Change Control Policy and Procedure

vStream Digital Media

Last updated 03/02/25

Definitions

Term	Definition
Company	means vStream Digital Media
ShineVR	means the ShineVR product developed and operated by vStream Digital Media
GDPR	means the General Data Protection Regulation
Responsible Person	means Andrés Pitt, CTO
Change	Any modification to IT infrastructure, applications, configurations, or data structures
Standard Change	Pre-approved, low-risk change following documented procedure
Normal Change	Change requiring evaluation and approval before implementation
Emergency Change	Urgent change required to resolve critical incident or security vulnerability
IT Resources	Computing, networking, communications, application, telecommunications systems, infrastructure, hardware, software, data, databases, and related materials
Infrastructure-as-Code (IaC)	Managing infrastructure through machine-readable definition files rather than manual processes
Continuous Integration/Continuous Deployment (CI/CD)	Automated pipeline for testing and deploying code changes

1. POLICY STATEMENT

vStream Digital Media is committed to managing all changes to Company IT resources through a systematic and controlled process to minimise service disruption, maintain security, and ensure operational reliability. All changes to ShineVR applications, infrastructure, configurations, and data structures must follow established change control procedures.

The Company leverages Infrastructure-as-Code, automated testing, and continuous integration/continuous deployment (CI/CD) pipelines to ensure changes are thoroughly tested, reviewed, and approved before deployment to production environments. All changes are logged, auditable, and reversible to support rapid recovery if issues arise.

2. PURPOSE

The purpose of this policy is to:

- Ensure all changes to IT resources are properly authorised, planned, and executed
- Minimise negative impact on services and users
- Prevent unauthorised or untested changes from affecting production systems
- Maintain comprehensive audit trails of all changes
- Enable rapid rollback of problematic changes
- Ensure changes comply with security and regulatory requirements
- Support business continuity through controlled change management
- Balance speed of innovation with operational stability
- Document lessons learned to improve change processes

3. SCOPE

This policy applies to:

All changes to Company IT resources including:

- ShineVR application code and configurations
- Google Cloud Platform infrastructure (compute, storage, networking)
- Database schemas and configurations
- Container images and deployments
- IAM policies and access controls
- Network configurations and firewall rules
- Backup and recovery procedures
- Monitoring and alerting configurations
- Third-party service integrations
- Documentation and procedures

All environments:

- Development environment
- Test/staging environment
- Production environment

All personnel:

- Employees (permanent, temporary, contractors)
- Third-party service providers with system access
- Anyone with ability to modify Company IT resources

4. CHANGE CONTROL PRINCIPLES

4.1 Core Principles

Systematic Approach:

- All changes follow documented procedures
- Changes are planned, tested, reviewed, and approved
- Emergency procedures exist for urgent situations
- Changes are reversible where possible

Risk-Based Management:

- Change risk assessed before approval
- Higher-risk changes require more rigorous review
- Risk mitigation measures implemented
- Rollback plans prepared before deployment

Automation and Testing:

- Automated testing validates changes before deployment
- Infrastructure-as-Code ensures consistency
- CI/CD pipelines enforce quality gates
- Manual changes minimised to reduce errors

Auditability:

- All changes logged with full audit trails
- Change history tracked in version control
- Who, what, when, why documented for every change
- Audit trails maintained for compliance demonstration

4.2 "Release Early, Release Often" Philosophy

Company Practice:

- Frequent small changes preferred over large infrequent changes
- Smaller changes easier to test, review, and rollback
- Regular deployment practice keeps team proficient
- Frequent releases reduce risk of each individual change
- **System reset, build, deploy, and restore operations occur very frequently**
- This practice inherently tests change and recovery procedures

5. CHANGE CATEGORIES

5.1 Standard Changes

Definition: Pre-approved, low-risk changes following documented procedures

Characteristics:

- Well-understood and documented
- Low risk of service disruption
- Successfully performed many times before
- Automated or semi-automated procedures
- Pre-approved by CTO

Examples:

- Routine security patches for non-critical systems
- Standard user account provisioning
- Routine backup configuration updates
- Minor documentation updates
- Development environment changes
- Container image updates following standard process

Approval: Pre-approved, no additional approval needed for execution

Documentation: Logged in change tracking system, limited documentation required

5.2 Normal Changes

Definition: Changes requiring evaluation and approval before implementation

Characteristics:

- Standard business changes
- Moderate risk of service disruption
- May affect multiple users or systems
- Requires testing and validation
- Requires formal approval

