

HX Infrastructure Visual Documentation



Comprehensive Visual Framework

This document contains all the visual diagrams for the HX Infrastructure project, providing comprehensive visual documentation for architecture, workflows, and processes.

Infrastructure Topology Diagram

```

graph TB
  subgraph "Internet"
    INTERNET[Internet Traffic<br/>🌐 External Users]
  end

  subgraph "DMZ - Load Balancer Tier"
    LB1[Load Balancer 1<br/>📍 10.0.1.10<br/>🔧 nginx + keepalived<br/>🔒 SSL Termination<br/>⚡ Priority: 110 MASTER]
    LB2[Load Balancer 2<br/>📍 10.0.1.11<br/>🔧 nginx + keepalived<br/>🔒 SSL Termination<br/>⚡ Priority: 100 BACKUP]
    VIP[Virtual IP<br/>📍 10.0.1.100<br/>🔄 Floating IP<br/>🎯 Active Endpoint]
  end

  subgraph "DMZ - Web Tier"
    WEB1[Web Server 1<br/>📍 10.0.2.10<br/>🔧 nginx + static content<br/>🚀 CDN Integration<br/>📊 Auto-scaling Ready]
    WEB2[Web Server 2<br/>📍 10.0.2.11<br/>🔧 nginx + static content<br/>🚀 CDN Integration<br/>📊 Auto-scaling Ready]
    WEB3[Web Server 3<br/>📍 10.0.2.12<br/>🔧 nginx + static content<br/>🚀 CDN Integration<br/>📊 Auto-scaling Ready]
  end

  subgraph "Private - Application Tier"
    APP1[App Server 1<br/>📍 10.0.3.10<br/>🔧 Application Runtime<br/>⚡ Auto-scaling<br/>💾 4GB Memory]
    APP2[App Server 2<br/>📍 10.0.3.11<br/>🔧 Application Runtime<br/>⚡ Auto-scaling<br/>💾 4GB Memory]
    APP3[App Server 3<br/>📍 10.0.3.12<br/>🔧 Application Runtime<br/>⚡ Auto-scaling<br/>💾 4GB Memory]
  end

  subgraph "Private - Database Tier"
    DB1[Database Master<br/>📍 10.0.4.10<br/>🔧 PostgreSQL 15<br/>💾 Primary Read/Write<br/>🔄 Streaming Replication<br/>📊 16GB Memory]
    DB2[Database Replica 1<br/>📍 10.0.4.11<br/>🔧 PostgreSQL 15<br/>📖 Read Replica<br/>🔄 Async Replication<br/>📊 16GB Memory]
    DB3[Database Replica 2<br/>📍 10.0.4.12<br/>🔧 PostgreSQL 15<br/>📖 Read Replica<br/>🔄 Async Replication<br/>📊 16GB Memory]
  end

  subgraph "Private - Cache Tier"
    CACHE1[Redis Master<br/>📍 10.0.5.10<br/>🔧 Redis 7.x<br/>⚡ Session Store<br/>💾 6GB Memory<br/>🔄 Master Role]
    CACHE2[Redis Replica<br/>📍 10.0.5.11<br/>🔧 Redis 7.x<br/>📖 Read Replica<br/>💾 6GB Memory<br/>🔄 Replica Role]
  end

  subgraph "Private - Monitoring & Management"
    MON1[Monitoring Server<br/>📍 10.0.6.10<br/>🔧 Prometheus + Grafana<br/>📊 Metrics & Dashboards<br/>🔔 Alertmanager<br/>📈 90-day Retention]
    LOG1[Log Server<br/>📍 10.0.6.11<br/>🔧 ELK Stack<br/>📝 Centralized Logging<br/>🔍 Elasticsearch<br/>📊 Kibana Dashboards]
  end

  %% Traffic Flow - Primary Path
  INTERNET --> VIP
  VIP --> LB1
  VIP --> LB2

  LB1 --> WEB1
  LB1 --> WEB2

