# Environments & Configuration Matrix

## Purpose and scope

This document defines the environments and configuration settings for the PineCone Pro ERP/IMS system throughout the software development lifecycle. Proper environment management ensures control, stability, security and reliability【984997393224141†L83-L96】 while allowing teams to work concurrently without risk to live operations【984997393224141†L83-L96】. It also provides a foundation for developers, QA and operations personnel to collaborate seamlessly【251501613233846†L83-L89】. The matrix outlines the differences in infrastructure, data, integrations, security settings, and operational parameters across Development, Test/QA, User Acceptance/Staging and Production environments. It adheres to best practices for environment separation and management【984997393224141†L102-L112】, ensuring that each environment serves its distinct purpose and that configuration changes are controlled and auditable.

## Environment definitions

| Environment | Purpose | Data characteristics | Access | Separation method |
| --- | --- | --- | --- | --- |
| **Development (Dev)** | Sandbox for developers to build, test and iterate new features; experimentation and rapid innovation【984997393224141†L88-L96】. | Synthetic or mocked data【984997393224141†L147-L149】; no production PII. | Developers and automation pipelines. | Logical separation using containers or virtualization to isolate dependencies and mimic production setup【251501613233846†L120-L124】. |
| **Quality Assurance (QA/Test)** | Controlled environment where QA engineers perform functional, integration, regression and performance testing. Should mirror production closely to catch issues early【251501613233846†L130-L134】. | Sanitized production data or realistic test datasets【984997393224141†L147-L149】. | QA team, automated test suites, developers for debugging. | Logical separation using dedicated Kubernetes namespaces and databases; functional separation via role‑based access controls【984997393224141†L102-L112】. |
| **User Acceptance Testing (UAT)/Staging** | Pre‑production environment used by stakeholders and end‑users to validate new features in a production‑like setting without impacting live operations【984997393224141†L91-L94】. | Near‑production configuration with sanitized production data; may include anonymized user data. | Business stakeholders, product owners, select customers; release managers. | Logical or physical separation; usually separate AKS cluster or namespace with same topology as production【984997393224141†L102-L112】. |
| **Production (Prod)** | Live environment where systems are customer‑facing and stability is non‑negotiable【984997393224141†L94-L96】. Must meet high availability and performance requirements. | Live customer data and financial data; full regulatory compliance. | End‑users, operations team, limited access for developers under strict change control. | Often physically separated network segments or dedicated clusters; strict functional separation enforced with policies and role‑based access control【984997393224141†L102-L112】. |

## Configuration management guidelines

1. **Externalised configuration**: The application code must not contain environment‑specific values. Use environment variables, config maps (Kubernetes) or secret management systems to inject configuration at runtime. This follows the principle that code should be environment agnostic, and the environment tells the code what it needs to run【984997393224141†L147-L149】.
2. **Version control for configuration**: Store configuration templates in a version control system and manage changes via pull requests and code reviews. Use separate files or branches for each environment; maintain a master template with all keys and document required values. Using version control ensures consistency and facilitates rollback【251501613233846†L120-L124】.
3. **Isolation and replicability**: Each environment should run in isolated containers or namespaces to prevent interference. Use infrastructure‑as‑code tools to provision environments consistently. Isolation and replicability are fundamental for Dev environments【251501613233846†L120-L124】 and apply to all environments.
4. **Environment data**: Use synthetic or mocked data in development; sanitized production data in QA and UAT; live data only in production【984997393224141†L147-L149】. Access to production data is restricted to authorised roles, and sensitive information must be masked in non‑production environments.
5. **Documentation and maintenance**: Maintain comprehensive documentation for each environment’s configuration and dependencies. Ongoing maintenance keeps environments up to date with patches and evolving requirements【251501613233846†L92-L104】.
6. **Separation of duties**: Enforce functional separation through role-based access controls, ensuring that changes in one environment do not inadvertently affect others【984997393224141†L102-L112】. Project managers or release managers oversee promotion of code and configuration between environments and maintain change approval gates【984997393224141†L123-L129】.