Examples:

- New feature deployments to production
- Database schema changes
- Infrastructure configuration changes
- IAM policy modifications
- Integration with new third-party services
- Changes to backup or monitoring configurations

Approval Requirements:

- Development/Staging: Backend developer or Product Manager approval
- Production: CTO or Product Manager approval required
- Documentation: Full change documentation required

5.3 Emergency Changes

Definition: Urgent changes required to resolve critical incident or security vulnerability

Characteristics:

- Required to resolve P1 critical incident
- Critical security vulnerability requiring immediate patching
- Service restoration following outage
- May bypass standard approval process
- Requires expedited review and approval
- Post-implementation review mandatory

Examples:

- Zero-day security vulnerability patching
- Critical bug fix for production outage
- Emergency access revocation
- Incident response actions
- Data recovery operations

Approval Requirements:

- CTO verbal/phone approval acceptable during incident
- Formal documentation completed within 24 hours post-change
- Post-implementation review required within 72 hours
- Lessons learned documented

6. SOFTWARE DEVELOPMENT CHANGE CONTROL

6.1 Development Workflow

Version Control (Git):

- All code changes tracked in Git version control (GitHub/Bitbucket)
- Feature branches used for development work
- Main/production branches protected (no direct commits)
- All changes via pull requests/merge requests
- Full change history maintained indefinitely

Branch Strategy:

- **Main/Production branch:** Production-ready code only
- **Staging branch:** Pre-production testing
- **Development branch:** Integration branch for development
- **Feature branches:** Individual features or bug fixes

6.2 Code Review Requirements

Mandatory Code Review:

- **All code changes require code review before merging**
- **No self-approval:** Developer cannot approve their own code
- **Reviewer requirements:**
    - CTO (for production deployments)
    - Product Manager (for production deployments)
    - Senior developer (for development/staging)

Code Review Process:

1. Developer creates pull request/merge request
2. Automated tests run (400+ tests)
3. Code reviewer examines code changes:
    - Code quality and standards compliance
    - Security implications
    - Performance considerations
    - Test coverage adequacy
    - Documentation completeness
4. Reviewer approves or requests changes
5. Developer addresses feedback
6. Final approval and merge

Code Review Criteria:

- Code follows established coding standards
- Adequate test coverage (unit and integration tests)
- No security vulnerabilities introduced
- Performance impact acceptable
- Error handling appropriate
- Documentation updated
- Breaking changes minimised and documented

6.3 Automated Testing (400+ Tests as of January 2025)

Comprehensive Test Suite:

- **400+ automated tests** run on every code change
- Tests must pass before code can be deployed
- **Failed tests block deployment** - no exceptions

Test Types:

Unit Tests:

- Test individual functions and components
- Fast execution (milliseconds per test)
- High code coverage target (>80%)

- Mock external dependencies

Integration Tests:

- Test interactions between components
- Database integration tests
- API integration tests
- External service integration tests

Security Tests:

- **Authentication and authorisation tests**
- **Role-based access control (RBAC) tests:**
    - User role cannot access Admin functions
    - Manager role cannot access SuperAdmin functions
    - Resources cannot access beyond assigned permissions
- Input validation tests
- SQL injection prevention tests
- Cross-site scripting (XSS) prevention tests
- API security tests

End-to-End Tests:

- Simulate complete user workflows
- Test entire application stack
- Verify critical business processes
- Test across multiple environments

Vulnerability Tests:

- Docker container vulnerability scanning
- Dependency vulnerability scanning
- Security library checks
- Known vulnerability detection

Performance Tests:

- Load and stress testing
- Response time validation
- Resource utilisation checks
- Scalability verification

6.4 Continuous Integration/Continuous Deployment (CI/CD)

Automated Pipeline: Every code commit triggers automated pipeline:

Stage 1: Build (0-5 minutes)

- Code compiled/built
- Container images created
- Build artifacts generated

- Build success/failure logged

Stage 2: Test (5-60 minutes)

- **400+ automated tests executed**
- Unit tests run first (fast feedback)
- Integration tests run
- Security tests run
- End-to-end tests run
- **Any test failure stops pipeline**

Stage 3: Security Scanning (5-10 minutes)

- **Docker container vulnerability scanning**
- Dependency vulnerability scanning
- Static code analysis (if applicable)
- **Critical/high vulnerabilities block deployment**

Stage 4: Staging Deployment (Automatic)

