



Trabajo en Clase SCM

Fecha clase @21 de mayo de 2025

Integrantes: Ronny Cartagena, Maximiliano Madrid

Prof.: Adrián Eguez

Paralelo: GR1SW

Tema: SCM

Enunciado

Quiero que se haga una lista del proceso y herramientas de configuración del software con los siguientes puntos tomados en cuenta:

- Vamos a usar GIT
- Que herramientas para pipelines usariamos y que repositorio
- como seria el proceso de analisis de codigo (test, lint, etc)
- cuales serian las pautas para la revision de codigo
- como es el flujo de despliegue para el aplicativo (cuando se hace merge a la rama TAL entonces enviamos los cambios a AWS y desplegamos etc etc)

Liste el proceso y herramientas

Desarrollo

Proceso y herramientas para configuración del software

1. Uso de Git para control de versiones

El equipo utiliza **Git** como sistema de control de versiones para gestionar el código fuente. Cada desarrollador crea ramas feature basadas en la rama principal (**master**), donde desarrolla nuevas funcionalidades y realiza commits para registrar los cambios. El repositorio remoto está alojado en **GitHub**, lo que facilita la colaboración y gestión de cambios.

2. Herramientas para pipelines e integración continua

Para automatizar la integración y despliegue del software, se utiliza **Jenkins** como servidor de CI/CD. Jenkins está configurado con un webhook a través de **ngrok**, que detecta automáticamente los merges realizados en

la rama **master** del repositorio de GitHub. Al activarse, Jenkins ejecuta un pipeline que compila el proyecto, ejecuta pruebas y genera artefactos.

3. Proceso de análisis de código

El pipeline de Jenkins incluye varias etapas para asegurar la calidad del código:

- **Compilación del proyecto Java** para validar que no existan errores de sintaxis.
- **Ejecución de pruebas unitarias y de integración** utilizando frameworks como **JUnit5** y **Mockito**, incluyendo pruebas parametrizadas para verificar distintos escenarios.
- **Análisis estático del código** con herramientas como **Checkstyle** o integración con **SonarQube** (opcional) para detectar problemas de estilo, complejidad y posibles vulnerabilidades.
- **Reporte detallado** de resultados para informar al equipo sobre el estado del build y la cobertura de pruebas.

4. Pautas para la revisión de código

El flujo de trabajo para revisión se basa en Pull Requests:

- Todo cambio debe ser introducido mediante un Pull Request hacia la rama **master**.
- Al menos un miembro del equipo debe revisar el código, verificar que todos los checks del pipeline hayan pasado y aprobar la solicitud.
- Si se detectan problemas, se solicitan cambios antes de poder hacer merge.
- Solo después de la aprobación, se realiza el merge para garantizar la calidad y estabilidad del código en la rama principal.

5. Flujo de despliegue del aplicativo

- Una vez mergeado el Pull Request a la rama **master**, el webhook de ngrok activa el pipeline en Jenkins.
- Jenkins ejecuta automáticamente la compilación, pruebas y generación de artefactos.
- Si todo es exitoso, se procede con el despliegue automático a la infraestructura en **AWS**, utilizando servicios como **Elastic Beanstalk** o **ECS**, configurados como parte del pipeline.
- El despliegue incluye monitoreo post-implementación para asegurar que la aplicación esté funcionando correctamente en producción.
- En caso de errores en el build o despliegue, Jenkins notifica inmediatamente a los desarrolladores para su pronta resolución.

Diagrama

