HX Infrastructure Inventory Documentation

📊 Comprehensive Inventory Structure and Visual Diagrams

This document provides detailed visual documentation for the HX Infrastructure inventory structure, showing environment organization, host groupings, service relationships, and variable inheritance patterns.

🏗️ Environment Overview Diagram

```
graph TB
    subgraph "HX Infrastructure Environments"
        subgraph "Development Environment"
            DEV_DOMAIN[Domain: dev-test.hana-x.ai<br/>Network: 192.168.10.0/24<br/>15 Servers]

            subgraph "Dev Services"
                DEV_INFRA[Infrastructure<br/>🏢 DC + CA<br/>2 servers]
                DEV_AI[AI/ML Services<br/>🤖 LiteLLM + Ollama<br/>4 servers]
                DEV_UI[UI Services<br/>🌐 WebUI + LB<br/>3 servers]
                DEV_OPS[Operations<br/>🔧 DB + Cache + Monitor<br/>6 servers]
            end
        end

        subgraph "Test Environment"
            TEST_DOMAIN[Domain: test.hana-x.ai<br/>Network: 192.168.20.0/24<br/>8 Servers]

            subgraph "Test Services"
                TEST_INFRA[Infrastructure<br/>🏢 DC Only<br/>1 server]
                TEST_AI[AI/ML Services<br/>🤖 CPU-only Testing<br/>2 servers]
                TEST_UI[UI Services<br/>🌐 Basic WebUI<br/>2 servers]
                TEST_OPS[Operations<br/>🔧 Essential Services<br/>3 servers]
            end
        end

        subgraph "Production Environment"
            PROD_DOMAIN[Domain: hana-x.ai<br/>Network: 10.0.0.0/16<br/>22 Servers]

            subgraph "Prod Services"
                PROD_INFRA[Infrastructure<br/>🏢 HA DC + CA<br/>4 servers]
                PROD_AI[AI/ML Services<br/>🤖 GPU Cluster<br/>6 servers]
                PROD_UI[UI Services<br/>🌐 HA WebUI + LB<br/>5 servers]
                PROD_OPS[Operations<br/>🔧 Enterprise Grade<br/>7 servers]
            end
        end
    end

    DEV_INFRA --> DEV_AI
    DEV_AI --> DEV_UI
    DEV_UI --> DEV_OPS

    TEST_INFRA --> TEST_AI
    TEST_AI --> TEST_UI
    TEST_UI --> TEST_OPS

    PROD_INFRA --> PROD_AI
    PROD_AI --> PROD_UI
    PROD_UI --> PROD_OPS

    %% Styling
    classDef devClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef testClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
    classDef prodClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px

    class DEV_DOMAIN,DEV_INFRA,DEV_AI,DEV_UI,DEV_OPS devClass
    class TEST_DOMAIN,TEST_INFRA,TEST_AI,TEST_UI,TEST_OPS testClass
    class PROD_DOMAIN,PROD_INFRA,PROD_AI,PROD_UI,PROD_OPS prodClass
```

