |
| data-subnet | 10.0.3.0/24 | Data |
| agw-subnet | 10.0.4.0/24 | Agw |

9.3 Load Balancing

* \*\*Global Load Balancer\*\*: Azure Front Door
* \*\*Regional Load Balancer\*\*: Azure Application Gateway
* \*\*Internal Load Balancer\*\*: AKS Internal Load Balancer

9.4 DNS Strategy

* \*\*Public DNS\*\*: Azure DNS
* \*\*Private DNS\*\*: Azure Private DNS Zones
* \*\*Split-Brain DNS\*\*: Internal and external resolution

9.5 Connectivity

* \*\*ExpressRoute/VPN\*\*: For hybrid connectivity (if required)
* \*\*Private Endpoints\*\*: All PaaS services accessed privately
* \*\*Service Endpoints\*\*: For Azure Storage and Key Vault
* \*\*NAT Gateway\*\*: For outbound internet connectivity

10. Data Architecture

10.1 Data Services Overview

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Store** | **Purpose** | **Azure Service** | **High Availability** | **Backup Strategy** |
| redis | Cache/Queue | Azure Cache for Redis | Standard Replication | Persistence enabled |
| database | Transactional Data | Azure Database for PostgreSQL | Zone Redundant | Daily, 35-day retention |

10.2 Data Migration Strategy

1. \*\*Assessment Phase\*\*

- Current data volume assessment

- Schema compatibility check

- Performance baseline

1. \*\*Migration Approach\*\*

- Azure Database Migration Service for PostgreSQL

- Redis replication for cache migration

- Minimal downtime approach

1. \*\*Validation\*\*

- Data integrity checks

- Performance validation

- Application testing

10.3 Data Security

* \*\*Encryption at Rest\*\*: Transparent Data Encryption (TDE)
* \*\*Encryption in Transit\*\*: SSL/TLS enforced
* \*\*Access Control\*\*: Azure AD authentication where supported
* \*\*Audit Logging\*\*: Enabled for all data operations

10.4 Backup and Recovery

* \*\*RPO\*\*: 1 hour (point-in-time restore)
* \*\*RTO\*\*: < 4 hours
* \*\*Backup Retention\*\*: 35 days
* \*\*Geo-redundancy\*\*: Enabled for critical databases
* \*\*DR Testing\*\*: Quarterly DR drills

11. Monitoring and Observability

11.1 Monitoring Architecture

Applications/Services

│

├──── Metrics ────▶ Azure Monitor

│

├──── Logs ──────▶ Azure Log Analytics

│

└──── Traces ────▶ Azure Application Insights

│

▼

Azure Dashboard

│

▼

Azure Monitor Alerts

11.2 Application Performance Monitoring

* \*\*APM Solution\*\*: Azure Application Insights
* \*\*Instrumentation\*\*: Auto-instrumentation + custom metrics
* \*\*Distributed Tracing\*\*: End-to-end transaction tracking
* \*\*Performance Baselines\*\*: Established during migration

11.3 Infrastructure Monitoring

* \*\*Metrics Collection\*\*: Azure Monitor
* \*\*Log Aggregation\*\*: Azure Log Analytics
* \*\*Resource Metrics\*\*: CPU, Memory, Disk, Network
* \*\*Custom Metrics\*\*: Business KPIs

11.4 Alerting Strategy

|  |  |  |  |
| --- | --- | --- | --- |
| **Alert Category** | **Examples** | **Severity** | **Action** |
| Availability | Service down, Health check failed | Critical | Immediate response |
| Performance | Response time > 1s, CPU > 80% | High | Investigation required |
| Capacity | Storage > 80%, Memory > 85% | Medium | Planning required |
| Security | Failed auth attempts, Suspicious activity | High | Security team notified |

11.5 Dashboards

* \*\*Executive Dashboard\*\*: Business KPIs, SLA compliance
* \*\*Operations Dashboard\*\*: Service health, performance metrics
* \*\*Security Dashboard\*\*: Security events, compliance status
* \*\*Cost Dashboard\*\*: Resource utilization, spending trends