- Successful builds deployed to staging automatically
- Staging environment mirrors production configuration
- Additional testing in staging environment
- Smoke tests verify basic functionality

Stage 5: Production Deployment (Manual Approval)

- **CTO or Product Manager approval required**
- Deployment scheduled during appropriate window
- Blue-green or rolling deployment strategy
- Monitoring during deployment
- Automatic rollback on critical errors

Pipeline Enforcement:

- **Pipeline cannot be bypassed for production deployments**
- Manual console changes prohibited (except emergency)
- All production changes through CI/CD pipeline
- Pipeline failures investigated and resolved

6.5 Deployment Approval

Production Deployment Requirements:

- All automated tests passed
- Security scans clean (no critical/high vulnerabilities)
- Code review approved by CTO or Product Manager
- Staging deployment successful and tested
- Change documentation completed
- Rollback plan prepared

Approval Process:

1. Developer requests production deployment
2. System verifies all prerequisites met
3. CTO or Product Manager reviews:
    - Test results
    - Security scan results
    - Change description and impact
    - Rollback plan
4. Approver approves or rejects deployment
5. If approved, deployment proceeds
6. Deployment monitored in real-time
7. Post-deployment validation performed

Deployment Sign-Off:

- Approver name and timestamp logged
- Deployment details recorded in change register
- Full audit trail maintained

7. INFRASTRUCTURE CHANGE CONTROL

7.1 Infrastructure-as-Code (IaC)

Mandatory Practice:

- **All infrastructure defined in code** (declarative configuration)
- Infrastructure code stored in version control (Git)
- Infrastructure changes follow same process as application code:
    - Pull request/merge request
    - Code review required
    - Automated testing where possible
    - Approval before deployment

Benefits:

- Consistent environments (dev, staging, production)
- Reproducible infrastructure
- Version control and audit trail
- Rapid rebuild capability
- Eliminates configuration drift

Infrastructure Components Managed as Code:

- Google Cloud Platform resources (projects, VPCs, subnets)
- Cloud Run services and configurations
- Cloud SQL databases and configurations
- Cloud Storage buckets and lifecycle policies
- IAM policies and role bindings
- Firewall rules and network configurations

- Monitoring and alerting configurations
- Load balancer configurations

7.2 Manual Infrastructure Changes

Strongly Discouraged: Manual changes via Google Cloud Console should be avoided except:

- Emergency incident response
- Initial exploration/prototyping (development environment only)
- Troubleshooting and diagnostics

If Manual Change Required:

1. Document reason for manual change
2. Obtain verbal approval from CTO
3. Make change with detailed documentation
4. Update Infrastructure-as-Code to reflect change
5. Validate IaC matches actual infrastructure
6. Formal approval and review within 24 hours

7.3 Google Cloud Platform Changes

IAM Policy Changes:

- All IAM policy changes via Infrastructure-as-Code
- Code review by CTO required
- Changes tested in non-production first
- Principle of least privilege enforced
- Changes logged in Cloud Audit Logs

Network Configuration Changes:

- Firewall rule changes via IaC
- Network topology changes reviewed by CTO
- Security impact assessed
- Changes tested in staging environment first

Database Configuration Changes:

- Cloud SQL configuration changes documented
- Performance impact assessed
- Backup taken before change
- Rollback procedure documented

8. DATA AND CONTENT CHANGE CONTROL

8.1 ShineVR Content and Data Changes

ShineVR Admin Functions:

- **Full audit trail for all program content and data changes**

- **System records who does what on any content or data changes**
- Content changes tracked with:
    - User ID of person making change
    - Timestamp of change
    - Description of change (before/after values)
    - Reason for change
- Audit log accessible to CTO and authorised personnel

Content Change Process:

1. Admin or SuperAdmin logs into ShineVR admin interface
2. Makes content or configuration change
3. Change automatically logged with user ID and timestamp
4. Change takes effect immediately (staging) or after approval (production)
5. Audit log entry created and immutable
6. Change can be reviewed in audit log

8.2 Database Schema Changes

Schema Change Procedure:

1. **Development:** Schema change developed and tested
2. **Version Control:** Schema migration scripts in version control
3. **Code Review:** Schema changes reviewed by CTO
4. **Testing:** Migration tested in development and staging
5. **Backup:** Production database backed up before migration
6. **Approval:** CTO approval required for production
7. **Deployment:** Migration applied with monitoring
8. **Validation:** Data integrity verified post-migration
9. **Rollback Ready:** Rollback script prepared and tested