## Configuration matrix

The following tables summarise key configuration settings for each environment. Secret values (e.g. passwords, API keys) are referenced symbolically; they should reside in secure secret stores (e.g. Azure Key Vault) and injected at runtime.

### 1. Core infrastructure

| Setting | Development | QA/Test | UAT/Staging | Production |
| --- | --- | --- | --- | --- |
| **Kubernetes cluster** | Local Docker Desktop or **Dev** AKS cluster; single node or low‑cost cluster; auto‑scaling disabled. | Shared **QA** AKS cluster or dedicated namespace; moderate auto‑scaling settings. | Dedicated AKS cluster or namespace replicating production scaling; auto‑scaling enabled. | Separate **Production** AKS cluster in Azure; multiple availability zones; auto‑scaling and surge capacity. |
| **Namespace naming** | erp-dev-<team> | erp-qa | erp-uat | erp-prod |
| **Container registry** | Azure Container Registry (ACR) – dev repository; includes feature branches and test tags. | ACR – staging repository; images built from develop branch. | ACR – staging repository; images built from release branch. | ACR – production repository; images built from tagged releases; only signed images allowed. |
| **Ingress / hostnames** | Localhost or dev.pinecone.local domains; self‑signed TLS certificates. | qa.pineconepro.com; wildcard subdomain; certificates from internal CA. | uat.pineconepro.com; TLS certificates via ACME staging authority. | api.pineconepro.com, app.pineconepro.com; TLS certificates via public CA (e.g. Let’s Encrypt or Azure App Gateway). |
| **Scaling parameters** | Horizontal Pod Autoscaler (HPA) disabled or set to min=1, max=2; CPU limit 0.5–1 vCPU; memory 512–1024 MB. | HPA enabled with min=2, max=4 pods; resource limits 1–2 vCPU; memory 1–2 GB. | HPA enabled with min=2, max=6 pods; resources similar to production but lower concurrency. | HPA enabled; min=4, max=10 pods per service; CPU and memory limits based on load testing; Pod Disruption Budgets ensure high availability. |
| **Logging & monitoring** | Debug level logging; send logs to local files or dev logging service; minimal monitoring. | Info level logging; centralised logging via Azure Monitor or ELK stack; APM instrumentation enabled. | Info level logging; same monitoring stack as production; synthetic transactions to test alerts. | Warning/Error level logging; logs shipped to central log analytics; comprehensive APM (Application Insights), metrics and alerting. |
| **Secrets and config** | Local .env files or dev Key Vault; secrets may be dummy values; rotated infrequently. | QA Key Vault; secrets include sanitized tokens; rotated regularly; limited access. | UAT Key Vault; includes staging API keys for external services; rotated according to policy. | Production Key Vault; real API keys and credentials; rotated per compliance; strict access policies. |
| **CI/CD pipeline** | Triggered on feature branches; builds images, runs unit tests and deploys to Dev environment. | Triggered on merges to develop branch; executes integration tests; deploys to QA. | Triggered on merges to release branch; executes smoke tests; deploys to UAT; requires release manager approval to promote to production. | Triggered on tagged releases; executes end‑to‑end tests; manual approval gates; deploys to production using blue/green or canary strategy. |

