TALLER #2

Crear un proyecto en Java que incluya pruebas unitarias y configurar un workflow de **GitHub Actions** para ejecutarlas automáticamente. Luego, simular la ejecución del workflow en tu máquina local utilizando **nektos/act** y **Docker**.

📝 **Instrucciones**

1. **Crea un proyecto Java con Maven**
    - Debe incluir una clase sencilla (por ejemplo, Calculator) con al menos dos métodos (suma y división).
    - Implementa validaciones simples, como lanzar una excepción si intentan dividir por cero.
2. **Crea pruebas unitarias con JUnit 5**
    - Debes cubrir los métodos que creaste en la clase.
    - Asegúrate de incluir un test que valide el manejo de excepciones (por ejemplo, división entre cero).
3. **Configura el workflow de CI en GitHub Actions**
    - El workflow debe:
        - Ejecutarse en cada push y pull_request hacia la rama main.
        - Instalar JDK 17.
        - Ejecutar mvn clean test para correr las pruebas.
4. **Prepara el entorno para ejecutarlo localmente con act**
    - Crea un Dockerfile que tenga Maven y JDK 17.
    - Construye la imagen y ejecuta el workflow localmente usando act.
5. **Valida el resultado**
    - Ejecuta las pruebas con act y asegúrate de que pasen todas.

    - Simula un error (por ejemplo, cambiando un valor esperado en una prueba) y verifica que el workflow falle localmente.

📦 **Entregables**

- Código fuente del proyecto Java (clase + pruebas unitarias).
- Archivo pom.xml con las dependencias necesarias.
- Workflow de GitHub Actions en .github/workflows/ci.yml.
- Dockerfile para correr el workflow en act.
- Evidencia (capturas o logs) de:
    - Ejecución exitosa del workflow con act.
    - Ejecución fallida del workflow al romper intencionalmente un test.

💡 **Puntos a evaluar**

- Correcta implementación de las pruebas unitarias.
- Configuración adecuada del workflow.
- Capacidad para correr el CI localmente con act y Docker.
- Documentación clara (README con instrucciones de uso).