🔗 Service Grouping Structure

```
graph TB
    subgraph "HX Infrastructure Service Groups"
        subgraph "Infrastructure Services"
            DC[Domain Controllers<br/>🏢 Active Directory<br/>DNS, DHCP, GPO<br/>Windows Server]
            CA[Certificate Authorities<br/>🔐 PKI Infrastructure<br/>SSL Certificates<br/>Code Signing]
        end

        subgraph "AI/ML Services"
            LITELLM[LiteLLM Gateway<br/>🚪 API Gateway<br/>Model Routing<br/>Rate Limiting]
            OLLAMA[Ollama Inference<br/>🧠 Local LLMs<br/>GPU Acceleration<br/>Model Management]
            MODELS[Model Storage<br/>📦 Model Repository<br/>Version Control<br/>Registry Service]
        end

        subgraph "UI Services"
            WEBUI[Web Interface<br/>💬 Chat Interface<br/>User Management<br/>Admin Panel]
            LB[Load Balancers<br/>⚖️ Traffic Distribution<br/>SSL Termination<br/>High Availability]
        end

        subgraph "Operations Services"
            DB[PostgreSQL Cluster<br/>🗄️ Primary Database<br/>Replication<br/>Backup & Recovery]
            CACHE[Redis Cluster<br/>⚡ Caching Layer<br/>Session Storage<br/>Pub/Sub Messaging]
            MONITOR[Monitoring Stack<br/>📊 Prometheus + Grafana<br/>Metrics Collection<br/>Alerting]
            LOGS[Logging Stack<br/>📝 ELK Stack<br/>Log Aggregation<br/>Search & Analysis]
        end
    end

    %% Service Dependencies
    DC --> CA
    CA --> LB
    LB --> WEBUI
    WEBUI --> LITELLM
    LITELLM --> OLLAMA
    OLLAMA --> MODELS
    WEBUI --> DB
    WEBUI --> CACHE

    %% Monitoring Connections
    MONITOR -.-> DC
    MONITOR -.-> CA
    MONITOR -.-> LITELLM
    MONITOR -.-> OLLAMA
    MONITOR -.-> WEBUI
    MONITOR -.-> LB
    MONITOR -.-> DB
    MONITOR -.-> CACHE

    %% Logging Connections
    LOGS -.-> DC
    LOGS -.-> CA
    LOGS -.-> LITELLM
    LOGS -.-> OLLAMA
```

```
    LOGS -.-> WEBUI
    LOGS -.-> LB
    LOGS -.-> DB
    LOGS -.-> CACHE
    LOGS -.-> MONITOR

    %% Styling
    classDef infraClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px
    classDef aiClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
    classDef uiClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef opsClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px

    class DC,CA infraClass
    class LITELLM,OLLAMA,MODELS aiClass
    class WEBUI,LB uiClass
    class DB,CACHE,MONITOR,LOGS opsClass
```

🌐 Network Topology by Environment

Development Environment Network

```
graph TB
    subgraph "Development Network - 192.168.10.0/24"
        subgraph "Infrastructure Subnet - .10-.19"
            DC_DEV[hx-dc-01<br/>192.168.10.10<br/>Domain Controller]
            CA_DEV[hx-ca-01<br/>192.168.10.11<br/>Certificate Authority]
        end

        subgraph "AI/ML Subnet - .20-.29"
            LITELLM_DEV[hx-litellm-01<br/>192.168.10.20<br/>LiteLLM Gateway]
            OLLAMA1_DEV[hx-ollama-01<br/>192.168.10.21<br/>Ollama Primary]
            OLLAMA2_DEV[hx-ollama-02<br/>192.168.10.22<br/>Ollama Secondary]
            MODELS_DEV[hx-models-01<br/>192.168.10.23<br/>Model Storage]
        end

        subgraph "UI Subnet - .30-.39"
            WEBUI1_DEV[hx-webui-01<br/>192.168.10.30<br/>WebUI Primary]
            WEBUI2_DEV[hx-webui-02<br/>192.168.10.31<br/>WebUI Secondary]
            LB_DEV[hx-lb-01<br/>192.168.10.32<br/>Load Balancer<br/>VIP: .100]
        end

        subgraph "Operations Subnet - .40-.59"
            DB1_DEV[hx-db-01<br/>192.168.10.40<br/>PostgreSQL Master]
            DB2_DEV[hx-db-02<br/>192.168.10.41<br/>PostgreSQL Replica]
            REDIS1_DEV[hx-redis-01<br/>192.168.10.42<br/>Redis Master]
            REDIS2_DEV[hx-redis-02<br/>192.168.10.43<br/>Redis Replica]
            MON_DEV[hx-monitor-01<br/>192.168.10.50<br/>Monitoring]
            LOG_DEV[hx-logs-01<br/>192.168.10.51<br/>Logging]
        end

        GATEWAY_DEV[Gateway<br/>192.168.10.1]
    end

    %% Network Connections
    GATEWAY_DEV --> LB_DEV
    LB_DEV --> WEBUI1_DEV
    LB_DEV --> WEBUI2_DEV
    WEBUI1_DEV --> LITELLM_DEV
    WEBUI2_DEV --> LITELLM_DEV
    LITELLM_DEV --> OLLAMA1_DEV
    LITELLM_DEV --> OLLAMA2_DEV
    OLLAMA1_DEV --> MODELS_DEV
    OLLAMA2_DEV --> MODELS_DEV
    WEBUI1_DEV --> DB1_DEV
    WEBUI2_DEV --> DB1_DEV
    DB1_DEV --> DB2_DEV
    WEBUI1_DEV --> REDIS1_DEV
    WEBUI2_DEV --> REDIS1_DEV
    REDIS1_DEV --> REDIS2_DEV

    %% Styling
    classDef infraClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px
    classDef aiClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
    classDef uiClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef opsClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px
    classDef netClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px

    class DC_DEV,CA_DEV infraClass
    class LITELLM_DEV,OLLAMA1_DEV,OLLAMA2_DEV,MODELS_DEV aiClass
    class WEBUI1_DEV,WEBUI2_DEV,LB_DEV uiClass
    class DB1_DEV,DB2_DEV,REDIS1_DEV,REDIS2_DEV,MON_DEV,LOG_DEV opsClass
    class GATEWAY_DEV netClass
```

