Planificación del Pipeline de Integración y Entrega Continua

-Herramientas:

Para la garantización de la calidad y desarrollo del software durante su desarrollo e implementación, mi objetivo es diseñar un pipeline de integración y entrega continua que automaticen las pruebas, construcción y despliegue de código.

Para esto en primer lugar usare las GitHub actions, estas permiten la automatización del flujo de trabajo CI/CD dentro del repositorio de GitHub.

Después otra herramienta a utilizar será Docker, este permite automatizar la implementación de aplicaciones en contenedores que son entornos ligeros y portables que facilitan la creación despliegue y ejecución de aplicaciones en diferentes entornos sin que haya problemas de compatibilidad.

Dentro de las acciones de GitHub puedo utilizar JUnit y SonarQube para la correcta integración del código. JUnit permite ejecutar pruebas automatizadas para verificar que el código funciona como se espera, después SonarQube puede analizar el código para identificar vulnerabilidades, errores y problemas de mantenimiento, este genera métricas y reportes de calidad del código.

Para la compilación del código me decidido por Maven ya que facilita la gestión de dependencias, proporciona estructura de proyecto estándar, ofrece ciclo de vida de construcción predefinido etc.

-Flujo del Pipeline:

1) Cuando se realice un cambio en el código, se realizará commit y push para subir los cambios a GitHub activando así las acciones automáticamente.

2)Una vez se activa las acciones, primero se ejecutarán Gradle build para copilar el código, después entrara en ejecución las pruebas unitarias con JUnit para la verificación del correcto funcionamiento del código.

3) Una vez realizada las pruebas SonarQube puede analizar el código antes de fusionar los cambios con el MVP final para asegurar que estos cumplen con los estándares de calidad definidos.

4) Creación de contenedor Docker, para que la ejecución se ejecute en un entorno de desarrollo controlado.

5) Despliegue en un entorno de prueba, el software se implementará el entorno para unas posibles pruebas adicionales.

6) Por último si todas las pruebas son correctas, el código será implementado automáticamente al repositorio listo para producción.

Bibliografía:

GitHub Actions: https://dev.to/n3wt0n/5-top-reasons-to-use-github-actions-for-your-next-project-cga

Docker: https://docs.docker.com/ ; https://www.dimensiona.com/es/que-es-docker-y-cuales-son-sus-ventajas/

SonarQube: https://sentrio.io/blog/que-es-sonarqube/ ; https://formadoresit.es/sonarqube-que-es-y-como-funciona/

JUnit: https://junit.org/junit5/docs/current/user-guide/

Maven: https://maven.apache.org/guides/introduction/introduction-to-the-pom.html ; https://www.campusmvp.es/recursos/post/java-que-es-maven-que-es-el-archivo-pom-xml.aspx