

CI/CD y quality gates

- Ejecute pruebas automatizadas en cada cambio significativo
- Fusiones de bloques en fallos críticos de calidad
- Mantenga las canalizaciones rápidas para preservar el flujo de desarrolladores

Referencias: <https://docs.github.com/en/actions> | <https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment>

Acciones de GitHub para control de calidad

- Definir flujos de trabajo como código en YAML
- Dividir trabajos por tipo de prueba y paralelizar
- Publicar informes/artefactos para mayor visibilidad

Referencias: <https://docs.github.com/en/actions>

Estrategia de ramificación e integración

- Prefiere sucursales de corta duración y fusiones frecuentes.
- Usa prácticas basadas en trunk-based para reducir el riesgo de merge
- Mantener la rama principal liberable

Referencias: <https://trunkbaseddevelopment.com/>

Libere la confianza y el control de versiones

- Usa versiones semánticas para señales de cambio claras
- Vincula las decisiones de liberación a la evidencia, no a la intuición
- Seguimiento de la preparación para la reversión y los riesgos conocidos

Referencias: <https://semver.org/>

Mayús-Izquierda y Mayús-Derecha juntas

- Mayús-izquierda: previene defectos antes
- Shift-right: monitorear y validar en producción
- Usa ambos para obtener ciclos completos de retroalimentación de calidad

Referencias: <https://dora.dev/> | <https://opentelemetry.io/docs/>

