Software Release Plan

Structured Deployment Process

Release Information

Release Version	
Target Deployment Date	
Release Manager	
Engineering Lead	
Deployment Strategy	☐ Blue-Green ☐ Canary ☐ Rolling ☐ Feature Flag

1. Core Components

Document the foundation: version control strategy, build pipeline, and testing protocols.

Version Control & Branching Strategy

Primary Branch: [e.g., main, master]
Feature Branch Pattern: [e.g., feature/*, feat/*]
Hotfix Branch Pattern: [e.g., hotfix/*]
Merge Approval Requirements: [# of reviewers, CI passing]
Hotfix Bypass Procedures: [describe emergency process]

Build Pipeline Configuration

Build Triggers: [branch patterns, tags]
Artifact Storage: [location/registry]
Environment Promotion Rules: [Dev → Test → Staging → Prod]
Build Timeout: [minutes]

Testing Protocols

- Unit Test Coverage Threshold: [____%]
- Integration Test Suite: [specify]
- Performance Benchmarks: [response time, throughput]
- Manual Verification Steps: [list critical paths]

2. Release Timeline

Track progress through each phase from feature freeze to deployment.

Pre-Release Phase (2-4 weeks before deployment)

Task	Owner	Due Date	Status
Feature freeze & finalize feature list			
Draft release notes			
Risk assessment & dependency mapping			
Schedule stakeholder reviews			

Development Phase (1-2 weeks before deployment)

Task	Owner	Due Date	Status
Code completion deadline			
Integration testing complete			
Automated test suite passing			
Manual testing for UI changes			

Staging & Validation (3-5 days before deployment)

Task	Owner	Due Date	Status
Deploy to staging environment			
Full regression testing			
Performance testing			
User acceptance testing (UAT)			
Stakeholder approval obtained			

3. Deployment Strategy Details

Document your chosen deployment approach and configuration.

Strategy: [Select One]

☐ Blue-Green Deployment

Configuration:
- Blue Environment: [URL/identifier]
- Green Environment: [URL/identifier]
- Traffic Switch Method: [load balancer, DNS]
- Validation Period Before Switch: [duration]

☐ Canary Deployment

Rollout Plan:
- Phase 1: 2% of users for [duration]
- Phase 2: 25% of users for [duration]
- Phase 3: 75% of users for [duration]
- Phase 4: 100% rollout

Success Metrics to Monitor: [error rate, response time, user engagement]

☐ Feature Flag Strategy

Feature Flags in This Release:

Feature Name	Flag Key	Initial State	Expiration Date
		☐ On ☐ Off	

4. Environment Management

Environment	Purpose	URL/Endpoint	Refresh Schedule
Development	Isolated testing by developers		After each test run
Testing	Integration & load testing		After each test run
Staging	Production-like validation		Before each release
Production	Live customer environment		N/A

Production Access Controls

Authorized Personnel: [list roles/names]
Change Approval Process: [describe ticket/approval workflow]
Emergency Access Procedures: [describe]

5. Risk Management & Rollback

70% of production failures result from changes. Document dependencies and define clear rollback triggers.

Dependency Mapping

Dependency	Type	Fallback Option
[External service/API]	[External Service / Database / API]	[Describe fallback]

Rollback Procedures

Rollback Method: [describe one-click vs. manual process]
Previous Version Storage: [location/retention policy]
Database Rollback Strategy: [backup/migration reversal]
Rollback Testing Date: [when tested in staging]

Rollback Triggers (Immediate Action Required)

- ☐ Response time degradation >20%
- ☐ Error rate >5%
- ☐ Critical feature failure
- ☐ Security vulnerability discovered
- ☐ Other: [specify]

Rollback Decision Authority

[Name/role authorized to trigger rollback without waiting for consensus]

6. Communication Protocols

Incident Communication Channels

Audience	Channel	Information Type
Engineering Team	Slack #incidents	Technical details, logs, metrics
Support Team	Email + status page	Customer-facing explanations
Executives	Email summary	Impact summary, resolution timeline

Pre-Written Communication Templates

[Link to or attach template documents for: Deployment Notification, Incident Alert, Resolution Update, Post-Mortem Summary]

7. Success Metrics & KPIs

Track these metrics to measure deployment efficiency and identify process bottlenecks.

Metric	Current Value	Target Value
Deployment Frequency		[e.g., daily, weekly]
Lead Time for Changes		[e.g., <1 day]
Mean Time to Recovery (MTTR)		[e.g., <1 hour]
Change Failure Rate		[e.g., <15%]

8. Go/No-Go Deployment Checklist

All items must be checked before proceeding with production deployment.

Pre-Deployment Requirements

- ☐ All automated tests passing (unit, integration, e2e)
- ☐ Code coverage meets threshold (____%)
- ☐ Staging environment tested successfully
- ☐ Performance benchmarks met
- ☐ Security scan completed with no critical issues
- ☐ Database migrations tested and validated
- ☐ Rollback procedure tested in staging
- ☐ Stakeholder approvals obtained
- ☐ Release notes finalized
- ☐ Monitoring and alerting configured
- ☐ Support team briefed on changes
- ☐ Customer communications prepared (if applicable)
- ☐ Deployment window scheduled (off-peak if possible)

Final Sign-Off

Release Manager: _____ Date: _____
Engineering Lead: _____ Date: _____
Product Owner: _____ Date: _____

9. Post-Deployment Activities

Immediate Monitoring (First 24 Hours)

- ☐ Monitor error rates and response times
- ☐ Check support ticket volume
- ☐ Verify key features functioning as expected
- ☐ Review system logs for anomalies

Retrospective (Within 1 Week)

What went well:
[Document successes]

What could be improved:
[Document challenges]

Action items for next release:
[List improvements with owners and due dates]

This template should adapt to your organization's needs. Start simple, measure results, and refine based on learnings.