```

```

LB1 --> WEB3
LB2 --> WEB1
LB2 --> WEB2
LB2 --> WEB3

WEB1 --> APP1
WEB2 --> APP2
WEB3 --> APP3

APP1 --> DB1
APP2 --> DB1
APP3 --> DB1

%% Database Replication
DB1 --> DB2
DB1 --> DB3

%% Cache Access
APP1 --> CACHE1
APP2 --> CACHE1
APP3 --> CACHE1

%% Cache Replication
CACHE1 --> CACHE2

%% Monitoring Connections (dotted lines)
MON1 -. -> LB1
MON1 -. -> LB2
MON1 -. -> WEB1
MON1 -. -> WEB2
MON1 -. -> WEB3
MON1 -. -> APP1
MON1 -. -> APP2
MON1 -. -> APP3
MON1 -. -> DB1
MON1 -. -> DB2
MON1 -. -> DB3
MON1 -. -> CACHE1
MON1 -. -> CACHE2

%% Logging Connections (dotted lines)
LOG1 -. -> LB1
LOG1 -. -> LB2
LOG1 -. -> WEB1
LOG1 -. -> WEB2
LOG1 -. -> WEB3
LOG1 -. -> APP1
LOG1 -. -> APP2
LOG1 -. -> APP3
LOG1 -. -> DB1
LOG1 -. -> DB2
LOG1 -. -> DB3
LOG1 -. -> CACHE1
LOG1 -. -> CACHE2

%% Styling
classDef internetClass fill:#e1f5fe,stroke:#01579b,stroke-width:3px
classDef lbClass fill:#f3e5f5,stroke:#4a148c,stroke-width:2px
classDef webClass fill:#e8f5e8,stroke:#1b5e20,stroke-width:2px
classDef appClass fill:#fff3e0,stroke:#e65100,stroke-width:2px
classDef dbClass fill:#fce4ec,stroke:#880e4f,stroke-width:2px
classDef cacheClass fill:#f1f8e9,stroke:#33691e,stroke-width:2px
classDef monClass fill:#e3f2fd,stroke:#0d47a1,stroke-width:2px

```

```

class INTERNET internetClass
class LB1, LB2, VIP lbClass
class WEB1, WEB2, WEB3 webClass
class APP1, APP2, APP3 appClass
class DB1, DB2, DB3 dbClass
class CACHE1, CACHE2 cacheClass
class MON1, LOG1 monClass

```



Phase Development Workflow

```

gantt
    title HX Infrastructure Development Timeline
    dateFormat YYYY-MM-DD
    section Phase 1.0 - Foundation
    Directory Structure      :done, phase1-1, 2025-09-17, 1d
    Documentation Framework :done, phase1-2, 2025-09-17, 1d
    Visual Documentation     :done, phase1-3, 2025-09-17, 1d
    Basic Configuration      :done, phase1-4, 2025-09-17, 1d
    Testing Framework Setup  :done, phase1-5, 2025-09-17, 1d

    section Phase 2.0 - Core Implementation
    Common Roles Development :phase2-1, after phase1-5, 2d
    Web Tier Implementation  :phase2-2, after phase2-1, 2d
    Database Tier Setup      :phase2-3, after phase2-2, 2d
    Basic Playbooks          :phase2-4, after phase2-3, 2d
    Integration Testing       :phase2-5, after phase2-4, 1d

    section Phase 3.0 - Advanced Features
    Application Tier         :phase3-1, after phase2-5, 2d
    Cache Tier Implementation :phase3-2, after phase3-1, 2d
    Load Balancer Setup     :phase3-3, after phase3-2, 2d
    Security Hardening       :phase3-4, after phase3-3, 2d
    SSL/TLS Configuration   :phase3-5, after phase3-4, 1d

    section Phase 4.0 - Production Ready
    Monitoring Stack         :phase4-1, after phase3-5, 3d
    Logging Infrastructure    :phase4-2, after phase4-1, 2d
    Backup & Recovery        :phase4-3, after phase4-2, 2d
    CI/CD Pipeline          :phase4-4, after phase4-3, 2d
    Performance Optimization :phase4-5, after phase4-4, 2d
    Documentation Completion :phase4-6, after phase4-5, 1d