Production Environment Network

```
graph TB
    subgraph "Production Network - 10.0.0.0/16"
        subgraph "Infrastructure Tier - 10.0.1.0/24"
            subgraph "AZ-1"
                DC1_PROD[hx-dc-prod-01<br/>10.0.1.10<br/>Primary DC]
                CA1_PROD[hx-ca-prod-01<br/>10.0.1.20<br/>Primary CA]
            end
            subgraph "AZ-2"
                DC2_PROD[hx-dc-prod-02<br/>10.0.1.11<br/>Secondary DC]
                CA2_PROD[hx-ca-prod-02<br/>10.0.1.21<br/>Secondary CA]
            end
        end

        subgraph "AI/ML Tier - 10.0.2.0/24"
            subgraph "AZ-1"
                LITELLM1_PROD[hx-litellm-prod-01<br/>10.0.2.10<br/>LiteLLM Primary]
                OLLAMA1_PROD[hx-ollama-prod-01<br/>10.0.2.20<br/>Ollama Primary]
                OLLAMA3_PROD[hx-ollama-prod-03<br/>10.0.2.22<br/>Ollama Tertiary]
                MODELS1_PROD[hx-models-prod-01<br/>10.0.2.30<br/>Model Primary]
            end
            subgraph "AZ-2"
                LITELLM2_PROD[hx-litellm-prod-02<br/>10.0.2.11<br/>LiteLLM Secondary]
                OLLAMA2_PROD[hx-ollama-prod-02<br/>10.0.2.21<br/>Ollama Secondary]
                MODELS2_PROD[hx-models-prod-02<br/>10.0.2.31<br/>Model Replica]
            end
        end

        subgraph "UI Tier - 10.0.3.0/24"
            subgraph "AZ-1"
                WEBUI1_PROD[hx-webui-prod-01<br/>10.0.3.10<br/>WebUI Primary]
                WEBUI3_PROD[hx-webui-prod-03<br/>10.0.3.12<br/>WebUI Tertiary]
                LB1_PROD[hx-lb-prod-01<br/>10.0.3.20<br/>LB Primary<br/>VIP: .100]
            end
            subgraph "AZ-2"
                WEBUI2_PROD[hx-webui-prod-02<br/>10.0.3.11<br/>WebUI Secondary]
                LB2_PROD[hx-lb-prod-02<br/>10.0.3.21<br/>LB Secondary<br/>VIP: .100]
            end
        end

        subgraph "Data Tier - 10.0.4.0/24"
            subgraph "AZ-1"
                DB1_PROD[hx-db-prod-01<br/>10.0.4.10<br/>PostgreSQL Master]
                DB3_PROD[hx-db-prod-03<br/>10.0.4.12<br/>PostgreSQL Replica 2]
                REDIS1_PROD[hx-redis-prod-01<br/>10.0.4.20<br/>Redis Master]
                REDIS3_PROD[hx-redis-prod-03<br/>10.0.4.22<br/>Redis Replica 2]
            end
            subgraph "AZ-2"
                DB2_PROD[hx-db-prod-02<br/>10.0.4.11<br/>PostgreSQL Replica 1]
                REDIS2_PROD[hx-redis-prod-02<br/>10.0.4.21<br/>Redis Replica 1]
            end
        end

        subgraph "Management Tier - 10.0.5.0/24"
            subgraph "AZ-1"
                MON1_PROD[hx-monitor-prod-01<br/>10.0.5.10<br/>Monitoring Primary]
                LOG1_PROD[hx-logs-prod-01<br/>10.0.5.20<br/>Logging Primary]
            end
            subgraph "AZ-2"
                MON2_PROD[hx-monitor-prod-02<br/>10.0.5.11<br/>Monitoring Secondary]
                LOG2_PROD[hx-logs-prod-02<br/>10.0.5.21<br/>Logging Secondary]
            end
        end
```

```
    end

    %% High Availability Connections
    DC1_PROD <--> DC2_PROD
    CA1_PROD <--> CA2_PROD
    LITELLM1_PROD <--> LITELLM2_PROD
    OLLAMA1_PROD <--> OLLAMA2_PROD
    MODELS1_PROD <--> MODELS2_PROD
    LB1_PROD <--> LB2_PROD
    DB1_PROD --> DB2_PROD
    DB1_PROD --> DB3_PROD
    REDIS1_PROD --> REDIS2_PROD
    REDIS1_PROD --> REDIS3_PROD
    MON1_PROD <--> MON2_PROD
    LOG1_PROD <--> LOG2_PROD

    %% Styling
    classDef az1Class fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef az2Class fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px

    class DC1_PROD,CA1_PROD,LITELLM1_PROD,OLLAMA1_PROD,OLLAMA3_PROD,MODELS1_PROD,WEBUI1
_PROD,WEBUI3_PROD,LB1_PROD,DB1_PROD,DB3_PROD,REDIS1_PROD,REDIS3_PROD,MON1_PROD,LOG1_PRO
D az1Class
    class DC2_PROD,CA2_PROD,LITELLM2_PROD,OLLAMA2_PROD,MODELS2_PROD,WEBUI2_PROD,LB2_PRO
D,DB2_PROD,REDIS2_PROD,MON2_PROD,LOG2_PROD az2Class
```

