Change Management Policy

1. Purpose

This policy establishes the process for managing changes to the Subcontext platform, infrastructure, and configuration. Its aim is to make sure every change to production is deliberate, tested, approved and reversible, so that customers get improvements without avoidable disruption or new security risk.

2. Scope

This policy applies to changes to:

- Application code, including the web application, REST API, MCP server, embeddable widgets and mobile apps
- Database schema and data migrations
- Infrastructure defined in code, including networking, firewall rules, scaling settings and alarms
- Secrets, credentials and encryption keys held for production
- Third-party dependencies
- Platform-wide configuration that affects all customers

Changes customers make to their own agents, journeys and settings within the product are outside this policy. They are governed by the product's own controls, including testing agents before they go live.

3. Definitions

Term	Meaning
Change	Any addition, modification or removal that could affect the production service
Release	A versioned container image built from the code and promoted to production
Rollback	Returning production to the previous known-good release
Change owner	The person who makes the change and is accountable for it until it is safely in production

4. Roles and Responsibilities

Role	Responsibility
Change owner	Prepares, tests and documents the change, and monitors it after release
Reviewer	Reviews the change for correctness, security and data safety
Founding team	Approves significant changes and production releases, and reviews emergency changes
Security Officer	Owns this policy and checks that it is followed

5. Change Categories

Category	Description	Approval Required
Standard	Routine deployments, dependency updates, minor feature additions	Peer review
Significant	New features, architecture changes, database migrations, security changes	Peer review + founding team approval
Emergency	Critical security patches, incident remediation, urgent production fixes	Post-deployment review

If a change could reasonably fall into two categories, it is treated as the higher one.

6. Change Process

6.1 Development

- All changes are developed in isolated branches
- Code is version-controlled in Git
- Automated tests must pass before merging
- Security-relevant changes are flagged for additional review

Infrastructure changes are made in the version-controlled infrastructure code, not by hand in the AWS console, so that every change is recorded and can be repeated or reversed. If a manual change is ever needed to resolve an incident, it is brought back into the code afterwards.

6.2 Review

- All changes require peer code review before deployment
- Reviews assess functionality, security, performance, and maintainability
- Database migrations are reviewed for data safety and reversibility

6.3 Testing

- Automated test suites run on every change (unit, integration, system tests)
- Significant changes are tested in a staging environment before production deployment
- Security scanning (Brakeman, Bundler Audit) runs as part of the deployment pipeline

Before release, the full check suite is run. It covers code style, known-vulnerability checks on Ruby gems and browser JavaScript packages, static security analysis that fails on any warning, the full automated test suite with a 100% line and branch coverage requirement, browser tests of complete customer journeys, and tests for the embeddable widgets and mobile apps. A change must not be released while any of these checks fails.

6.4 Deployment

- Deployments to production are automated and repeatable
- Database migrations are applied as part of the deployment process
- Deployments are monitored for errors and performance regression

- Rollback procedures are documented and tested

In practice:

1. The release is built as an immutable, versioned container image, which is scanned for vulnerabilities when published.
2. A manual approval step in the deployment pipeline must be completed before the release reaches production.
3. Database migrations run as a separate task before the new release goes live. A failed migration stops the release, and production stays on the previous version.
4. The new release is rolled out gradually, with health checks confirming that new servers are healthy before old ones are retired.
5. The change owner watches error monitoring and infrastructure alarms after release.
6. If a problem is found, the previous release image is redeployed.

6.5 Emergency Changes

- Emergency changes may bypass the standard review process
- A post-deployment review must be completed within 24 hours
- The change, rationale, and review are documented

An emergency change still goes through the deployment pipeline, including its approval step, and the full check suite is run as far as time allows. Only a member of the founding team may authorise a change as an emergency. The post-deployment review confirms the change was necessary and correct, and identifies any follow-up work.

6.6 Worked example

A dependency update that fixes a High severity vulnerability in a library is a Standard change: it is made on a branch, reviewed, checked with the full suite and released. If that vulnerability were being actively exploited against the platform, it would become an Emergency change: released as quickly as safely possible under the Incident Response Plan, then reviewed within 24 hours.

7. Communication

- Material changes to the Service are communicated to customers in accordance with the Terms of Service
- Scheduled maintenance windows are communicated in advance where practicable

During a planned maintenance window, a holding page is shown to users while the work runs.

8. Records

Git history, the deployment pipeline and the container registry together provide a record of what changed, who changed it, who approved it and when it reached production. These records are kept to support audits and investigations.

9. Exceptions

Any departure from this policy other than the emergency route must be approved in advance by a director and recorded with its reason.

10. Monitoring Compliance

The Security Officer periodically samples recent releases to check that they were reviewed, passed the check suite and were approved, and that emergency changes were reviewed within 24 hours. Gaps are recorded and corrected.

11. Review

This policy is reviewed annually and updated as development practices evolve.

12. Related Documents

- Secure Development Policy
- Information Security Policy
- Vulnerability Management Policy
- Incident Response Plan
- Business Continuity & Disaster Recovery Policy

13. Document Control

Item	Detail
Owner	Security Officer
Approved by	The directors of Subcontext Ltd
Review	At least annually
Effective	October 2026

Subcontext Ltd, registered in England and Wales, Company No. 17019645. Registered office: 124 City Road, London, EC1V 2NX. The current version of this document is published at subcontext.com.