```

Variable Hierarchy Diagram

```

graph TB
    subgraph "Variable Hierarchy"
        GLOBAL[Global Variables<br/>📁 inventory/group_vars/all.yml<br/>🌐 Applies to all hosts<br/>🔧 Base configuration]

        subgraph "Environment Level"
            ENV_PROD[Production Variables<br/>📁 inventory/environments/production/<br/>📋 Production-specific settings]
            ENV_STAGE[Staging Variables<br/>📁 inventory/environments/staging/<br/>🧑 Staging-specific settings]
            ENV_DEV[Development Variables<br/>📁 inventory/environments/development/<br/>🔧 Development-specific settings]
        end

        subgraph "Group Level"
            GROUP_LB[Load Balancer Group<br/>📁 inventory/group_vars/load_balancers.yml<br/>📋 LB-specific configuration]
            GROUP_WEB[Web Group<br/>📁 inventory/group_vars/web_servers.yml<br/>🌐 Web-specific configuration]
            GROUP_APP[Application Group<br/>📁 inventory/group_vars/app_servers.yml<br/>🔧 App-specific configuration]
            GROUP_DB[Database Group<br/>📁 inventory/group_vars/database_servers.yml<br/>📋 DB-specific configuration]
            GROUP_CACHE[Cache Group<br/>📁 inventory/group_vars/cache_servers.yml<br/>⚡ Cache-specific configuration]
            GROUP_MON[Monitoring Group<br/>📁 inventory/group_vars/monitoring_servers.yml<br/>📊 Monitoring-specific configuration]
        end

        subgraph "Host Level"
            HOST_SPECIFIC[Host Variables<br/>📁 inventory/host_vars/<hostname>.yml<br/>💻 Host-specific overrides<br/>🔧 Individual customization]
        end

        subgraph "Role Level"
            ROLE_DEFAULTS[Role Defaults<br/>📁 roles/*/defaults/main.yml<br/>📋 Default role values<br/>🔧 Lowest priority]
            ROLE_VARS[Role Variables<br/>📁 roles/*/vars/main.yml<br/>📋 Role-specific values<br/>🔧 High priority]
        end

        subgraph "Runtime Level"
            EXTRA_VARS[Extra Variables<br/>🔧 Command line: -e "var=value"<br/>🔧 Highest priority<br/>⚡ Runtime overrides]
        end

        subgraph "Secrets Management"
            VAULT[Ansible Vault<br/>📁 vars/secrets.yml<br/>🔒 Encrypted secrets<br/>🔒 Sensitive data]
        end
    end

    end

    %% Precedence Flow (Higher to Lower)
    EXTRA_VARS --> ROLE_VARS
    ROLE_VARS --> HOST_SPECIFIC
    HOST_SPECIFIC --> GROUP_LB
    HOST_SPECIFIC --> GROUP_WEB
    HOST_SPECIFIC --> GROUP_APP
    HOST_SPECIFIC --> GROUP_DB
    HOST_SPECIFIC --> GROUP_CACHE
    HOST_SPECIFIC --> GROUP_MON

```

```

GROUP_LB --> ENV_PROD
GROUP_WEB --> ENV_STAGE
GROUP_APP --> ENV_DEV
GROUP_DB --> GLOBAL
GROUP_CACHE --> GLOBAL
GROUP_MON --> GLOBAL
ENV_PROD --> GLOBAL
ENV_STAGE --> GLOBAL
ENV_DEV --> GLOBAL
GLOBAL --> ROLE_DEFAULTS

%% Vault Integration
VAULT -. -> HOST_SPECIFIC
VAULT -. -> GROUP_LB
VAULT -. -> GROUP_WEB
VAULT -. -> GROUP_APP
VAULT -. -> GROUP_DB
VAULT -. -> GROUP_CACHE
VAULT -. -> GROUP_MON










%% Styling
classDef highPriority fill:#ffcdd2,stroke:#d32f2f,stroke-width:3px
classDef mediumPriority fill:#fff3e0,stroke:#f57c00,stroke-width:2px
classDef lowPriority fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
classDef secretClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px

class EXTRA_VARS highPriority
class ROLE_VARS,HOST_SPECIFIC mediumPriority
class GROUP_LB,GROUP_WEB,GROUP_APP,GROUP_DB,GROUP_CACHE,GROUP_MON mediumPriority
class ENV_PROD,ENV_STAGE,ENV_DEV,GLOBAL lowPriority
class ROLE_DEFAULTS lowPriority
class VAULT secretClass

```


Deployment Workflow Diagram

sequenceDiagram

participant User as  User/DevOps
 participant Ansible as  Ansible Controller
 participant Vault as  Ansible Vault
 participant DB as  Database Tier
 participant Cache as  Cache Tier
 participant App as  Application Tier
 participant Web as  Web Tier
 participant LB as  Load Balancer
 participant Monitor as  Monitoring

Note over User,Monitor: HX Infrastructure Deployment Sequence

User->>Ansible: 1. Execute site playbook

Note right of User: ansible-playbook -i inventory/environments/production
playbooks/site/main.yml