12. Migration Phases

12.1 Migration Timeline Overview

Week 1 2 3 4 5 6 7 8 9 10 11 12

│--Phase 1--│--Phase 2--│--P3-│-P4-│P5│

Foundation Data Svcs NCrit Crit Cut

12.2 Detailed Phase Breakdown

Phase 1: Foundation and Infrastructure

**Duration**: 2-3 weeks

**Dependencies**: None

**Components**:

**Key Activities**:

1. Set up Azure subscription and resource groups
2. Configure networking (VNet, Subnets, NSGs)
3. Set up Azure Key Vault for secrets
4. Configure Azure Container Registry
5. Set up monitoring infrastructure
6. Create CI/CD pipelines

**Risks**:

* Azure subscription limits
* Network configuration complexity

**Success Criteria**:

* All infrastructure components deployed
* Connectivity established
* Security baseline configured

---

Phase 2: Data Services Migration

**Duration**: 2-3 weeks

**Dependencies**: Phase 1

**Components**:

* redis
* database

**Key Activities**:

1. Deploy redis, database
2. Migrate data with minimal downtime
3. Set up database replication
4. Test data integrity
5. Configure backup policies

**Risks**:

* Data migration failures
* Performance degradation
* Connection string updates

**Success Criteria**:

* All data migrated successfully
* Performance benchmarks met
* Zero data loss verified

---

Phase 3: Non-Critical Services Migration

**Duration**: 1-2 weeks

**Dependencies**: Phase 2

**Components**:

* voting-app

**Key Activities**:

1. Containerize and deploy voting-app
2. Update service configurations
3. Test service functionality
4. Update DNS entries

**Risks**:

* Service compatibility issues

**Success Criteria**:

* All non-critical services operational

---

Phase 4: Cutover and Optimization

**Duration**: 1-2 weeks

**Dependencies**: Phase 3

**Components**:

* all

**Key Activities**:

1. Complete traffic cutover
2. Decommission old infrastructure
3. Performance optimization
4. Cost optimization
5. Documentation updates
6. Knowledge transfer

**Risks**:

* Rollback complexity

**Success Criteria**:

* 100% traffic on Azure
* Old infrastructure decommissioned
* Cost targets achieved

---

13. Risk Assessment and Mitigation

13.1 Risk Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Probability** | **Impact** | **Severity** | **Mitigation Strategy** |
| Data Loss During Migration | Low | High | High | Implement comprehensive backup strategy before migration, maintain source systems until validation |
| Service Disruption | Medium | High | High | Use blue-green deployment with gradual traffic shifting |
| Performance Degradation | Medium | Medium | Medium | Conduct thorough performance testing, implement auto-scaling |
| Cost Overrun | Medium | Medium | Medium | Implement cost monitoring and alerts from day 1 |
| Security Vulnerabilities | Low | High | High | Security assessment and remediation before migration |

13.2 Detailed Mitigation Strategies

Data Loss

**Strategy**: Implement comprehensive backup strategy before migration, maintain source systems until validation

Service Disruption

**Strategy**: Use blue-green deployment with gradual traffic shifting

Performance Degradation

**Strategy**: Conduct thorough performance testing, implement auto-scaling

Security Vulnerabilities

**Strategy**: Security assessment and remediation before migration

Cost Overrun

**Strategy**: Implement cost monitoring and alerts from day 1

Skill Gaps

**Strategy**: Provide Azure training, engage Azure FastTrack program

Integration Failures

**Strategy**: Comprehensive integration testing in staging environment

Compliance Issues

**Strategy**: Ensure Azure services meet compliance requirements

13.3 Contingency Planning

1. \*\*Rollback Strategy\*\*

- Maintain source systems until validation complete

- Database replication for quick rollback

- DNS-based traffic switching

1. \*\*Business Continuity\*\*

- Parallel run capability

- Gradual migration approach

- Communication plan for stakeholders

1. \*\*Technical Contingencies\*\*

- Alternative Azure services identified

- Hybrid operation mode possible

- Performance tuning playbooks ready

14. Cost Analysis

14.1 Estimated Monthly Costs

|  |  |  |
| --- | --- | --- |
| **Category** | **Estimated Cost/Month** | **Percentage** |
| Compute | $63 | 13.5% |
| Data | $225 | 48.1% |
| Networking | $100 | 21.4% |
| Monitoring | $50 | 10.7% |
| Backup | $30 | 6.4% |
| \*\*Total\*\* | \*\*$468\*\* | \*\*100%\*\* |