Database Data Changes:

- Routine data changes via application interfaces (logged)
- Direct database modifications require CTO approval
- Data corrections documented with justification
- All direct queries logged in audit trail

8.3 Configuration Changes

Application Configuration:

- Configuration stored in Google Secret Manager or environment variables
- Configuration changes via Infrastructure-as-Code or admin interfaces
- Configuration changes tested in non-production first
- Sensitive configuration (API keys, passwords) never in source code

System Configuration:

- Operating system configurations managed via Docker base images
- Container configurations in version control

- Configuration changes trigger container rebuild and retest

9. CHANGE DOCUMENTATION

9.1 Change Request Documentation

Required for Normal Changes:

- **Change ID:** Unique identifier
- **Change Type:** Standard, Normal, or Emergency
- **Requestor:** Who is requesting the change
- **Change Description:** What is being changed and why
- **Systems Affected:** Which systems/applications impacted
- **Risk Assessment:** Potential risks and impact
- **Business Justification:** Why change is needed
- **Implementation Plan:** Step-by-step procedure
- **Rollback Plan:** How to reverse change if needed
- **Testing Performed:** What testing validates the change
- **Approval Required:** Who must approve
- **Implementation Window:** When change will occur
- **Expected Downtime:** Any service interruption expected

9.2 Change Tracking

Change Register:

- All changes logged in centralised change register
- Automated logging from CI/CD pipeline
- Manual changes logged by implementer
- Change register accessible to CTO and management

Tracked Information:

- Change ID and classification
- Date/time of change
- Implementer and approver
- Systems/environments affected
- Success or failure status
- Issues encountered
- Rollback performed (if any)
- Post-implementation review results

Audit Trail Requirements:

- Comprehensive audit trails maintained for all changes
- Trails include:
    - Version control commit history (Git)
    - CI/CD pipeline execution logs
    - Google Cloud Audit Logs (infrastructure changes)
    - ShineVR admin audit logs (content/data changes)

    - Change register entries
- Audit trails retained per data retention policy (minimum 1 year, up to 7 years for compliance)

10. CHANGE IMPLEMENTATION

10.1 Change Windows

Standard Change Windows:

- **Preferred window:** Outside business hours (evenings, weekends)
- **Business hours:** 9:00 AM - 5:30 PM Irish time, Monday-Thursday
- **Ideal times:**
    - Weekday evenings: 6:00 PM - 11:00 PM

Production Changes:

- Scheduled during low-usage periods when possible
- Customer communication for changes causing downtime
- Emergency changes: Implemented immediately as needed

Development/Staging Changes:

- Can occur any time (24/7)
- Should avoid disrupting team working hours when possible

10.2 Implementation Procedure

Pre-Implementation:

1. Verify all approvals obtained
2. Review implementation plan and rollback plan
3. Notify relevant stakeholders
4. Take backups if applicable
5. Prepare monitoring and alerting
6. Confirm rollback procedure tested

During Implementation:

1. Follow documented implementation plan
2. Monitor systems in real-time
3. Document any deviations from plan
4. Watch for errors or unexpected behaviour
5. Be ready to rollback if issues arise

Post-Implementation:

1. Verify change successful
2. Test affected functionality
3. Monitor for 24-48 hours for issues
4. Update documentation if needed

5. Close change request with status
6. Document lessons learned

10.3 Rollback Procedures

Rollback Triggers:

- Change causes critical errors or service outage
- Performance degradation exceeds acceptable thresholds
- Security vulnerability introduced
- Data integrity compromised
- Unforeseen negative impacts discovered

Rollback Process:

1. **Decision:** CTO or on-call engineer decides to rollback
2. **Notification:** Team notified of rollback decision
3. **Execution:** Rollback plan executed immediately
4. **Verification:** System functionality verified post-rollback
5. **Monitoring:** Enhanced monitoring for stability
6. **Investigation:** Root cause analysis of why rollback needed
7. **Documentation:** Rollback documented in change record

Rollback Capability:

- Container-based architecture enables rapid rollback (redeploy previous container)
- Database migrations include rollback scripts
- Infrastructure-as-Code enables infrastructure rollback
- Backups available for data rollback
- Version control enables code rollback