Ansible->>Vault: 2. Decrypt secrets

Vault-->>Ansible: 2a. Return decrypted secrets

Note over Ansible,Monitor: Phase 1: Database Tier

Ansible->>DB: 3. Deploy PostgreSQL cluster

DB->>DB: 3a. Configure **master**-replica setup

DB->>DB: 3b. Create application database

DB->>DB: 3c. Setup backup procedures

DB-->>Ansible: 3d. Database tier ready

Note over Ansible,Monitor: Phase 2: Cache Tier

Ansible->>Cache: 4. Deploy Redis cluster

Cache->>Cache: 4a. Configure **master**-replica

Cache->>Cache: 4b. Setup persistence

Cache-->>Ansible: 4c. Cache tier ready

Note over Ansible,Monitor: Phase 3: Application Tier

Ansible->>App: 5. Deploy application servers

App->>DB: 5a. Test database connectivity

App->>Cache: 5b. Test cache connectivity

App->>App: 5c. Start application services

App-->>Ansible: 5d. Application tier ready

Note over Ansible,Monitor: Phase 4: Web Tier

Ansible->>Web: 6. Deploy web servers

Web->>App: 6a. Configure upstream connections

Web->>Web: 6b. Setup SSL certificates

Web->>Web: 6c. Configure caching

Web-->>Ansible: 6d. Web tier ready

Note over Ansible,Monitor: Phase 5: Load Balancer

Ansible->>LB: 7. Deploy **load** balancers

LB->>Web: 7a. Configure backend pools

LB->>LB: 7b. Setup keepalived (HA)

LB->>LB: 7c. Configure SSL termination

LB-->>Ansible: 7d. Load balancer ready

Note over Ansible,Monitor: Phase 6: Monitoring

Ansible->>Monitor: 8. Deploy monitoring stack

Monitor->>DB: 8a. Setup database monitoring

Monitor->>Cache: 8b. Setup cache monitoring

Monitor->>App: 8c. Setup application monitoring

Monitor->>Web: 8d. Setup web monitoring

Monitor->>LB: 8e. Setup **load** balancer monitoring

Monitor->>Monitor: 8f. Configure dashboards & alerts

Monitor-->>Ansible: 8g. Monitoring active

Note over Ansible,Monitor: Phase 7: Final Validation

Ansible->>LB: 9. Run health checks

LB->>Web: 9a. Validate web tier

Web->>App: 9b. Validate application tier

App->>DB: 9c. Validate database tier

App->>Cache: 9d. Validate cache tier

Monitor->>Monitor: 9e. Validate monitoring

Ansible-->>User: 10. Deployment complete

Note right of Ansible:  All 15 servers deployed
 All services running



Monitoring active

Git Workflow Diagram

```

gitgraph
  commit id: "Initial Setup"
  branch develop
  checkout develop
  commit id: "Phase 1.0 Foundation"

  branch feature/phase-1-docs
  checkout feature/phase-1-docs
  commit id: "Add documentation"
  commit id: "Add visual diagrams"
  commit id: "Update README"

  checkout develop
  merge feature/phase-1-docs
  commit id: "Merge Phase 1 docs"

  branch feature/phase-2-core
  checkout feature/phase-2-core
  commit id: "Add common roles"
  commit id: "Add web tier"
  commit id: "Add database tier"
  commit id: "Add basic playbooks"

  checkout develop
  merge feature/phase-2-core
  commit id: "Merge Phase 2 core"

  branch feature/phase-3-advanced
  checkout feature/phase-3-advanced
  commit id: "Add app tier"
  commit id: "Add cache tier"
  commit id: "Add load balancer"
  commit id: "Add security hardening"

  checkout develop
  merge feature/phase-3-advanced
  commit id: "Merge Phase 3 advanced"

  branch feature/phase-4-production
  checkout feature/phase-4-production
  commit id: "Add monitoring stack"
  commit id: "Add logging infrastructure"
  commit id: "Add backup & recovery"
  commit id: "Add CI/CD pipeline"

  checkout develop
  merge feature/phase-4-production
  commit id: "Merge Phase 4 production"

  checkout main
  merge develop
  commit id: "Release v1.0.0"
  tag: "v1.0.0"

  checkout develop
  branch hotfix/security-patch
  checkout hotfix/security-patch
  commit id: "Security patch"

  checkout main
  merge hotfix/security-patch
  commit id: "Hotfix v1.0.1"

```

```
tag: "v1.0.1"
```

```
checkout develop
```

```
merge hotfix/security-patch
```

```
commit id: "Merge hotfix to develop"
```