14.2 Annual Projection

* \*\*Monthly Cost\*\*: $468
* \*\*Annual Cost\*\*: $5,616
* \*\*3-Year TCO\*\*: $16,848

14.3 Cost Optimization Opportunities

* Use Reserved Instances for 30-60% savings
* Implement auto-scaling to reduce costs during low usage
* Use Azure Hybrid Benefit if applicable
* Regular cost reviews and right-sizing

14.4 Cost Comparison

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Current (Estimated)** | **Azure** | **Savings** |
| Infrastructure | $X,XXX | $468 | TBD |
| Operations | $X,XXX | Reduced | ~30-40% |
| Licensing | $X,XXX | Included | Varies |

14.5 Return on Investment

**Quantifiable Benefits**:

* Reduced operational overhead
* Improved scalability
* Enhanced security posture
* Faster time to market

**Expected ROI Timeline**: 18-24 months based on operational savings and efficiency gains

15. Success Criteria

15.1 Technical Success Criteria

* [ ] All services successfully migrated to Azure
* [ ] Zero data loss during migration
* [ ] Performance SLAs met or exceeded
* [ ] Security baseline implemented
* [ ] Automated CI/CD pipelines operational
* [ ] Monitoring and alerting configured
* [ ] Disaster recovery tested successfully

15.2 Operational Success Criteria

* [ ] Operations team trained on Azure
* [ ] Runbooks and documentation updated
* [ ] Support processes established
* [ ] Incident response procedures tested
* [ ] Cost tracking and optimization in place

15.3 Business Success Criteria

* [ ] Minimal business disruption (< 4 hours total)
* [ ] User experience maintained or improved
* [ ] Scalability objectives achieved
* [ ] Cost targets met (± 10%)
* [ ] Compliance requirements satisfied

15.4 Migration Acceptance Criteria

**Phase Gate Reviews**: Each phase must meet the following criteria before proceeding:

1. \*\*Functional Testing\*\*: 100% pass rate
2. \*\*Performance Testing\*\*: Meet or exceed baseline
3. \*\*Security Validation\*\*: No critical vulnerabilities
4. \*\*Documentation\*\*: Updated and reviewed
5. \*\*Stakeholder Approval\*\*: Sign-off obtained

16. Appendices

Appendix A: Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| AKS | Azure Kubernetes Service |
| PaaS | Platform as a Service |
| IaC | Infrastructure as Code |
| NSG | Network Security Group |
| WAF | Web Application Firewall |
| RPO | Recovery Point Objective |
| RTO | Recovery Time Objective |

Appendix B: Reference Architecture

* [Azure Well-Architected Framework](https://docs.microsoft.com/azure/architecture/framework/)
* [AKS Best Practices](https://docs.microsoft.com/azure/aks/best-practices)
* [Azure Security Best Practices](https://docs.microsoft.com/azure/security/fundamentals/best-practices-and-patterns)

Appendix C: Configuration Templates

Sample Terraform configuration for AKS:

resource "azurerm\_kubernetes\_cluster" "main" {

name = "${var.prefix}-aks"

location = azurerm\_resource\_group.main.location

resource\_group\_name = azurerm\_resource\_group.main.name

dns\_prefix = var.prefix

default\_node\_pool {

name = "default"

node\_count = 3

vm\_size = "Standard\_D4s\_v3"

enable\_auto\_scaling = true

min\_count = 2

max\_count = 10

}

identity {

type = "SystemAssigned"

}

network\_profile {

network\_plugin = "azure"

network\_policy = "calico"

}

}

Appendix D: Contact Information

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Name** | **Email** | **Phone** |
| Project Manager | TBD | TBD | TBD |
| Technical Lead | TBD | TBD | TBD |
| Azure Architect | TBD | TBD | TBD |
| Security Lead | TBD | TBD | TBD |