📊 Variable Inheritance Hierarchy

```
graph TB
    subgraph "Ansible Variable Precedence (High to Low)"
        EXTRA[Extra Variables<br/>🚀 ansible-playbook -e<br/>Highest Priority<br/>Runtime Overrides]

        HOST_VARS[Host Variables<br/>📁 host_vars/<hostname>.yml<br/>Host-specific Settings<br/>Individual Customization]

        subgraph "Group Variables"
            GROUP_SPECIFIC[Specific Groups<br/>📁 group_vars/infrastructure.yml<br/>📁 group_vars/ai_ml.yml<br/>📁 group_vars/ui.yml<br/>📁 group_vars/operations.yml]

            GROUP_ALL[All Group<br/>📁 group_vars/all.yml<br/>Global Defaults<br/>Common Configuration]
        end

        subgraph "Role Variables"
            ROLE_VARS[Role Variables<br/>📁 roles/*/vars/main.yml<br/>Role-specific Values<br/>High Priority within Role]

            ROLE_DEFAULTS[Role Defaults<br/>📁 roles/*/defaults/main.yml<br/>Default Values<br/>Lowest Priority]
        end

        subgraph "Inventory Variables"
            INV_HOST[Inventory Host Vars<br/>📁 inventory/host_vars/<hostname>.yml<br/>Environment-specific Host Settings]

            INV_GROUP[Inventory Group Vars<br/>📁 inventory/group_vars/<group>.yml<br/>Environment-specific Group Settings]
        end

        subgraph "Secrets Management"
            VAULT[Ansible Vault<br/>🔐 Encrypted Variables<br/>Sensitive Data<br/>Cross-referenced in other vars]
        end
    end

    %% Precedence Flow (Higher to Lower)
    EXTRA --> HOST_VARS
    HOST_VARS --> GROUP_SPECIFIC
    GROUP_SPECIFIC --> GROUP_ALL
    GROUP_ALL --> INV_HOST
    INV_HOST --> INV_GROUP
    INV_GROUP --> ROLE_VARS
    ROLE_VARS --> ROLE_DEFAULTS

    %% Vault Integration (dotted lines show cross-references)
    VAULT -.-> EXTRA
    VAULT -.-> HOST_VARS
    VAULT -.-> GROUP_SPECIFIC
    VAULT -.-> GROUP_ALL
    VAULT -.-> INV_HOST
    VAULT -.-> INV_GROUP

    %% 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 highPriority
```

```
    class HOST_VARS,GROUP_SPECIFIC mediumPriority
    class GROUP_ALL,INV_HOST,INV_GROUP,ROLE_VARS mediumPriority
    class ROLE_DEFAULTS lowPriority
    class VAULT secretClass
```