CI/CD Pipeline Diagram

```

graph TB
  subgraph "Source Control"
    GIT[Git Repository<br/>📁 GitHub<br/>🔒 Branch Protection<br/>📋 Pull Re-
    quests]
  end

  subgraph "Continuous Integration"
    TRIGGER[Pipeline Trigger<br/>🔄 Push/PR Events<br/>⚡ Webhook Integration]

    subgraph "Code Quality Stage"
      LINT[Linting<br/>📝 ansible-lint<br/>📄 yamllint<br/>🐛 flake8]
      SYNTAX[Syntax Check<br/>✅ Playbook validation<br/>🔍 YAML validation]
      SECURITY[Security Scan<br/>🔒 Vulnerability check<br/>🛡️ Secret detection]
    end

    subgraph "Testing Stage"
      UNIT[Unit Tests<br/>🔧 Molecule tests<br/>🧑 Role validation]
      INTEGRATION[Integration Tests<br/>🔗 Multi-role testing<br/>🔧 Testinfra v
      alidation]
      SMOKE[Smoke Tests<br/>🌪️ Basic functionality<br/>⚡ Quick validation]
    end

    subgraph "Build Stage"
      BUILD[Build Artifacts<br/>📦 Package creation<br/>🔧 Asset preparation]
      VALIDATE[Validation<br/>✅ Artifact integrity<br/>🔍 Dependency check]
    end
  end

  subgraph "Continuous Deployment"
    subgraph "Development Environment"
      DEV_DEPLOY[Deploy to Dev<br/>🔧 Development servers<br/>🔧 Feature test-
      ing]
      DEV_TEST[Dev Testing<br/>🔍 Functional tests<br/>📊 Performance baseline]
    end

    subgraph "Staging Environment"
      STAGE_DEPLOY[Deploy to Staging<br/>🧑 Pre-production<br/>🔄 Production mir-
      ror]
      STAGE_TEST[Staging Testing<br/>🔧 E2E tests<br/>📊 Load testing<br/>🔒 Se-
      curity testing]
      UAT[User Acceptance<br/>👥 Stakeholder approval<br/>✅ Business valida-
      tion]
    end

    subgraph "Production Environment"
      PROD_APPROVAL[Production Approval<br/>👤 Manual gate<br/>📋 Change manage-
      ment]
      PROD_DEPLOY[Deploy to Production<br/>🏢 Live environment<br/>🔄 Rolling de-
      ployment]
      PROD_VALIDATE[Production Validation<br/>📊 Health checks<br/>🚨
      Monitoring alerts]
    end

    subgraph "Monitoring & Feedback"
      MONITOR[Monitoring<br/>📊 Prometheus metrics<br/>📈 Grafana dashboards]
      ALERTS[Alerting<br/>🚨 Alert manager<br/>📧 Notifications]
      LOGS[Logging<br/>📝 ELK stack<br/>🔍 Log analysis]
      FEEDBACK[Feedback Loop<br/>🔄 Continuous improvement<br/>📊 Performance met-
      rics]
    end
  end

```



```

%% Flow Connections
GIT --> TRIGGER
TRIGGER --> LINT
TRIGGER --> SYNTAX
TRIGGER --> SECURITY

LINT --> UNIT
SYNTAX --> UNIT
SECURITY --> UNIT

UNIT --> INTEGRATION
INTEGRATION --> SMOKE

SMOKE --> BUILD
BUILD --> VALIDATE

VALIDATE --> DEV_DEPLOY
DEV_DEPLOY --> DEV_TEST

DEV_TEST --> STAGE_DEPLOY
STAGE_DEPLOY --> STAGE_TEST
STAGE_TEST --> UAT

UAT --> PROD_APPROVAL
PROD_APPROVAL --> PROD_DEPLOY
PROD_DEPLOY --> PROD_VALIDATE

PROD_VALIDATE --> MONITOR
MONITOR --> ALERTS
MONITOR --> LOGS
ALERTS --> FEEDBACK
LOGS --> FEEDBACK
FEEDBACK --> GIT

%% Styling
classDef sourceClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
classDef ciClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
classDef cdClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
classDef monitorClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px

class GIT sourceClass
class TRIGGER,LINT,SYNTAX,SECURITY,UNIT,INTEGRATION,SMOKE,BUILD,VALIDATE ciClass
class DEV_DEPLOY,DEV_TEST,STAGE_DEPLOY,STAGE_TEST,UAT,PROD_APPROVAL,PROD_DEPLOY,PROD_VALIDATE cdClass
class MONITOR,ALERTS,LOGS,FEEDBACK monitorClass