### 2. Database & messaging configuration

| Component | Development | QA/Test | UAT/Staging | Production |
| --- | --- | --- | --- | --- |
| **Database engine** | Azure SQL Development edition or local SQL Server Express; small compute tier; restore from synthetic data sets. | Azure SQL Standard tier; restored from sanitized production snapshot; performance similar to production. | Azure SQL Standard tier; near‑production sizing; sanitized data; replicate indexes and statistics. | Azure SQL Production tier with high availability (HA) replicas; geo‑replication for disaster recovery; encrypted at rest and in transit. |
| **Connection strings** | Server=dev-sql.local, Database=ERPDev; User Id=erpdev; Password=$DEV\_DB\_PASS | Server=qa-sql.pineconepro.com; Database=ERPQA; User Id=erpqa; Password=$QA\_DB\_PASS | Server=uat-sql.pineconepro.com; Database=ERPUAT; User Id=erpuat; Password=$UAT\_DB\_PASS | Server=prod-sql.pineconepro.com; Database=ERPProd; User Id=erpprod; Password=$PROD\_DB\_PASS |
| **Message broker** | Azure Service Bus (dev namespace); partitioning disabled; low throughput units. | Azure Service Bus (qa namespace); moderate throughput units; duplicate detection enabled. | Azure Service Bus (uat namespace); throughput similar to production; advanced features (dead‑lettering, sessions) configured. | Azure Service Bus (prod namespace); high throughput units; geo‑disaster recovery; access controlled via managed identities. |
| **Event storage** | In‑memory event store or dev Cosmos DB; event retention 7 days. | Cosmos DB (qa account); retention 30 days for debugging. | Cosmos DB (uat account); retention 30 days. | Cosmos DB (prod account); retention 90 days for auditing; throughput configured for high volume. |
| **Cache** | Redis cache (dev instance); minimal size; disabled in local environment. | Redis cache (qa instance); replicates production TTL and eviction policies. | Redis cache (uat instance); similar configuration to production; used to test failover. | Redis cache (prod instance); clustered with replicas; TLS enabled; dynamic scaling. |

### 3. External services & integrations

| Service | Development | QA/Test | UAT/Staging | Production |
| --- | --- | --- | --- | --- |
| **Web store API** | Endpoint pointing to dev version of web store (e.g. https://dev-store.pineconepro.com/api); use sandbox credentials. | Endpoint pointing to QA instance of web store; sanitized data. | Endpoint pointing to UAT web store; features flagged off for untested features. | Endpoint pointing to live web store; high throughput; integrated payment gateway. |
| **Amazon marketplace** | Use Amazon Seller Central sandbox; test credentials and products. | Use Amazon staging account; test orders; restrictions on creating shipments. | Use Amazon pre‑prod environment; replicate product catalog; test order processing end‑to‑end. | Use Amazon production account; real SKUs; handle FBA/FBM orders. |
| **3PL integration** | Simulated 3PL API or dev 3PL sandbox; send test ASNs and receipts. | Connect to 3PL staging environment; test inbound/outbound flows; sanitized PO numbers. | Connect to 3PL staging or partner UAT environment; validate cross‑dock workflows. | Connect to 3PL production API; handle actual shipments; monitor SLA metrics. |
| **Payment gateway** | Use payment gateway sandbox (e.g. Stripe test mode); test cards; no actual charges. | Use payment gateway test environment; test fraud rules; test refunds. | Use payment gateway pre‑prod environment; test 3D Secure flows; tokenization. | Use payment gateway live environment; enable real payments; ensure PCI DSS compliance. |
| **Tax calculation service** | Use TaxJar sandbox; limited jurisdictions; test API keys. | Use TaxJar staging environment; same features as production but lower rate limits. | Use TaxJar UAT environment; replicate production load; ensure rate limit threshold. | Use TaxJar production API; high accuracy and uptime【118742738972405†L9-L20】; monitor consumption and throttling. |
| **Accounting system** | Connect to accounting software sandbox (QuickBooks/NetSuite); test ledger mappings; no financial impact. | Use accounting system QA tenant; test AP/AR sync; sanitized vendor/customer names. | Use accounting system UAT tenant; confirm reconciliation; test error handling. | Use accounting system production tenant; perform live journal postings; ensure segregation of duties. |
| **Email/SMS services** | Use sandbox or mock email service (e.g. Mailtrap) to capture outbound emails; test Twilio sandbox for SMS. | Use staging email/SMS environment; ensure deliverability to test inboxes/phones. | Use near‑production email/SMS environment; send communications to internal testers. | Use live email/SMS providers; abide by opt‑out rules and data privacy policies. |
| **Hazmat compliance system** | Use internal mock to test label generation and packaging instructions; no regulatory submission. | Connect to staging version of hazmat compliance provider; test documentation workflows. | Connect to UAT compliance system; ensure packaging guidelines are correct. | Connect to production compliance system (e.g. UPS HazMat module); ensure all hazmat shipments meet DOT and IATA regulations【79366541072420†L13-L27】【79366541072420†L73-L86】. |