🔄 Environment-Specific Configuration Patterns

```
graph TB
    subgraph "Configuration Management Pattern"
        subgraph "Base Configuration"
            BASE_ALL[group_vars/all.yml<br/>🌍 Global Defaults<br/>Common to all environments<br/>Security, monitoring, backup settings]

            BASE_GROUPS[Group Variables<br/>📁 group_vars/<service>.yml<br/>Service-specific defaults<br/>PostgreSQL, Redis, Nginx configs]
        end

        subgraph "Environment Overrides"
            ENV_DEV[Development<br/>🔧 inventories/dev/<br/>Debug enabled<br/>Relaxed security<br/>Test data enabled]

            ENV_TEST[Test<br/>🧪 inventories/test/<br/>Validation enabled<br/>Performance testing<br/>Minimal resources]

            ENV_PROD[Production<br/>🏭 inventories/prod/<br/>Security hardened<br/>High availability<br/>Performance optimized]
        end

        subgraph "Host-Specific Customization"
            HOST_CUSTOM[Host Variables<br/>📁 host_vars/<hostname>.yml<br/>Individual server settings<br/>Resource allocation<br/>Special configurations]
        end

        subgraph "Runtime Flexibility"
            RUNTIME[Runtime Variables<br/>🚀 Extra vars (-e)<br/>Deployment-specific<br/>Feature flags<br/>Emergency overrides]
        end
    end

    %% Configuration Flow
    BASE_ALL --> ENV_DEV
    BASE_ALL --> ENV_TEST
    BASE_ALL --> ENV_PROD

    BASE_GROUPS --> ENV_DEV
    BASE_GROUPS --> ENV_TEST
    BASE_GROUPS --> ENV_PROD

    ENV_DEV --> HOST_CUSTOM
    ENV_TEST --> HOST_CUSTOM
    ENV_PROD --> HOST_CUSTOM

    HOST_CUSTOM --> RUNTIME

    %% Example Overrides
    ENV_DEV -.-> |"debug_mode: true<br/>ssl_verification: false<br/>log_level: DEBUG"| HOST_CUSTOM
    ENV_TEST -.-> |"test_mode: true<br/>performance_testing: true<br/>cleanup_enabled: true"| HOST_CUSTOM
    ENV_PROD -.-> |"security_hardening: true<br/>high_availability: true<br/>monitoring_enhanced: true"| HOST_CUSTOM

    %% Styling
    classDef baseClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef envClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
    classDef customClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
    classDef runtimeClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px

    class BASE_ALL,BASE_GROUPS baseClass
```

```
    class ENV_DEV,ENV_TEST,ENV_PROD envClass
    class HOST_CUSTOM customClass
    class RUNTIME runtimeClass
```