```

Secrets Management Diagram

```

graph TB
    subgraph "Secret Sources"
        VAULT_FILE[Ansible Vault Files<br/>📁 vars/secrets.yml<br/>🔒 Encrypted at rest<br/>🔒 Vault password protected]
        EXTERNAL_VAULT[External Vault<br/>🏛️ HashiCorp Vault<br/>🔒 Centralized secrets<br/>🔄 Dynamic secrets]
        ENV_VARS[Environment Variables<br/>🌐 Runtime secrets<br/>⚡ Temporary access<br/>🔒 Process isolation]
        KEY_MGMT[Key Management<br/>🔑 SSH keys<br/>📜 SSL certificates<br/>📄 API tokens]
    end

    subgraph "Secret Processing"
        DECRYPT[Decryption Process<br/>🔒 Vault password<br/>🔒 AES-256 encryption<br/>⚡ Runtime decryption]
        INJECT[Secret Injection<br/>🔧 Template rendering<br/>🔗 Variable substitution<br/>🎯 Target-specific]
        VALIDATE[Validation<br/>✅ Secret format check<br/>🕒 Expiration validation<br/>🛡️ Access control]
    end


    subgraph "Secret Distribution"
        PLAYBOOK[Playbook Execution<br/>📄 Ansible playbooks<br/>🎯 Target hosts<br/>🔒 Secure transport]
        TEMPLATE[Template Rendering<br/>📄 Jinja2 templates<br/>🔗 Configuration files<br/>🔄 Dynamic content]
        SERVICE_CONFIG[Service Configuration<br/>⚙️ Application configs<br/>📁 Data-base credentials<br/>🔑 API keys]
    end

    subgraph "Security Controls"
        ENCRYPTION[Encryption in Transit<br/>🔒 SSH/TLS encryption<br/>🛡️ Secure channels<br/>🔒 End-to-end security]
        ACCESS_CONTROL[Access Control<br/>👤 RBAC implementation<br/>🔒 Principle of least privilege<br/>📄 Audit logging]
        ROTATION[Secret Rotation<br/>🔄 Automated rotation<br/>🕒 Scheduled updates<br/>🔄 Zero-downtime rotation]
        AUDIT[Audit & Compliance<br/>📊 Access logging<br/>🕒 Usage tracking<br/>📄 Compliance reporting]
    end

    subgraph "Target Systems"
        DATABASES[Database Systems<br/>🗄️ PostgreSQL<br/>🔑 Connection credentials<br/>🔒 Encrypted connections]
        APPLICATIONS[Applications<br/>⚙️ App servers<br/>🔑 API keys<br/>📄 Service tokens]
        LOAD_BALANCERS[Load Balancers<br/>⚖️ SSL certificates<br/>🔒 TLS termination<br/>🔒 Secure backends]
    end

```

```

MONITORING[Monitoring Systems<br/> Metrics collection<br/> Service ac-
counts<br/> Alert credentials]
end

%% Flow Connections
VAULT_FILE --> DECRYPT
EXTERNAL_VAULT --> DECRYPT
ENV_VARS --> INJECT
KEY_MGMT --> VALIDATE

DECRYPT --> INJECT
INJECT --> VALIDATE

VALIDATE --> PLAYBOOK
PLAYBOOK --> TEMPLATE
TEMPLATE --> SERVICE_CONFIG

SERVICE_CONFIG --> ENCRYPTION
ENCRYPTION --> ACCESS_CONTROL
ACCESS_CONTROL --> ROTATION
ROTATION --> AUDIT

ENCRYPTION --> DATABASES
ENCRYPTION --> APPLICATIONS
ENCRYPTION --> LOAD_BALANCERS
ENCRYPTION --> MONITORING

%% Security Feedback Loops
AUDIT -. -> ACCESS_CONTROL
ROTATION -. -> VAULT_FILE
ROTATION -. -> EXTERNAL_VAULT

%% Styling
classDef sourceClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
classDef processClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
classDef distributionClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
classDef securityClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px
classDef targetClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px

class VAULT_FILE,EXTERNAL_VAULT,ENV_VARS,KEY_MGMT sourceClass
class DECRYPT,INJECT,VALIDATE processClass
class PLAYBOOK,TEMPLATE,SERVICE_CONFIG distributionClass
class ENCRYPTION,ACCESS_CONTROL,ROTATION,AUDIT securityClass
class DATABASES,APPLICATIONS,LOAD_BALANCERS,MONITORING targetClass