### 4. Application configuration

| Configuration category | Development | QA/Test | UAT/Staging | Production |
| --- | --- | --- | --- | --- |
| **Logging level** | DEBUG for verbose output; console logging; optional file logging. | INFO; send structured logs to log aggregation with environment tag. | INFO/WARN; logs aggregated; include correlation IDs. | WARN/ERROR; logs aggregated and retained per policy; sensitive fields masked. |
| **Feature toggles** | Many experimental features enabled by default; toggles controlled via environment variables or config maps. | Only features ready for testing; toggles mirror production but can be enabled for QA. | Production features with controlled enablement; final toggling before go‑live. | Only stable features; toggles changed via change management; dynamic toggles for A/B tests or phased rollouts. |
| **Security settings** | Relaxed security to speed development (e.g. self‑signed certificates, simplified authentication flows). | Secure defaults; test identity provider integration; enforce RBAC. | Same security configuration as production; test MFA flows; data encryption in transit. | Full security hardening: TLS enforced; RBAC per module; secrets management; audit logging; compliance controls. |
| **Tracing and telemetry** | Disabled or minimal instrumentation. | Enabled with sampling; evaluate performance overhead. | Same instrumentation as production; test integration with observability tools. | Full tracing with adaptive sampling; monitor dependencies; create alerts for anomalies. |
| **Error handling** | Developer‑friendly error pages; verbose stack traces. | Generic error pages; capture errors and send to bug tracking; no sensitive info. | Production error handling; user‑friendly messages; error IDs for support. | Same as UAT; plus integration with incident management (PagerDuty, Opsgenie). |
| **Rate limits and throttling** | Disabled or generous limits; accelerate development. | Throttling configured similar to production; test systems’ resilience. | Rate limits reflecting expected production traffic; adjust for pre‑launch tests. | Strict rate limits and quotas; align with SLAs; backpressure mechanisms across services. |

### 5. Data & compliance

| Aspect | Development | QA/Test | UAT/Staging | Production |
| --- | --- | --- | --- | --- |
| **Data source** | Synthetic or randomly generated data; no PII or financial data【984997393224141†L147-L149】. | Sanitized production data; anonymized IDs; replicates size and shape of production. | Sanitized production data with additional test cases; may include anonymised transaction histories. | Live production data; PII stored encrypted; must meet all regulatory requirements (SOX, GDPR, CCPA). |
| **Backups & retention** | Optional backups; used for developer convenience; short retention (e.g. 7 days). | Regular backups; 14–30 days retention; used for debugging test failures. | Regular backups; 30 days retention; test restore procedures. | Full backups; retention per compliance (e.g. 7 years); cross‑region replication; tested recovery objectives (RPO, RTO). |
| **Compliance & regulatory** | No compliance requirements; but adhere to internal security guidelines. | Simulate compliance controls; test auditing and logging. | Mirror production compliance settings; test regulatory reporting. | Full compliance (PCI DSS, SOX, tax regulations, hazmat rules); regular audits; encryption at rest and in transit; data masking and pseudonymisation for internal access. |

## Environment promotion & change control

Changes flow from Dev → QA/Test → UAT → Production. Promotion requires:

1. **Automated pipelines**: CI/CD pipelines automatically build, test and deploy across environments. Each stage requires passing tests and manual approval gates in accordance with the release policy.
2. **Configuration promotion**: Use templated configuration files; update environment variables in a secure store. Configuration changes must be reviewed and version controlled.
3. **Data migration**: Apply database migrations via migration scripts; migrations are tested in QA and UAT before running in production.
4. **Change approval**: Project managers and change advisory board review and approve releases. Approval gates ensure governance, change control and risk management【984997393224141†L123-L136】.
5. **Rollback and disaster recovery**: Maintain rollback plans and disaster recovery procedures; test them regularly in non‑production environments【984997393224141†L132-L136】. Production environment must have defined RPO and RTO as per the Non‑Functional Requirements document.

## Benefits of environment separation

Implementing clear environment separation provides increased stability, improved quality, enhanced security and streamlined workflows【984997393224141†L163-L169】. By ring‑fencing work in each environment and tailoring configurations and data sets, PineCone Pro can accelerate delivery while minimizing risks and ensuring regulatory compliance.