🎯 Deployment Target Groups

```
graph TB
    subgraph "Deployment Target Groupings"
        subgraph "Functional Groups"
            INFRA_GROUP[Infrastructure Services<br/>🏢 Domain Controllers<br/>🔐 Certificate Authorities<br/>Windows-based services]

            AI_GROUP[AI/ML Services<br/>🤖 LiteLLM Gateway<br/>🧠 Ollama Inference<br/>📦 Model Storage<br/>GPU-enabled hosts]

            UI_GROUP[UI Services<br/>💬 Web Interface<br/>⚖️ Load Balancers<br/>🌐 Frontend services]

            OPS_GROUP[Operations Services<br/>🗄️ PostgreSQL Cluster<br/>⚡ Redis Cache<br/>📊 Monitoring Stack<br/>📝 Logging Stack]
        end

        subgraph "Platform Groups"
            LINUX_HOSTS[Linux Hosts<br/>🐧 Ubuntu/RHEL<br/>Container-ready<br/>Most AI/ML and web services]

            WINDOWS_HOSTS[Windows Hosts<br/>🪟 Windows Server<br/>Active Directory<br/>Certificate Services]
        end

        subgraph "Role-Based Groups"
            PRIMARY[Primary Services<br/>🎯 Master nodes<br/>Main service instances<br/>High priority]

            SECONDARY[Secondary Services<br/>🔄 Replica nodes<br/>Backup instances<br/>Failover targets]

            GPU_HOSTS[GPU-Enabled Hosts<br/>🎮 NVIDIA GPUs<br/>CUDA support<br/>AI/ML inference]
        end

        subgraph "Availability Groups"
            AZ1[Availability Zone 1<br/>🏢 Primary datacenter<br/>Main service instances<br/>Active workloads]

            AZ2[Availability Zone 2<br/>🏢 Secondary datacenter<br/>Replica instances<br/>Disaster recovery]
        end

        subgraph "Maintenance Groups"
            CRITICAL[Critical Services<br/>🚨 Zero-downtime required<br/>Database masters<br/>Load balancers]

            STANDARD[Standard Services<br/>⚙️ Planned maintenance windows<br/>Application servers<br/>Monitoring services]
        end
    end

    %% Group Relationships
    INFRA_GROUP --> WINDOWS_HOSTS
    AI_GROUP --> LINUX_HOSTS
    AI_GROUP --> GPU_HOSTS
    UI_GROUP --> LINUX_HOSTS
    OPS_GROUP --> LINUX_HOSTS

    PRIMARY --> CRITICAL
    SECONDARY --> STANDARD
```

```
    AZ1 --> PRIMARY
    AZ2 --> SECONDARY

    %% Styling
    classDef functionalClass fill:#e3f2fd,stroke:#1976d2,stroke-width:2px
    classDef platformClass fill:#f3e5f5,stroke:#7b1fa2,stroke-width:2px
    classDef roleClass fill:#e8f5e8,stroke:#388e3c,stroke-width:2px
    classDef availabilityClass fill:#fff3e0,stroke:#f57c00,stroke-width:2px
    classDef maintenanceClass fill:#ffebee,stroke:#d32f2f,stroke-width:2px

    class INFRA_GROUP,AI_GROUP,UI_GROUP,OPS_GROUP functionalClass
    class LINUX_HOSTS,WINDOWS_HOSTS platformClass
    class PRIMARY,SECONDARY,GPU_HOSTS roleClass
    class AZ1,AZ2 availabilityClass
    class CRITICAL,STANDARD maintenanceClass
```

📋 Inventory Usage Examples

Basic Deployment Commands

```
# Deploy entire development environment
ansible-playbook -i inventories/dev/hosts.yml playbooks/site/main.yml

# Deploy only AI/ML services in production
ansible-playbook -i inventories/prod/hosts.yml playbooks/services/ai-ml.yml --limit ai_
ml

# Deploy to specific host group
ansible-playbook -i inventories/prod/hosts.yml playbooks/services/database.yml --limit
databases

# Deploy with environment-specific overrides
ansible-playbook -i inventories/prod/hosts.yml playbooks/site/main.yml -e "environ-
ment=production" -e "debug_mode=false"

# Deploy to primary services only
ansible-playbook -i inventories/prod/hosts.yml playbooks/maintenance/update.yml --lim-
it primary_services

# Deploy to specific availability zone
ansible-playbook -i inventories/prod/hosts.yml playbooks/services/monitoring.yml --lim-
it availability_zone_1
```

Maintenance Operations

```
# Update only GPU-enabled hosts
ansible-playbook -i inventories/prod/hosts.yml playbooks/maintenance/gpu-update.yml --
limit gpu_enabled_hosts

# Backup all databases
ansible-playbook -i inventories/prod/hosts.yml playbooks/maintenance/backup.yml --lim-
it databases

# Restart secondary services (safe maintenance)
ansible-playbook -i inventories/prod/hosts.yml playbooks/maintenance/restart.yml --lim-
it secondary_services

# Check health of critical services
ansible-playbook -i inventories/prod/hosts.yml playbooks/monitoring/health-check.yml --
limit critical_services
```

Environment-Specific Operations

```
# Development environment with debug enabled
ansible-playbook -i inventories/dev/hosts.yml playbooks/site/main.yml -e "de-
bug_mode=true" -e "log_level=DEBUG"

# Test environment with performance testing
ansible-playbook -i inventories/test/hosts.yml playbooks/testing/performance.yml -e "lo
ad_testing=true"

# Production deployment with security hardening
ansible-playbook -i inventories/prod/hosts.yml playbooks/site/main.yml -e "secur-
ity_hardening=true" -e "high_availability=true"
```

This comprehensive inventory documentation provides clear visual representations of the HX Infrastructure organization, making it easy to understand the relationships between environments, services, and configuration management patterns.