```



Monitoring Architecture Diagram

```

graph TB
    subgraph "Data Collection Layer"
        subgraph "System Metrics"
            NODE_EXP[Node Exporter<br/>📊 System metrics<br/>💾 CPU, Memory, Disk<br/>🌐 Network statistics<br/>📈 Host-level monitoring]

            PROCESS_EXP[Process Exporter<br/>⚙️ Process metrics<br/>📊 Resource usage<br/>🔍 Process monitoring]
        end

        subgraph "Application Metrics"
            APP_METRICS[Application Metrics<br/>📈 Custom metrics<br/>🎯 Business KPIs<br/>⚡ Performance counters<br/>🔍 Error tracking]

            JVM_METRICS[JVM Metrics<br/>☕ Java applications<br/>💾 Heap usage<br/>🗑️ Garbage collection<br/>🧵 Thread pools]
        end

        subgraph "Infrastructure Metrics"
            POSTGRES_EXP[PostgreSQL Exporter<br/>📄 Database metrics<br/>📊 Query performance<br/>🔄 Replication status<br/>💾 Storage usage]

            REDIS_EXP[Redis Exporter<br/>⚡ Cache metrics<br/>📊 Hit/miss ratios<br/>💾 Memory usage<br/>🔄 Replication lag]

            NGINX_EXP[Nginx Exporter<br/>🌐 Web server metrics<br/>📊 Request rates<br/>🕒 Response times<br/>🔍 Error rates]
        end

        subgraph "Log Collection"
            FILEBEAT[Filebeat<br/>📝 Log shipping<br/>📁 File monitoring<br/>🔄 Real-time streaming<br/>🎯 Multi-line parsing]

            LOGSTASH[Logstash<br/>🔄 Log processing<br/>🧩 Data transformation<br/>📊 Enrichment<br/>🎯 Routing]
        end
    end

    subgraph "Data Storage Layer"
        PROMETHEUS[Prometheus<br/>📊 Time series database<br/>🕒 Metrics storage<br/>🔍 Query engine<br/>📈 90-day retention]

        ELASTICSEARCH[Elasticsearch<br/>📝 Log storage<br/>🔍 Full-text search<br/>📊 Aggregations<br/>📈 30-day retention]
    end

    subgraph "Visualization Layer"
        GRAFANA[Grafana<br/>📈 Dashboards<br/>📊 Visualization<br/>🧩 Custom panels<br/>👥 Multi-tenancy]

        KIBANA[Kibana<br/>📊 Log analysis<br/>🔍 Search interface<br/>📈 Visualizations<br/>🎯 Discover & analyze]
    end

    subgraph "Alerting Layer"
        ALERTMANAGER[Alertmanager<br/>📢 Alert routing<br/>📧 Notifications<br/>🔇 Silencing<br/>👥 Team routing]

        subgraph "Notification Channels"
            EMAIL[Email Notifications<br/>📧 SMTP delivery<br/>👥 Team distribution<br/>📄 Alert details]
        end
    end

```

```

    SLACK[Slack Integration<br/>💬 Team channels<br/>🤖 Bot notifications<br/>
    📶 Channel routing]

    PAGERDUTY[PagerDuty<br/>📱 Incident management<br/>🔥 Escalation
    policies<br/>☎ On-call rotation]
    end
end

subgraph "Target Infrastructure"
    LB_TARGETS[Load Balancers<br/>⚖ lb-01, lb-02<br/>📊 Traffic metrics<br/>🔍 Health status]

    WEB_TARGETS[Web Servers<br/>🌐 web-01, web-02, web-03<br/>📊 Request metrics<br/>🕒 Response times]

    APP_TARGETS[App Servers<br/>⚙ app-01, app-02, app-03<br/>📊 Application metrics<br/>💾 Resource usage]

    DB_TARGETS[Database Servers<br/>🗄 db-01, db-02, db-03<br/>📊 Query metrics<br/>🔄 Replication status]

    CACHE_TARGETS[Cache Servers<br/>⚡ cache-01, cache-02<br/>📊 Cache metrics<br/>💾 Memory usage]
end

%% Data Flow - Metrics
LB_TARGETS --> NODE_EXP
LB_TARGETS --> NGINX_EXP
WEB_TARGETS --> NODE_EXP
WEB_TARGETS --> NGINX_EXP
APP_TARGETS --> NODE_EXP
APP_TARGETS --> APP_METRICS
APP_TARGETS --> JVM_METRICS
DB_TARGETS --> NODE_EXP
DB_TARGETS --> POSTGRES_EXP
CACHE_TARGETS --> NODE_EXP
CACHE_TARGETS --> REDIS_EXP

NODE_EXP --> PROMETHEUS
PROCESS_EXP --> PROMETHEUS
APP_METRICS --> PROMETHEUS
JVM_METRICS --> PROMETHEUS
POSTGRES_EXP --> PROMETHEUS
REDIS_EXP --> PROMETHEUS
NGINX_EXP --> PROMETHEUS

%% Data Flow - Logs
LB_TARGETS --> FILEBEAT
WEB_TARGETS --> FILEBEAT
APP_TARGETS --> FILEBEAT
DB_TARGETS --> FILEBEAT
CACHE_TARGETS --> FILEBEAT

FILEBEAT --> LOGSTASH
LOGSTASH --> ELASTICSEARCH

%% Visualization
PROMETHEUS --> GRAFANA
ELASTICSEARCH --> KIBANA

%% Alerting