11. CHANGE REVIEW AND APPROVAL

11.1 Approval Requirements by Change Type

Change Type	Environment	Approver	Documentation	Testing Required
Standard	Any	Pre-approved	Minimal	Per standard procedure
Normal	Development	Backend Developer	Change ticket	Unit tests
Normal	Staging	Product Manager or CTO	Change ticket + test results	Full test suite
Normal	Production	CTO or Product Manager	Complete documentation	Full test suite + staging validation

Change Type	Environment	Approver	Documentation	Testing Required
Emergency	Production	CTO (verbal acceptable)	Completed within 24 hours	As time permits, full review post-implementation

11.2 Change Advisory Board (CAB)

For Company Size:

- Formal CAB not currently required given small team size
- CTO serves as primary approver and change manager
- Product Manager serves as secondary approver
- Senior leadership consulted for major changes

CAB Activation Criteria: If company grows significantly, CAB may be implemented for:

- Major infrastructure changes
- Changes affecting multiple systems/customers
- High-risk changes
- Changes during critical business periods

11.3 Approval Documentation

Documented Approvals:

- Approver name and role
- Approval date and time
- Approval method (written, verbal, electronic)
- Any conditions or concerns noted
- Approval logged in change register

Electronic Approvals:

- Pull request/merge request approvals (code changes)
- Change ticket approvals (infrastructure changes)
- Email approvals (for formal change requests)
- All electronic approvals timestamped and auditable

12. MONITORING AND VALIDATION

12.1 Change Monitoring

Real-Time Monitoring During Changes:

- Google Cloud Monitoring dashboards watched
- Application logs monitored for errors
- Performance metrics tracked
- User impact assessed

- Security alerts monitored

Post-Change Monitoring:

- Enhanced monitoring for 24-48 hours post-change
- Error rates and performance metrics compared to baseline
- User feedback monitored
- Any anomalies investigated immediately

12.2 Change Validation

Validation Checklist:

- Functionality testing confirms change works as intended
- Regression testing confirms existing functionality unaffected
- Performance testing confirms acceptable performance
- Security testing confirms no vulnerabilities introduced
- User acceptance testing (for feature changes)
- Documentation updated to reflect changes

12.3 Change Success Criteria

Successful Change Indicators:

- All planned functionality delivered
- No service disruption or within acceptable limits
- No security vulnerabilities introduced
- Performance within acceptable parameters
- No data loss or corruption
- User feedback positive or neutral
- Rollback not required

13. EMERGENCY CHANGES

13.1 Emergency Change Process

When to Use Emergency Process:

- P1 critical incident requiring immediate action
- Active security breach requiring immediate containment
- Critical vulnerability requiring immediate patching
- Data loss prevention requiring immediate action
- Service restoration following outage

Expedited Approval:

1. Incident responder assesses situation
2. CTO contacted immediately (phone call)
3. Verbal approval obtained if CTO available
4. If CTO unavailable, Product Manager or CEO contacted
5. Change implemented immediately

6. Formal documentation completed within 24 hours

Emergency Change Controls:

- Change still logged and documented (after the fact if necessary)
- Testing performed to extent possible
- Rollback plan prepared
- Monitoring enhanced
- Post-implementation review mandatory

13.2 Post-Emergency Review

Required Within 72 Hours:

- Full incident review conducted
- Change documentation completed
- Root cause analysis performed
- Effectiveness of emergency change assessed
- Process improvements identified
- Lessons learned documented
- Change formally approved retrospectively

14. FAILED CHANGES

14.1 Handling Failed Changes

When Change Fails:

1. **Immediate:** Assess impact and severity
2. **Containment:** Rollback change if causing outage or data loss
3. **Notification:** Notify CTO and stakeholders
4. **Investigation:** Determine root cause of failure
5. **Documentation:** Document failure and lessons learned
6. **Remediation:** Fix issues and retest
7. **Retry:** Submit revised change request

Failed Change Analysis:

- Why did change fail?
- Was testing adequate?
- Was the rollback plan effective?
- What could prevent similar failures?
- What process improvements are needed?