```

```









PROMETHEUS --> ALERTMANAGER
ALERTMANAGER --> EMAIL
ALERTMANAGER --> SLACK
ALERTMANAGER --> PAGERDUTY

%% Styling
classDef collectionClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
classDef storageClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
classDef visualClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
classDef alertClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px
classDef targetClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px

class NODE_EXP,PROCESS_EXP,APP_METRICS,JVM_METRICS,POSTGRES_EXP,REDIS_EXP,NGINX_EXP
,FILEBEAT,LOGSTASH collectionClass
class PROMETHEUS,ELASTICSEARCH storageClass
class GRAFANA,KIBANA visualClass
class ALERTMANAGER,EMAIL,SLACK,PAGERDUTY alertClass
class LB_TARGETS,WEB_TARGETS,APP_TARGETS,DB_TARGETS,CACHE_TARGETS targetClass

```


Service Orchestration Diagram

```
sequenceDiagram
    participant User as  End User
    participant CDN as  CDN/Edge
    participant LB as  Load Balancer
    participant Web as  Web Server
    participant App as  Application
    participant Cache as  Redis Cache
    participant DB as  Database
    participant Monitor as  Monitoring

    Note over User,Monitor: Request Processing Flow

    User->>CDN: 1. HTTP/HTTPS Request
    Note right of User: GET /api/data

    CDN->>CDN: 2. Check edge cache
    alt Cache Hit
        CDN-->>User: 2a. Return cached content
    else Cache Miss
        CDN->>LB: 2b. Forward to origin
    end

    LB->>LB: 3. Health check backends
    LB->>LB: 4. Apply load balancing algorithm
    Note right of LB: Round-robin selection

    LB->>Web: 5. Route to web server
    Note right of LB: Selected: web-02

    Web->>Web: 6. Check local cache
    alt Static Content
        Web-->>LB: 6a. Serve static files
    else Dynamic Content
        Web->>App: 6b. Proxy to application
    end

    App->>Cache: 7. Check Redis cache
    Note right of App: GET user:123:profile

    alt Cache Hit
        Cache-->>App: 7a. Return cached data
        Note right of Cache: Cache hit - fast response
    else Cache Miss
        App->>DB: 7b. Query database
        Note right of App: SELECT * FROM users WHERE id=123

        DB->>DB: 7c. Execute query
        alt Master Query (Write)
            Note right of DB: Route to master: db-01
        else Read Query
            Note right of DB: Route to replica: db-02/db-03
        end

        DB-->>App: 7d. Return query results

        App->>Cache: 7e. Store in cache
        Note right of App: SET user:123:profile TTL=3600
        Cache-->>App: 7f. Cache stored
    end

    App->>App: 8. Process business logic
    App->>App: 9. Generate response
```

```

App-->>Web: 10. Return processed data
Web-->>LB: 11. Return response
LB-->>CDN: 12. Return to CDN
CDN-->>CDN: 13. Cache response (if cacheable)
CDN-->>User: 14. Final response

Note over User,Monitor: Monitoring & Logging

par Metrics Collection
  LB-->>Monitor: Load balancer metrics
  Web-->>Monitor: Web server metrics
  App-->>Monitor: Application metrics
  Cache-->>Monitor: Cache metrics
  DB-->>Monitor: Database metrics
and Log Aggregation
  LB-->>Monitor: Access logs
  Web-->>Monitor: Access & error logs
  App-->>Monitor: Application logs
  DB-->>Monitor: Query logs
end

Monitor-->>Monitor: Process metrics & logs

alt Alert Condition
  Monitor-->>Monitor: Trigger alert
  Note right of Monitor: High response time detected
end

Note over User,Monitor: End-to-End Request Complete

```

This comprehensive visual documentation provides detailed diagrams for all aspects of the HX Infrastructure project, enabling clear understanding of architecture, workflows, and processes.