14.2 Lessons Learned

Continuous Improvement:

- Failed changes provide valuable learning
- Root causes analysed and addressed
- Testing procedures enhanced

- Documentation improved
- Team training updated
- Process refinements implemented

15. CHANGE CONTROL METRICS

15.1 Key Performance Indicators

Tracked Metrics:

- **Change success rate:** Percentage of changes deployed successfully without rollback
- **Change frequency:** Number of changes per week/month
- **Failed change rate:** Percentage of changes requiring rollback
- **Emergency change percentage:** Emergency changes as percentage of total
- **Mean time to implement:** Average time from approval to deployment
- **Change-related incidents:** Incidents caused by changes
- **Rollback frequency:** Number of rollbacks per period

Target Metrics:

- Change success rate: >95%
- Emergency changes: <5% of total changes
- Change-related incidents: <2% of changes
- Mean time to implement: <48 hours for normal changes

15.2 Reporting

Weekly Change Report:

- Changes deployed in past week
- Changes scheduled for next week
- Any failed changes and root causes
- Outstanding change requests

Monthly Change Summary:

- Change volume and trends
- Success rate and metrics
- Failed changes analysis
- Process improvements implemented

Quarterly Change Review:

- Comprehensive metrics analysis
- Trends and patterns identified
- Process effectiveness assessment
- Recommendations for improvement

16. ROLES AND RESPONSIBILITIES

Role	Responsibilities
CTO (Responsible Person)	Overall change control policy ownership; approve production changes; review infrastructure changes; incident response for change-related issues; change process improvement; monitor change metrics; emergency change approval authority
Product Manager	Approve staging and production deployments; prioritise feature changes; validate business requirements; participate in change review; approve normal changes to production
Backend Developers	Develop and test changes; create pull requests; implement code reviews; deploy to development/staging; follow change procedures; document changes; fix failed changes
DevOps/Infrastructure (if dedicated role)	Implement Infrastructure-as-Code changes; manage CI/CD pipeline; configure automated testing; monitor deployments; manage rollback procedures; infrastructure change implementation
All Employees	Follow change control procedures; report issues caused by changes; participate in testing when required; provide feedback on changes

17. TRAINING AND AWARENESS

17.1 Change Control Training

Required Training:

- **New developers:** Comprehensive change control procedures training
- **All technical staff:** Change control awareness and procedures
- **Managers:** Change approval procedures and criteria

Training Topics:

- Change control policy and procedures
- Version control and branching strategy
- Code review process and standards

- Automated testing requirements
- Infrastructure-as-Code practices
- Deployment and rollback procedures
- Emergency change procedures
- Audit trail requirements

17.2 Continuous Learning

- Regular team discussions on change processes
- Lessons learned from failed changes
- Process improvement suggestions welcomed
- Training materials updated based on lessons learned

18. COMPLIANCE AND AUDIT

18.1 Regulatory Compliance

GDPR Compliance:

- Changes to personal data processing require impact assessment
- Audit trails support accountability principle
- Data protection by design in change process

Industry Standards:

- ISO 27001 change control requirements addressed
- SOC 2 change management controls implemented
- Documented and auditable change process

18.2 Audit Support

Audit Evidence:

- Change register with all changes documented
- Version control commit history
- CI/CD pipeline execution logs
- Code review records
- Approval documentation
- Test results and reports
- Incident records for change-related issues

Audit Procedures:

- Quarterly internal change control audits
- Annual comprehensive review
- External audits supported with documentation
- Audit findings tracked and remediated

19. EXCEPTIONS

19.1 Exception Process

Exceptions to change control requirements may be requested for:

- True emergencies requiring immediate action
- One-time situations with unique circumstances
- Technical limitations preventing standard process

All exceptions must:

- Be documented with detailed justification
- Be approved by CTO in writing (or verbally for emergencies, documented within 24 hours)
- Document compensating controls
- Be time-limited
- Be reviewed in post-implementation review

19.2 Emergency Exceptions

- Emergency changes may bypass standard approval temporarily
- All emergency changes must be formally documented within 24 hours
- Post-implementation review mandatory within 72 hours
- Emergency process abuse is policy violation

20. POLICY REVIEW AND UPDATES

This policy will be reviewed:

- **Annually:** Comprehensive review by CTO
- **After major incidents:** Update based on lessons learned from change-related incidents
- **Process changes:** When CI/CD pipeline or tools change significantly
- **Organisational changes:** New team members, roles, or structure
- **Technology changes:** New platforms, languages, or frameworks

21. RELATED POLICIES

This policy should be read in conjunction with:

- Information Security Policy
- Access Management Policy
- Cloud Security Policy
- Backup and Recovery Policy
- Incident Response Plan
- Data Protection Policy

22. CONTACT INFORMATION

For questions regarding this policy or to report change control issues:

Data Protection Officer / CTO: Andrés Pitt Email: andres@vstream.ie Phone: (086) 788 6570